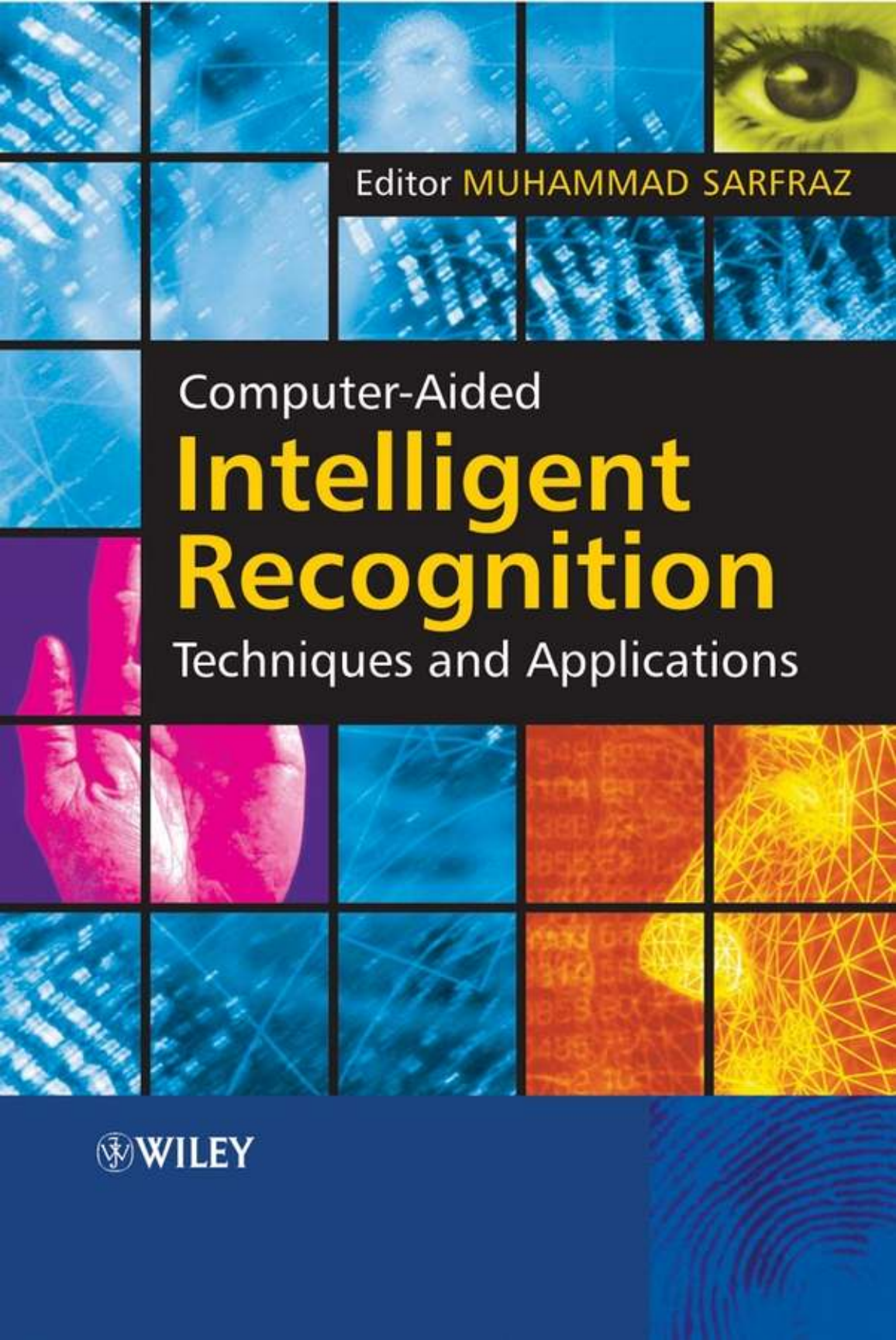

Editor MUHAMMAD SARFRAZ
Computer-Aided
Intelligent
Recognition
Techniques and Applications
WILEY

# COMPUTER-AIDED INTELLIGENT RECOGNITION TECHNIQUES AND APPLICATIONS

Edited by

**Muhammad Sarfraz**
*King Fahd University of Petroleum and Minerals, Kingdom of Saudi Arabia*

John Wiley & Sons, Ltd

# COMPUTER-AIDED INTELLIGENT RECOGNITION TECHNIQUES AND APPLICATIONS

# COMPUTER-AIDED INTELLIGENT RECOGNITION TECHNIQUES AND APPLICATIONS

Edited by

**Muhammad Sarfraz**
*King Fahd University of Petroleum and Minerals, Kingdom of Saudi Arabia*

John Wiley & Sons, Ltd

Telephone (+44) 1243 779777

Email (for orders and customer service enquiries): cs- books@wiley.co.uk
Visit our Home Page on www.wiley.com



***Other Wiley Editorial Offices***

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 33 Park Road, Milton, Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01, Jin Xing Distripark, Singapore 129809

John Wiley & Sons Canada Ltd, 22 Worcester Road, Etobicoke, Ontario, Canada M9W 1L1

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

***Library of Congress Cataloging-in-Publication Data***

Sarfraz, Muhammad.
Computer-aided intelligent recognition techniques and applications / Muhammad Sarfraz.
p. cm.
Includes bibliographical references and Index.
ISBN-13 978-0-470-09414-3 (cloth)
ISBN-10 0-470-09414-1 (cloth)
1. Logic circuits—Computer-aided design. 2. Pattern perception. I. Title.
TK7868.L6S257 2005
006.3—dc22
2004030987

***British Library Cataloguing in Publication Data***

A catalogue record for this book is available from the British Library

ISBN-13 978-0-470-09414-3 (HB)
ISBN-10 0-470-09414-1 (HB)

Typeset in 9/11pt Times by Integra Software Services Pvt. Ltd, Pondicherry, India
Printed and bound in Great Britain by Antony Rowe Ltd, Chippenham, Wiltshire
This book is printed on acid-free paper responsibly manufactured from sustainable forestry in which at least two trees are planted for each one used for paper production.

# Contents

# Preface

Intelligent recognition techniques, applications, systems and tools are extremely useful in a number of academic and industrial settings. Specifically, intelligent recognition plays a significant role in multidisciplinary problem solving. It is extremely useful for personal identification in various situations like security, banking, police, postal services, etc. In particular, various problems like character recognition, iris recognition, license plate recognition, fingerprint recognition, signature recognition, recognition in medical pattern understanding, mine recognition, and many others can be intelligently solved and automated. In addition to its critical importance in the traditional fields of character recognition, natural language processing and personal identification, more recently, intelligent recognition methods have also proven to be indispensable in a variety of modern industries, including computer vision, robotics, medical imaging, visualization, and even media.

This book aims to provide a valuable resource, which focuses on interdisciplinary methods and affiliated up-to-date methodologies in the area. It aims to provide the user community with a variety of techniques, applications, systems and tools necessary for various real-life problems in areas such as:

- ***Information Technology***
- ***Education***
- ***Security***
- ***Banking***
- ***Police***
- ***Postal services***
- ***Manufacturing***
- ***Mining***
- ***Medicine***
- ***Multimedia***
- ***Entertainment***
- ***Communications***
- ***Data visualization***
- ***Knowledge extraction***
- ***Pattern classification***
- ***Pose estimation***
- ***Virtual reality***
- ***Others***

It aims to collect and disseminate information in various disciplines including:

- ***Pattern recognition***
- ***Artificial Intelligence***
- ***Computer vision***
- ***Computer graphics***
- ***Image processing***
- ***Neural networks***
- ***Cryptography***
- ***Fuzzy logic***
- ***Databases***
- ***Evolutionary algorithms***
- ***Shape analysis and description***
- ***Numerical analysis***
- ***Others***

The major goal of this book is to stimulate views and provide a source from which students, researchers and practitioners can find the latest developments in the field of intelligent recognition. Due to speedy scientific developments, there is a great deal of thirst within the scientific community worldwide to be equipped with state-of-the-art and up-to-date theory and practice to get their problems solved in diverse areas of various disciplines. Although a good deal of work has been done by

researchers, yet interest is increasing tremendously everyday due to complicated problems being faced in academia and industry.

The twenty-four chapters within this book will cover computer-aided intelligent recognition techniques, applications, systems and tools. The book is planned to be of most use to researchers, computer scientists, practising engineers, and many others who seek state-of-the-art techniques, applications, systems and tools for intelligent recognition. It will also be equally and extremely useful for undergraduate senior students, as well as graduate students, in the areas of computer science, engineering, and other computational sciences.

The editor is thankful to the contributors for their valuable efforts towards the completion of this book. A lot of credit is also due to the various experts who reviewed the chapters and provided helpful feedback. The editor is happy to acknowledge the support of King Fahd University of Petroleum and Minerals towards the compilation of this book. The project has been funded by King Fahd University of Petroleum and Minerals under Project # ICS/INT.RECOGNITION/271.

***M. Sarfraz***

# List of Contributors

Ahmed, M. J.
Aleemuddin, M.
Al-Hajj, R.
Ali, S. S. A.
Anuzis, P.
Arana-Daniel, N.
Bayro-Corrochano, E.
Bazen, A.
Bloch, I.
Bote-Lorenzo, M. L.
Chen, K-Z.
Chen, M.
Chia, H-F.
Deriche, M.
Dimitriadis, Y. A.
Elnagar, A.
Feng, X. A.
Fiaidhi, J. A. W.
Gelman, L.
Gerez, S.
Gómez-Sánchez, E.
Ha, J. E.
Han, C-C.
Hasnah, A.
Iglesias, A.
Jaam, J.
Jaoua, A.
Kang, D. J.
Khare, R.
Kweon, I. S.
Li, B.
Milisavljević, N.
Mohammed, S. M. A.
Moinuddin, M.
Nawaz, S. N.
Ogiela, M. R..
Ou, Z-Y.
Perucchio, R.

Pham, T. D.
Sanderson, M.
Sarfraz, M.
Sedmera, D.
Sharma, R.
Tadeusiewicz, R.
Thompson, C.
Thompson, R. P.
Veldhuis, R.
Wang, X.
Wong, E. K.
Xie, M.
Xie, W
Yang, J.
Yang, L.
Yeasin, M.
Zhang, D.
Zhang, X-W.
Zidouri, A.

# 1

# On Offline Arabic Character Recognition

**Muhammad Sarfraz**
Department of Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

**Abdulmalek Zidouri**
Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

**Syed Nazim Nawaz**
Information Technology Department, College of Business Administration, Jeddah 21361, Kingdom of Saudi Arabia

*Machine recognition of characters has received considerable research interest in the area of pattern recognition in the past few decades. This chapter presents the design and implementation of a system that recognizes machine printed Arabic characters. The proposed system consists of four stages: preprocessing of the text, segmentation of the text into individual characters, feature extraction using the moment invariant technique and recognition of characters based on two approaches. In the preprocessing of the text we deal with two problems: isolated pixel removal and drift detection and correction. Next, the given text is segmented into individual characters using horizontal and vertical projection profiles. Moment invariants are used for feature extraction for each character. Finally, the system is trained and the characters are recognized. Recognition of characters is attempted using two approaches, a syntactic approach and a neural network approach.*

## 1. Introduction

In the past two decades, valuable work has been carried out in the area of character recognition and a large number of technical papers and reports have been devoted to this topic. Character recognition systems offer potential advantages by providing an interface that facilitates interaction between man

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

and machine. Some of the applications of OCR include automatic processing of data, check verification and a large variety of banking, business and scientific applications. OCR provides the advantage of little human intervention and higher speed in both data entry and text processing, especially when the data already exists in machine-readable forms.

With all the above advantages considered, Arabic character recognition proves to be an interesting area for research. Many papers concerned with Optical Character Recognition (OCR) have been reported in the literature [1]. Several recognition techniques have been used over the past few decades by many researchers. These techniques were applied for the automatic recognition of both machine and hand-printed characters. An immense amount of research has been carried out on the recognition of Latin and Chinese characters. Against this background, only few papers have addressed the problem of Arabic character recognition. One of the main reasons behind this is the difficulty involved in processing printed Arabic text. The connectivity and variant shape of characters in different word positions present significant difficulty in processing of the text. Some of the features of Arabic script that have limited the research in this area are the following:

- Arabic script constitutes 28 characters, in addition to ten numerals.
- Each character can appear in four different shapes/forms depending on the position of the word (Beginning form, BF; Middle form, MF; Isolated form, IF; and End form, EF). Table 1.1 shows some Arabic characters in their different forms.
- The Arabic characters of a word are connected along a baseline. A baseline is the line with the highest density of black pixels. This calls for different segmentation methods from those used in other unconnected scripts.
- Characters in Arabic script are connected even when typed or printed.
- In addition to connectivity, vowel diacritic signs are an essential part of written Arabic. Vowels are represented in the form of overscores or underscores (see Figure 1.1).

**Table 1.1** Different forms of Arabic characters.

| IF | BF | MF | EF | IF | BF | MF | EF |
|---|---|---|---|---|---|---|---|
| أ | أ | ـا | ـا | ظ | ظـ | ـظـ | ـظ |
| ب | بـ | ـبـ | ـب | ك | كـ | ـكـ | ـك |
| ت | تـ | ـتـ | ـت | ل | لـ | ـلـ | ـل |
| ث | ثـ | ـثـ | ـث | م | مـ | ـمـ | ـم |
| ج | جـ | ـجـ | ـج | ن | نـ | ـنـ | ـن |
| ح | حـ | ـحـ | ـح | ع | عـ | ـعـ | ـع |
| خ | خـ | ـخـ | ـخ | غ | غـ | ـغـ | ـغ |
| د | د | ـد | ـد | ف | فـ | ـفـ | ـف |
| ذ | ذ | ـذ | ـذ | ق | قـ | ـقـ | ـق |
| ر | ر | ـر | ـر | س | سـ | ـسـ | ـس |
| ز | ز | ـز | ـز | ش | شـ | ـشـ | ـش |
| ص | صـ | ـصـ | ـص | ه | هـ | ـهـ | ـه |
| ض | ضـ | ـضـ | ـض | و | و | ـو | ـو |
| ط | طـ | ـطـ | ـط | ي | يـ | ـيـ | ـي |

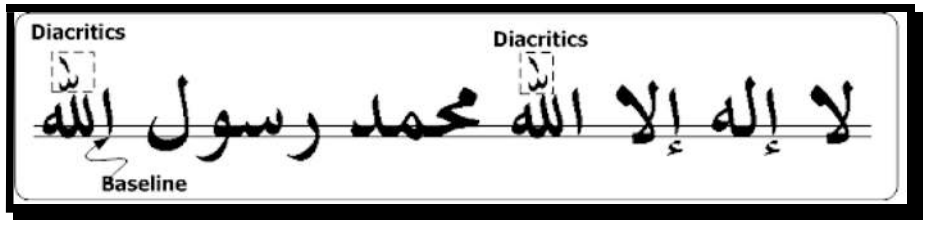



**Figure 1.1** The baseline and diacritics of a given Arabic text.

**Figure 1.2** Example of different characters with the same body.

- Many Arabic characters have dots, which are positioned above or below the letter body. Dots can be single, double or triple. Different Arabic letters can have the same body and differ in the number of dots identifying them, as shown in Figure 1.2.
- The problem of overlapping makes the problem of determination of spacing between characters and words difficult.
- Characters in an Arabic script usually overlap with their neighboring letters depending on their position in the word. The degree of overlapping varies depending on the size and style of the character.

Figure 1.3 depicts some of the features of Arabic script. As shown in the figure, the ligature in the beginning and the middle of the figure consists of two vertically stacked characters. The text baseline is between the two horizontal lines. The text shown in Figure 1.3 is a text with diacritics.

These and other special characteristics make it impossible to directly adapt English or other text recognition systems to Arabic text recognition.

The rest of the chapter is organized as follows: Section 2 describes the structure of the proposed OCR system and the work that has been carried out in this area. The preprocessing stage is explained in Section 3. Section 4 discusses the proposed method for segmentation of text into individual characters. Section 5 explains feature extraction based on the moment invariant technique by Hu [2]. Section 6

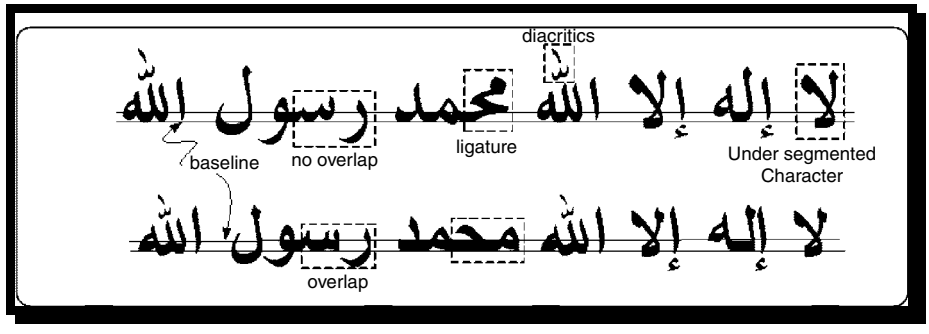



**Figure 1.3** The characteristics of an Arabic text.

discusses the two approaches that have been used for recognition of characters. Finally, Section 7 discusses experimental analysis and the chapter is concluded in Section 8.

## 2. Structure of the Proposed OCR System

Figure 1.4 shows the block diagram of the proposed character recognition system. The system involves four image-processing stages: Preprocessing, Segmentation, Feature Extraction and Classification. The recognition of any script starts by acquiring a digitized image of the text using a suitable scanning system. Next, the preprocessing of the image takes place. There are two processes for handling the acquired image in the proposed system: drift correction and removal of isolated pixels. In the second stage, the segmentation of the text into individual characters takes place. Segmentation of connected script text into individual characters is the most important and difficult stage in an Arabic optical character recognition system. To achieve good recognition rate and performance, a character recognition system should have a good segmentation algorithm.

Since the early eighties there have been reports about successful research projects in the field of printed Arabic character recognition. Connectivity of characters being an inherent property of Arabic writing mean that it is of primary importance to tackle the problem of segmentation in any potentially practical Arabic OCR system. A state of the art on offline Arabic character recognition can be found in [1].

Segmentation of the text into individual characters can be achieved using a variety of techniques. Different approaches for the segmentation of the text in to individual characters are presented in [3]. Segmentation of text in to characters can be achieved based on the geometrical and topological features of the characters [4,5,6], closed contours [7] and horizontal and vertical projection of pixel rows and columns [8–11].

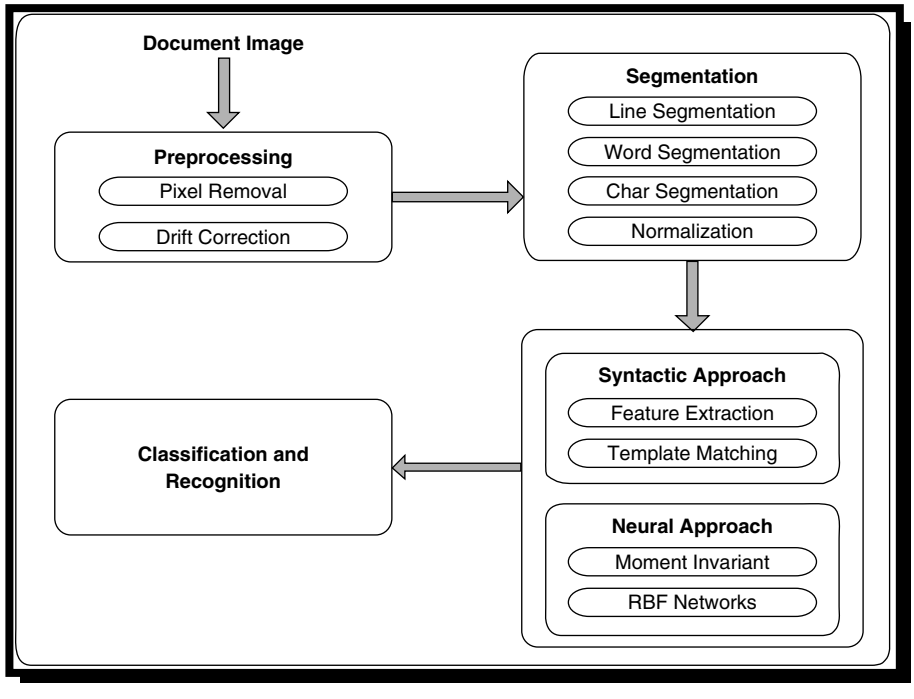



**Figure 1.4** Structure of the proposed OCR system.

Almuallim and Yamaguchi [4] proposed a structural recognition technique for Arabic handwritten words. In their approach, an Arabic word is segmented into strokes which are classified based on their geometrical and topological properties. The relative positions of the classified strokes are examined and the strokes are combined in several steps into the string of characters that represents the recognized word.

Segmentation of text can also be achieved using closed contours. The SARAT system [7] used outer contours to segment Arabic words into characters. The word is divided into a series of curves by determining the start and end points of words. Whenever the outer contour changes sign from positive to negative, a character is segmented.

Hamami and Berkani [8] employed a simple segmentation method. Their method is based on the observation of projections of pixel rows and columns, and uses these projections to guide the segmentation process. In their method, undersegmentation of characters, which is a common problem, is treated by considering the entire undersegmented set of characters as a single character. On the other hand, the other common problem of oversegmentation is solved by taking care of the situations in which it may occur and resolving them accordingly.

After segmentation, numerical features of the character are extracted. Features of the characters are extracted and matched with the existing ones to identify the character under investigation. Trier *et al.* [12] present a survey of the different feature extraction methods available in the literature. Features can be extracted using template matching [13], closed contours [7], zoning [14] and moment invariants [2,4,15–18].

Fakir and Hassani [19,20] utilized moment invariant descriptors developed by Hu [2] to recognize the segmented Arabic printed characters. A look-up table is used for the recognition of isolated handwritten Arabic characters [21]. In this approach, the character is placed in a frame which is divided into six rectangles and a contour tracing algorithm is used for coding the contour as a set of directional vectors using a Freeman Code. Jambi [22] adopted a look-up table for the recognition of isolated Arabic characters.

Using nonlinear combinations of geometric moments, Hu [2] derived a set of seven invariant moments, which has the desirable property of being invariant under image translation, scaling and rotation. Abdul Wahab [15] has shown, based on the work of Hu [2], that there exists a set of invariant momentum features that can be used for classification purposes.

The step following the segmentation and extraction of appropriate features is the classification and recognition of characters. Many types of classifier are applicable to the OCR problem. Recognition of a pattern can be done using a statistical approach (decision theoretic), a syntactic approach or a neural approach. Among the three, syntactic and neural approaches are showing promising results compared to the statistical approach.

Chinveerphan *et al.* [23] and Zidouri *et al.* [11,24,25] presented a structural approach for the recognition of Arabic characters and numerals. Their approach is based on the modified Minimum Covering Run (MCR) expression. The given Arabic text is represented in its MCR form and its feature values are extracted. From the feature values, the different character shapes are described and the reference prototypes are built. Finally, the character is identified by matching the data of the document image with the reference prototypes.

Sadoun *et al.* [26] proposes a new structural technique for the recognition of printed Arabic text. Here, the recognition is done by parsing the sentence in which the given image occurs. The advantage of this technique is that it is font independent.

Among the many applications that have been proposed for neural networks, character recognition has been one of the most successful. Compared to other methods used in pattern recognition, the advantages most often stated in favor of a neural network approach to pattern recognition are:

- it requires less input of knowledge about the problem than other approaches;
- it is amenable to high-performance parallel-processing implementation;
- it provides good fault tolerance compared to statistical and structural approaches [1].

Altuwaijri *et al.* [16] used a multilayer perceptron network with one hidden layer and back-propagation learning to classify a character. Their input layer consisted of 270 neurons. Smagt [27] tested the performance of three general purpose neural networks: the feed-forward network, the Hopfield network and the competitive learning network for OCR applications. The classification capabilities of the network are compared to the nearest neighbor algorithm applied to the same feature vectors. The feed-forward and the competitive learning networks showed better recognition rates compared to the Hopfield network. Each of the three techniques is based on learning approaches where an adjustment to the weight is made following each iteration.

One problem with the neural networks approach is that it is difficult to analyze and fully understand the decision making process [28]. It is often quite difficult to analyze which kind of classifier is most suitable for a recognition process. An unbiased comparison between each of the recognition processes is quite difficult and many conditions need to be fulfilled. To have an efficient comparison between the three approaches, the best statistical or structural classifiers have to be compared with the same quality neural network classifier [29]. In this work we try to compare the results of the syntactic approach and the neural network approach for the Arabic OCR problem. The first two stages of preprocessing and segmentation are common to both approaches. The features used and recognition stage has been attempted using the two approaches. Results and discussion about the two methods are illustrated. The syntactic approach gave a higher recognition rate than the neural network approach. Recommendations on how to improve the recognition rate and performance of the system are given at the end of this chapter.

## 3. Preprocessing

In the preprocessing stage for this work we concentrate on removal of non-useful information that can be considered as noise and skew detection and correction. To remove the noise as isolated pixels for any given pixel, we check for the existence of a neighboring pixel in all the possible eight directions (Figure 1.5). If a pixel exists in any of the possible directions, then the pixel is not an isolated pixel. However, if there is no pixel in any of these directions, then the pixel under investigation is considered to be an isolated pixel and is removed.

The text to be recognized may be transferred to the system slanted. This affects the accuracy of segmentation and recognition. To tackle the problem of skew detection and correction we have employed the following drift correction procedure. First we determine the rotation angle of the text by computing the tangents of all the line segments that can be constructed between any pair of black pixels in the image. Then, the corresponding angles are computed. To reduce the computation cost, one could apply this process just to a selected window of text instead of the whole image, assuming that the whole page is skewed in the same way everywhere.

The angle that has the highest number of occurrences is assumed to be the angle of rotation of the image. After determining the angle of rotation, the baseline drift is corrected by rotating the image by

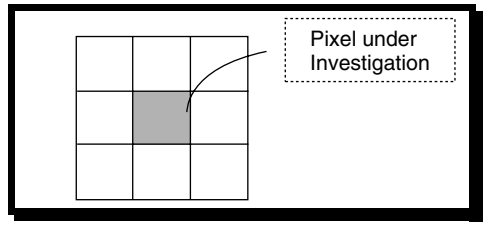
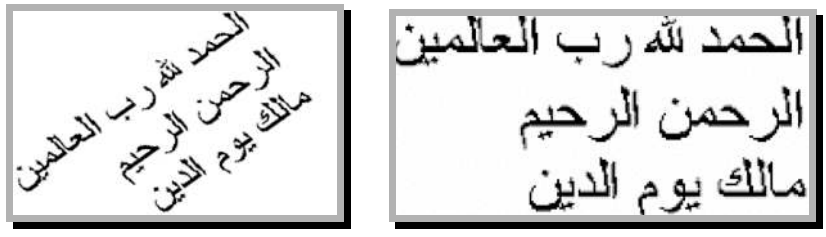



**Figure 1.5** Eight-connected components.




(a) (b)

**Figure 1.6** (a) Original image skewed by 36°; (b) image after drift correction.

the same angle in the opposite direction. Figure 1.6(a) and Figure 1.6(b) show the original image and the result obtained by applying the preprocessing on an image that is severely skewed by 36 degrees with respect to the horizontal.

## 4. Segmentation

Segmentation is the process of isolating the individual characters to be passed to the recognition phase. Character segmentation is the most crucial and the most difficult step in Arabic OCR. A poor segmentation process produces misrecognition or rejection. The segmentation process for the character recognition problem can be divided into three levels: line segmentation, word segmentation and character segmentation. Figure 1.7 illustrates the entire segmentation process for the proposed system.

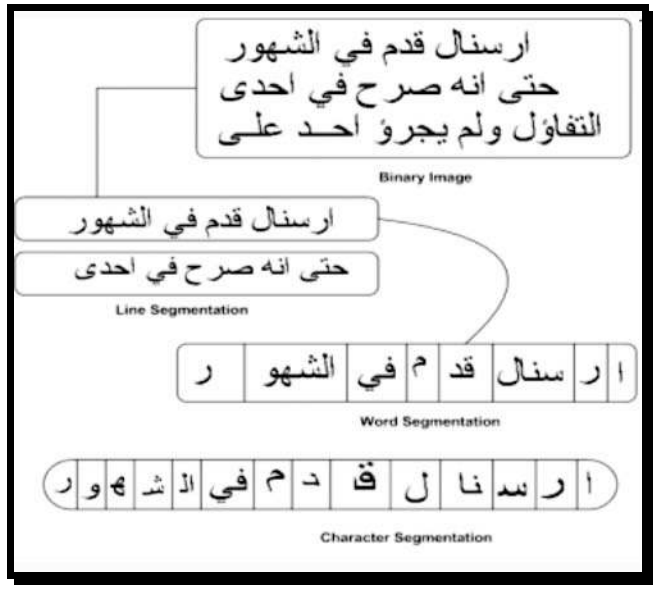



**Figure 1.7** The overall segmentation process.

## *4.1 Line Segmentation and Zoning*

After preprocessing, the given Arabic text is first segmented into individual lines of text. Segmentation of text into lines is done using the horizontal projection profile of an image. This is popular because it is efficient and easily implemented. Next, the segmented line of text is divided into four *zones*; namely the baseline zone, the middle zone, the upper zone and the lower zone (Figure 1.8).

### 4.1.1 Line Segmentation Using Horizontal Projection

First we determine the size of the image. For an image of a particular size we project horizontally along each row of the image. If the number of black pixels encountered during horizontal projection is zero, then we just increment the row number, i.e. we project horizontally (determine the number of black pixels) in the next row. This process is repeated until the number of black pixels along a row is not equal to zero. When this happens we initialize the row number to a variable. The moment it happens it means that a line of text has just started. We again repeat the above process but this time we check for a row number with zero pixels. The occurrence of a row number with zero pixels indicates the end of the line of text. The values of the row numbers indicate the starting and ending positions of a line in the text, and the line is extracted.

### 4.1.2 Zone Classification

The baseline zone is the zone with the highest density of black pixels. In Figure 1.8 this zone can be identified as the area within the horizontal lines of the histogram. The region just above the baseline zone is the middle zone and then the upper zone. Anything below the baseline is classified to belong to the lower zone. Any zone that is just above the baseline and twice the thickness of the baseline is the middle zone. This middle zone is useful for segmenting the words into individual characters. Figure 1.8 gives the horizontal projection profile of a sample Arabic text.

## *4.2 Word Segmentation*

Next, the line of text is segmented into words or subwords. This is done using the vertical projection of the image. Segmentation of a line of text into words is simple and is similar to the horizontal projection. The only difference between the horizontal and vertical projections is that in the latter, we count the number of black pixels on each column of the image line by line of text. If the number of black pixels is not equal to zero, it indicates a connected part and consequently the text is not segmented. On the other hand, if the number of black pixels in a particular column is equal to zero, the word is segmented. Figure 1.9 shows the vertical projection profile for a given image.

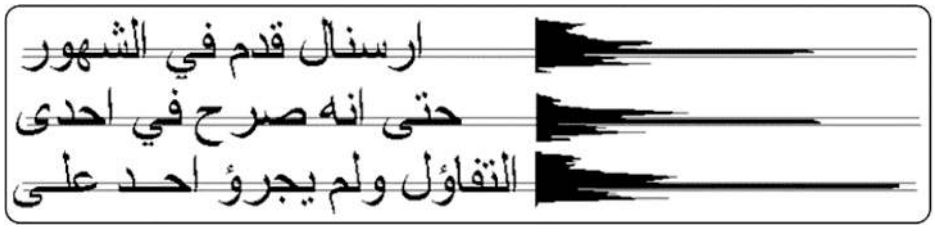



**Figure 1.8** Horizontal projection profile for a given Arabic text.


**Figure 1.9** Vertical projection profile for a given Arabic text.

## *4.3 Segmentation of Words into Individual Characters*

Finally, the word or subword is segmented into characters. First, the vertical projection of the middle zone is created. The middle zone is the zone right above the baseline. Next, the word is scanned from right to left. A fixed threshold is used for segmenting the word into characters. Whenever the value of the vertical profile of the middle zone is less than two thirds of the baseline thickness, the area is considered a connection area between two characters. Then, any area that has a larger value is considered as the start of a new character, as long as the profile is greater than one third of the baseline. This process is repeated until the full length of the line is exhausted.

The overall segmentation procedure is as follows:

*Procedure 1: Line segmentation*

```
1. Identify the height and width of the image.
2. Traverse from top to bottom of the entire image.
3. For each row, identify the number of black pixels.
   While the number of black pixels is zero:
   (a) Increment row number;
   (b) Move until the number of black pixels is not equal to zero;
   (c) Initialize the row number to startx.
   While the number of black pixels is not zero:
   (a) Increment startx.
   (b) If the number of black pixels is zero:
   (a) Assign startx to endx.
   (b) Row numbers between startx and endx indicate a line of text.
4. End for.
```

*Procedure 2: Word and character segmentation*

```
1. Project vertically and repeat the above process to segment lines into
   individual parts or words.
2. For each word:
   (a) Project horizontally and determine the rows with the highest value.
       This zone of large row values is the baseline zone.
   (b) Middle zone = 2*baseline zone thickness.
   (c) If vertical projection(middle zone area) > 1/2 (baseline zone) or
       < 2/3 (baseline zone) The area is a connection area.
       Else isolate the character.
       End if.
3. End for.
```

## 5. Feature Extraction

The next stage in our Arabic recognition system is the feature extraction stage. This is a very important stage in any recognition system. The performance of the classifier depends directly on the feature selection and extraction. The main advantage of feature extraction is that it removes redundancy from the data and represents the character image by a set of numerical features. These features are used by the classifier to classify the data.

In our implementation, moment invariants used by Hu [2] have been utilized to build the feature space. Using nonlinear combinations of geometric moments, we derived a set of invariant moments which has the desirable property of being invariant under image translation, scaling and rotation.

The central moments, which are invariant under any translation, are defined as

$$M_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\overline{x})^p (y-\overline{y})^q f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{1.1}$$

where

$$\left.\begin{aligned} \overline{x} &= \frac{\overline{M}_{10}}{\overline{M}_{00}}, \overline{y} = \frac{\overline{M}_{01}}{\overline{M}_{00}} \quad \text{and} \\ \overline{M}_{pq} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)\,\mathrm{d}x\mathrm{d}y \end{aligned}\right\} \tag{1.2}$$

However, for images, the continuous image intensity function $f(x, y)$ is replaced by a matrix, where $x$ and $y$ are the discrete locations of the image pixels. The integrals in Equations (1.1) and (1.2) are approximated by the summations:

$$M_{pq} = \sum_{x=0}^{m}\sum_{y=0}^{n} (x-\overline{x})^p (y-\overline{y})^q f(x,y)\,\mathrm{d}x\mathrm{d}y \tag{1.3}$$

$$\overline{M}_{pq} = \sum_{x=0}^{m}\sum_{y=0}^{n} x^p y^q f(x,y)\,\mathrm{d}x\mathrm{d}y \tag{1.4}$$

where $m$ and $n$ are dimensions of the image. The set of moment invariants that has been used by Hu are given by:

$$\phi_1 = M_{20} + M_{02} \tag{1.5a}$$

$$\phi_2 = (M_{20} - M_{02})^2 + 4M_{11}^2 \tag{1.5b}$$

$$\phi_3 = (M_{30} - 3M_{12})^2 + (3M_{21} - M_{03})^2 \tag{1.5c}$$

$$\phi_4 = (M_{30} + M_{12})^2 + (M_{21} + M_{03})^2 \tag{1.5d}$$

$$\begin{aligned} \phi_5 = {} & (M_{30} - 3M_{12})(M_{30} + M_{12})[(M_{30} + M_{12})^2 - 3(M_{21} + M_{03})^2] \\ & + (3M_{12} - M_{03})(M_{21} + M_{03}) * [3(M_{30} + M_{12})^2 - (M_{21} + M_{03})^2] \end{aligned} \tag{1.5e}$$

$$\begin{aligned} \phi_6 = {} & (M_{20} - M_{02})[(M_{30} + M_{12})^2 - (M_{21} + M_{03})^2] \\ & + 4M_{11}(M_{30} + M_{12})(M_{21} + M_{03}) \end{aligned} \tag{1.5f}$$

$$\begin{aligned} \phi_7 = {} & (3M_{21} - M_{03})(M_{30} + M_{12})[(M_{30} + M_{12})^2 - 3(M_{21} + M_{03})^2] \\ & + 3(M_{21} - M_{03})(M_{21} + M_{03}) * [3(M_{30} + M_{12})^2 - (M_{21} + M_{03})^2] \end{aligned} \tag{1.5g}$$

**Table 1.2** Moment invariants represented by a $4 \times 4$ array.

| $M$ | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | X | X | √ | √ |
| 1 | X | √ | √ | X |
| 2 | √ | √ | X | X |
| 3 | √ | X | X | X |

**Table 1.3** Moment invariant values for some characters.

| | ة | ي | ف | ظ | ا |
|---|---|---|---|---|---|
| $\phi_1$ | −1.180 | −1.422 | −1.595 | −1.536 | 0.375 |
| $\phi_2$ | −4.027 | −5.823 | −4.790 | −6.244 | 0.636 |
| $\phi_3$ | −7.293 | −9.088 | −7.642 | −8.935 | −2.668 |
| $\phi_4$ | −6.459 | −8.621 | −10.190 | −13.024 | −2.804 |
| $\phi_5$ | −13.338 | −17.739 | −19.181 | −24.368 | −5.541 |
| $\phi_6$ | −8.576 | −11.578 | −12.584 | −16.325 | −2.582 |
| $\phi_7$ | −13.175 | −17.404 | −20.016 | −23.981 | −9.058 |

These functions can be normalized to make them invariant under a scale change by using the normalized central moments instead of the central moments. The normalized central moments are defined by

$$m_{pq} = \frac{M_{pq}}{M_{00}^{a}} \text{ where } a = \frac{(p+q)}{2} + 1 \tag{1.6}$$

These, when substituted into the above equations, will give seven moments which are invariant to translation, scale change and rotation.

The $\phi$s have large dynamic values. Thus, it was found that it was more practical to deal with the logarithms of the magnitudes of the $\phi$s [15] Thus, the seven moment invariants used in the proposed system are replaced by their logarithmic values.

In the final implementation to consider all the four possible positions of the characters, the four shapes of the letter are represented in the feature space. From Table 1.2 it can be noted that for seven moment invariants if a $4 \times 4$ matrix is constructed, only moments $M_{02}, M_{03}, M_{11}, M_{12}, M_{20}, M_{21}, M_{30}$ (upper diagonal elements) are used. However, if the number of moments is increased, other cells are also expected to be filled. For each character, the above moment invariant descriptors are calculated and fed to the artificial neural network. Table 1.3 shows the rounded values of $\phi$ obtained for some of the characters in the training set. Each of the characters is a $100 \times 100$ binary image.

## 6. Recognition Strategy

The process of recognizing a pattern is basically dealt with by three possible approaches: statistical, syntactic and neural approaches [30]. The statistical approach is based on decision making (the probabilistic model), the syntactic approach deals with the structural description of pattern (formal grammar) and the neural approach is based on training the system with a large input data set and

storing the weights (stable state), which are used later for recognition of trained patterns. A survey on character recognition shows that the syntactic and neural approaches have gained widespread attention compared to the decision theoretic approach. This is due to the simplicity and efficiency of the first two approaches compared to the latter. Not surprisingly, in our system we also discuss the recognition of Arabic characters using the syntactic and neural approaches.

## *6.1 Recognition Using the Syntactic Approach*

In the structural approach, recognition is based on the stored structural pattern information. The information stored is the pixel value for the investigated character and the prototyped character. For the proposed system, the pattern of the character is stored as a matrix. Each cell stores the information related to each pixel of a particular character. The pixel value can be either 0 (indicating a black pixel), or 1 (indicating a white pixel). Recognition is done by matching the investigated character with the prototype character. The segmented character is passed through the following two stages to achieve recognition.

1. Normalization.
2. Recognition using template matching.

### 6.1.1 Normalization

In this phase, first the extracted characters are refined to fit the characters into a window without white spaces on all four sides. Figure 1.10(a) shows the template of the extracted character 'ح'. Now, each character image is normalized to a size of $30 \times 30$. Figure 1.10(b) shows the character 'ح' normalized to $30 \times 30$. The normalization is done using a window to view port transformation. This mapping is used to map every pixel of the original image to the corresponding pixel in the normalized image. The image obtained is invariant to scaling because the characters were refined (i.e. all the white spaces on the top, bottom, left and right were removed, so as to fit the character exactly into the window).

### 6.1.2 Template Matching using the Hamming Distance

Template matching for character recognition is straightforward and is reliable. This method is more tolerant to noise. In this approach, the templates are normalized to $30 \times 30$ pixels and stored in the

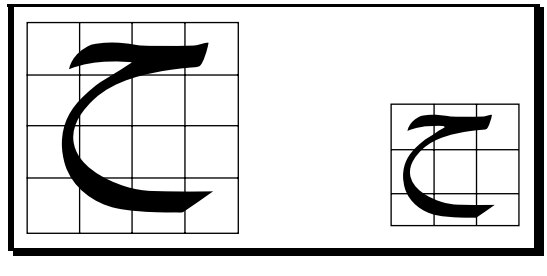


**Figure 1.10** (a) Original template for character 'ح' (b) *normalized* $30 \times 30$ template for character 'ح'.

database. The extracted character, after normalization, is matched with all the characters in the database using a Hamming distance approach. This approach is shown in Equations (1.7) and (1.8).

$$\sum_{i=1}^{nrows} \sum_{j=1}^{ncols} \text{mismatch}_{i,j} \tag{1.7}$$

where

$$\text{mismatch}_{i,j} = \begin{cases} 1, & \text{if original}_{i,j} \neq \text{extracted}_{i,j} \\ 0, & \text{if original}_{i,j} = \text{extracted}_{i,j} \end{cases} \tag{1.8}$$

where *nrows* and *ncols* are the number of rows and columns in the original and extracted images. In our case, $nrows = ncols = 30$, as the image is normalized to $30 \times 30$. The mismatches for each of the extracted characters are found by comparison with the original characters in the database, and the character with the least mismatch value is taken as the recognized character. The database consists of a total of 33 classes of character. There are 149 characters in all the 33 classes of character. In addition to the usual 28 classes of character, oversegmented ones (such as س) are also included in the database.

## 6.2 Recognition Using the Neural Network Approach

Characters are classified according to their computed modified moment invariants by means of artificial neural networks. Among the many applications that have been proposed for neural networks, character recognition has been one of the most successful.

Many neural network architectures have been used in OCR implementation. MLP is usually a common choice. Unfortunately, as the number of inputs and outputs grow, the MLP grows quickly and its training becomes very expensive. In addition, it is not easy to come up with a suitable network design and many trial-and-error cycles are required. Radial Basis Function (RBF) networks, on the other hand, offer better features for recognition applications. In addition to highly efficient training, no design details are required since the network automatically adjusts its single hidden layer.

However, before implementing the neural network, numerical features of the character are extracted and the system is trained to associate the segmented character with the desired character. In our implementation, we used the invariant moment technique by Hu [2] to build the feature space. This measure is invariant with respect to size, translation and scaling.

Characters are classified according to their computed moment invariants by means of artificial neural networks. In our method, we used a Radial Basis Function (RBF) network with a back-propagation error learning algorithm. In implementing the RBF network architecture, the Brain Construction Kit (BCK) has been a very helpful tool. BCK is a Java package developed in the public domain that enables users to create, train, modify and use Artificial Neural Networks (ANNs).

RBF networks have been successfully applied to solve some difficult problems by training them in a supervised manner with an error back-propagation algorithm [31]. This algorithm is based on the error correction learning rule. Basically, error back-propagation learning consists of two passes through the different layers of the network: a forward pass and a backward pass [31]. In the forward pass, an input vector is applied to the nodes in the input layer and its effect propagated through the network layer by layer. Finally, a set of output is produced as the actual response of the network. During the forward pass, the weights of the network are all fixed. During the backward pass, the weights of the network are all adjusted in accordance with the error correction rule, i.e. the actual response of the network is subtracted from the desired response to produce an error signal. This error is then back propagated through the network and the weights are adjusted to make the actual response of the network move closer to the desired response. Figure 1.11 shows the block diagram of the implemented RBF network.

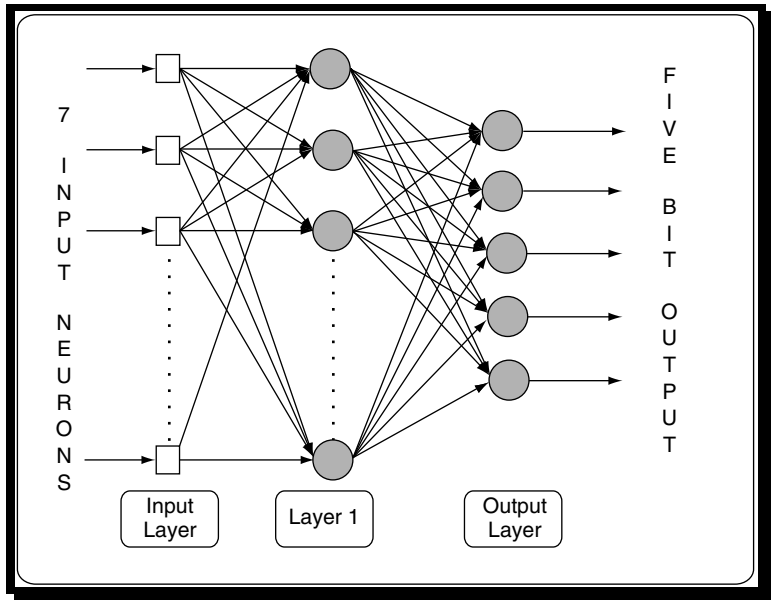



**Figure 1.11** Three-layered radial basis function network.

In an RBF network there is an important difference between a normal synapse and a BCKNetConnection. During a forward pass through a network, in which an input vector is fed into the input layer and each non-input neuron evaluates a response to the supplied input, synapses are used to transfer the output of one neuron to the input of another. BCKNetConnections cause the output of the source neuron to provide the output of the target neuron, that is, during a forward pass through cascaded networks, the output vector of the source network is used as the input vector for the target network, not as inputs to the neurons in the target network input layer. The activation is calculated as the Euclidean distance of the weight vector from the input vector, output is calculated using a Gaussian transfer function. This neuron contains an extra parameter, the standard deviation value, for use in the transfer function.

The RBF network consists of three layers, an input layer of dummy neurons, a hidden layer of radial neurons and an output layer of linear neurons. Unlike the other architectures, which contain a fixed number of nodes, the RBF architecture is dynamic, in that it adds neurons to its hidden layer as it is trained. There are a total of 20 nodes in the hidden layer in the implemented systems, determined heuristically. The input layer is composed of seven neurons. These seven input neurons are the seven moment invariant features extracted from the feature extraction phase. Based on the input features that are passed to the neural network, the output layer gives the output character belonging to a particular class. The number of the output depends on the number of the characters in the character set. The output layer consists of five neurons, each neuron represented by a bit '1' or '0'. For example, for the character 'ت', the output layer gives the output '00011'. This output is then converted to a decimal value and this decimal value represents a particular character of that class. For example, '00011' when converted to a decimal value gives a value 3, which is the index for the class of character 'ت'.

The learning time is reduced by partitioning the classes of characters into four sets, one for each character form (Arabic characters have four forms: beginning, middle, end and isolated). The training set is composed of a total of 33 classes of character. There are 149 characters in all the 33 classes of character. In addition to the usual 28 classes of character, oversegmented ones (such as س) are also

included in the training set. The training document that is passed to the neural network is a $100 \times 100$ character image.

## 7. Experimental Results and Analysis

This section presents the implementation details and the results obtained using the proposed system.

### *7.1 System Training*

In the syntactic approach, the training set is sampled data. The Hamming distance of the investigated character is matched with each character present in the sample data. The sampled data set consists of a total of 149 characters. To make the reference prototypes cover a wide range of possible variations without causing misrecognition, the sample data may be increased as new character samples are encountered. In the neural approach, the training of the system is based on the features extracted using the moment invariant technique. First, the RBF network computes an output for a given input. Next, the training of the system takes place using a weight adjustment strategy until the actual output converges with the desired output.

### *7.2 Experimental Set-up*

Experiments have been performed to test the above system. The developed Arabic text recognition system has been tested using randomly selected text. The system is designed in JDK 1.4.1 for the recognition of a popular font called Naskh font. The system developed is a single font, multisize system. The image to be tested is captured using a scanner and is passed to the system as a bitmap file. The system has been tested with many different sizes of Naskh font. The experiments for the above system were implemented under different situations, i.e. the experiments were carried out for both document images in normal shapes and images that are skewed with some angle of slant.

As stated earlier, classification and recognition of characters is done using both the structural approach and the neural approach. In the structural approach, after the text is segmented into characters, each segmented character is normalized to a $30 \times 30$ image. The normalized character is matched pixel by pixel with the characters present in the training set using the Hamming distance approach. The character with the least distance is considered to be the output character. Even though this technique is costly in terms of time, the accuracy of the system is greatly improved.

In the neural approach, first the features of the character are extracted and the system is trained for the extracted features. The RBF network adopted consists of three layers, an input layer of dummy neurons, a hidden layer of radial neurons and an output layer of linear neurons. The input to the system is the seven moment invariant features. The single hidden layer consists of 20 nodes. Based on the input features that are passed to the neural network, the output layer gives the output character belonging to a particular class. The training set is composed of a total of 33 classes of character. There are 149 characters in all the 33 classes of character. The training document that is passed to the neural network is a $30 \times 30$ character image.

### *7.3 Results Achieved*

The implemented system shows a recognition rate of 89 %–94 % using the structural approach, and an 83 % recognition rate using the neural network approach. When recognition is performed on small words or text with isolated characters ‘الحمد,يؤب , يؤم’, the recognition rate achieved using the structural approach is 98–100 %, and the recognition rate achieved using neural networks is 88 %.

The experimental results show that the shortcoming of the system using neural networks is mainly due to the closeness among the features of the characters. The features obtained using the moment invariant technique are not unique themselves, but closely related to the features of other characters. This prevents the neural network from clear classification. However, one possible solution is to improve the number of moments for better recognition. Dabi *et al.* [18] have shown that if the number of moments used in the feature extraction technique is increased to 11, the recognition rate can be improved considerably.

Figure 1.12(a) shows a test document used for verification of the system accuracy. The recognition rate achieved for this image using the syntactic approach was 94 %, and using the neural approach was 83 %. Most of the failures in the recognition of the text using the syntactic approach are due to improper segmentation of the text into individual characters. To improve the accuracy using the syntactic approach, one solution is to increase the number of classes in the training set. The test

(a)

يتحدث عـن احراز الثلاثية المكونة من بطولة الدوري والكأس المحلية ودوري ابطال اوروبا
خصوصـاً ان ارسنال قدم في الشهور الاولى من الموسم بعض اجمل العروض التي شهدتها
الكرة الانكليزية في الفترة الحديثة، حتى انه صرح في احدى المناسبات انه يتوقع ان يخوض
الفريق الموسم من دون اي هزيمة. ولم يشكك كثيرون بهذا التفاؤل ولم يجرؤ احـد علـى نعته بالمغر.
ولكن كيف خسر ارسنال لقباً كان بين قبضتيه.
بسهولة يقول فينغر ان الخروج من الدور الثاني من دوري الابطال كان السبب الرئيسي
وراء تقهقر معنويات اللاعبين والنتائج. وفعلاً، فان عدداً من النقاد رشح الفريق لاحراز
كأس الابطال في احدى الفترات لعروضه القوية خصوصاً بفوزه على روما خارج ارضه في بداية
الدور الثاني. لكن أربعة تعادلات متتالية وهزيمة في المباراة الحاسمة الاخيرة امام فالنسيا حطمت
معنويات اللاعبين وشكلت لهم صدمة وزعزعت ثقتهم وبدأ التقهقر.

(b)

**Figure 1.12** (a) Image used to test the system after preprocessing; (b) horizontal and vertical projections of the test document image.

document consisted of approximately 580 characters. Figure 1.12(a) shows an image that has been used to test the system. The original image was skewed by 18 degrees with respect to the horizontal. Figure 1.12(a) shows the resultant image when the image is passed through the preprocessing stage of the proposed system. The resultant image is inclined to zero degrees with respect to the horizontal. This shows that the proposed system is highly invariant to the rotation transformation. Figure 1.12(b) shows the horizontal and vertical projections of this test document image.

## 8. Conclusion

A method for offline recognition of Arabic text has been presented. Two approaches have been investigated: the syntactic approach and the neural network approach. In the preprocessing stage, we solved the problem of skew when scanning the documents. Even though the system developed is being tested for Naskh font only, it provides an ample scope for research towards the development of a system for multifont recognition. Extending the system to other most used fonts by extending the existing technique can be of great interest. We have shown the algorithms in detail for both recognition approaches and have suggested ways of improving the system. On the general improvement of the recognition rate using neural networks, an increase in the number of features extracted from seven to nine, or some higher number, may help in increasing the recognition rate. Other improvements on increasing the accuracy of the system include: computing the variance between the different momentum values obtained for a character; and trying to increase the variance among the momentum values. This will be important, especially for multifont recognition. The system is implemented in Java and is still in progress.

## Acknowledgment

The authors also acknowledge the support of King Fahd University of Petroleum and Minerals for funding this work under the Project No. EE/AUTOTEXT/232.

## References

[1] Amin, A. "Off-Line Arabic Character Recognition system: State of the Art," *Pattern Recognition*, **31**(5), pp. 517–530, 1998.

[2] Hu, M. K. "Visual Pattern Recognition by Moment Invariant," *IRE Transactions on Information Theory*, **IT – 8**, pp. 179–187, 1962.

[3] Amin A. and Al-Sadoun, H. B. "A new Segmentation Technique of Arabic Text," *11th IAPR*, The Hague **2**, pp. 441–445, 1992.

[4] Almuallim, H. and Yamaguchi, S. "A method of recognition of Arabic Cursive Handwriting," *IEEE Transaction Pattern Analysis and Machine Intelligence*, PAMI – 9, pp. 715–722, 1987.

[5] Cheung, A., Bennamoun, M. and Bergmann N. W. "An Arabic optical character recognition system using recognition-based segmentation," *Pattern Recognition*, **34**, pp. 215–233, 2001.

[6] Zidouri, A. "A Structural Description of Binary Document Images: Application for Arabic Character Recognition," *Proceedings of International Conference On Image Science, Systems, and Technology*, Las Vegas, **I**, pp. 458–464, June 25–28, 2001.

[7] Margner, V. "SARAT: A system for the Recognition of Arabic Printed Text," *Proceedings of the 11th International Conference on Pattern Recognition*, pp. 561–564, 1992.

[8] Hamami, H. and Berkani, D. "Recognition System for Printed Multi-Font and Multi-Size Arabic Characters," *Arabian Journal for Science and Engineering*, **27**(1B), pp. 57–72, 2002.

[9] Nazim Nawaz, S., Sarfraz, M., Zidouri, A. B. C. and Al-Khatib, W. "An Approach to Offline Arabic Character Recognition using Neural Networks," *10th International Conference on Electronics, Circuits and Systems, ICECS 2003*, Sharjah, UAE, 2003.

[10] Sarfraz, M., Nazim Nawaz, S. and Al-Khoraidly, A. "Offline Arabic Text Recognition System," *International Conference on Geometric Modelling and Graphics, GMAG 2003*, London, England, 2003.

[11] Zidouri, A. B. C., Sarfraz, M., Nazim Nawaz, S. and Ahmed, M. J. "PC Based Offline Character Recognition System," *Seventh International Symposium on Signal Processing and its Applications, ISSPA 2003*, Paris, France, 2003.

[12] Trier, O. D., Jain, A. K. and Taxt, T. "Feature Extraction methods for character recognition: A Survey," *Pattern Recognition*, **29**(4), pp. 641–662, 1996.

[13] Tubbs, J. D. "A Note on Binary Template Matching," *Pattern Recognition*, **22**(4), pp. 359–365, 1989.

[14] Mori, S. Suen, C. Y. and Yamamoto, K. "Historical review of OCR research and development," *Proceedings of the IEEE*, **80**, pp. 1029–1058, 1992.

[15] Abdul Wahab, O. A. "Application of Artificial Neural Networks to Optical Character Recognition," Thesis Dissertation, King Fahd University of Petroleum and Minerals, Dhahran, K. S. A, June 1994.

[16] Altuwaijri, M. and Bayoumi, M. "Arabic text recognition using Neural Networks," *Proceedings International Symposium on Circuits and Systems – ISCAS'94*, pp. 415–418, 1994.

[17] Boyce, J. F. and Hossack, W. J. "Moment Invariants for pattern recognition," *Pattern Recognition Letters*, **1**, pp. 451–456, 1983.

[18] El-Dabi, S. S., Ramsis, R. and Aladin Kamel, R. "Arabic Character Recognition System: A Statistical approach of recognizing cursive typewritten text," *Pattern Recognition*, **23**(5), pp. 485–495, 1990.

[19] Fakir, M. and Hassani, M. M. "Automatic Arabic Character recognition by moment invariants," *Colloque international de telecommunications*, Fes, Morocco, pp. 100–103, 1997.

[20] Fakir, M., Hassani, M. M. and Sodeyama, C. "Recognition of Arabic Characters using Karhunen–Loeve Transform and Dynamic Programming," *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, **6**, pp. 868–873, 1999.

[21] Sadallah, S. and Yacu, S. "Design of an Arabic Character Reading Machine," *Proceedings of Computer Processing of Arabic Language*, Kuwait, 1985.

[22] Jambi, K. "Arabic Character Recognition: Many approaches and one Decade," *Arabian Journal for Science and Engineering*, **16**, pp. 499–509, 1991.

[23] Chinveerphan, S. and Zidouri, A. B. C. "Modified MCR Expression using Binary Document Images," *IEICE Transactions Information & Systems*, **E78 –D**(4), pp. 503–507, 1995.

[24] Zidouri, A. B. C. *Arabic Document Image Analysis and Recognition Based on Minimum Covering Run*, Thesis Dissertation, Tokyo Institute of Technology, Japan, 1995.

[25] Zidouri, A. B. C., Chinveerphan, S. and Sato, M. "Classification of Document Image Blocks using Stroke Index," *IEICE Transactions Information & Systems*, **E78 –D**(3), pp. 290–503, 1995.

[26] Al–Sadoun, H. B. and Amin, A. "A new structural technique for recognizing printed Arabic Text," *International Journal on Pattern Recognition and Artificial Intelligence*., **9**, pp. 101–125, 1995.

[27] Smagt, P. P. V. "A Comparative Study of Neural Network Algorithms Applied to Optical Character Recognition," *IEA/AIE*, **2**, pp. 1037–1044, 1990.

[28] Duin, R. P. W. "Superlearning and Neural Network Magic," *Pattern Recognition Letters*, **15**, pp. 215–217, 1994.

[29] Jain, A. K. and Mao, J. "Neural Networks and pattern recognition," In *Proceedings of the IEEE World congress on Computational Intelligence*, Orlando, Florida, 1994.

[30] Schalkoff, R. J. *Pattern Recognition: Statistical, Structural and Neural Approaches*, John Wiley and Sons Inclusive.

[31] Haykin, S. *Neural Networks: A Comprehensive Foundation*, second edition, Prentice Hall, 1999.

# 2

# License Plate Recognition System: Saudi Arabian Case

**Muhammad Sarfraz**
Department of Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

**Mohammed Jameel Ahmed**
Information Technology Department, College of Business Administration, Jeddah 21361, Kingdom of Saudi Arabia

*License Plate Recognition (LPR) is one kind of intelligent transport system and is of considerable interest because of its potential applications to areas such as highway electronic toll collection, traffic monitoring systems and so on. It was developed to identify vehicles by the contents of their license plate. Research is in progress for the recognition of Korean, Chinese, European and other license plates. However, the proposed work seems to be the first attempt towards the recognition of Saudi Arabian license plates.*

*This chapter proposes an automatic license plate recognition system for Saudi Arabian license plates. The system captures images of the vehicles with a digital camera. An algorithm for the extraction of the license plate has been designed and an algorithm for segmentation of characters is proposed. The performance of the system has been investigated on real images of about 610 vehicles captured under various illumination conditions. Recognition of about 95% shows that the system is quite effective.*

## 1. Introduction

Vehicle License Plate Recognition (LPR) is one form of automatic vehicle identification system. Real-time LPR plays a major role in automatic monitoring of traffic rules and maintaining law enforcement on public roads. This area is challenging because it requires an integration of many computer vision problem solvers, which include object detection and character recognition [1]. The automatic identification of vehicles by the contents of their license plates is important in private transport applications. There are many applications of such recognition systems, some of them are: border crossing control;

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

identification of stolen vehicles; automatic parking attendants; petrol station forecourt surveillance; speed limit enforcement; security; and customer identification enabling personalized services [2].

There are multiple commercial license plate recognition systems available in the current literature [3]. Applications such as SeeCar, Perceptics, and Pearpoint are commercially available and some of the research projects include Esprit 5184 Locomotive [4], software architecture using DLLs [2]. The steps involved in recognition of a license plate are image acquisition, license plate extraction, segmentation and recognition. Image acquisition is the first step in an LPR system. The current literature discusses different image acquisition methods. Naito *et al.* [5] developed a sensing system that uses two CCDs (Charge Coupled Devices) and a prism to split an incident ray into two lights with different intensities. Salgado *et al.* [4] used a sensor subsystem involving a high-resolution CCD camera supplemented by a number of new digital operation capabilities. The proposed system uses a high-resolution digital camera for image acquisition.

License plate extraction is the key step in an LPR system, it influences the accuracy of the system significantly. Different approaches for the extraction of the license plate depending upon the background color of the image are presented in [6]. Hontani *et al.* [7] proposed a method for extracting characters without prior knowledge of their position and size in the image. Kim *et al.* [1] used two neural network-based filters and a postprocessor to combine two filtered images in order to locate the license plates. Park *et al.* [8] devised a method to extract Korean license plates depending on the color of the plate. The proposed approach uses matching of vertical edges. This approach involves three steps, vertical edge detection, filtering and vertical edge matching.

In the segmentation phase, individual characters are isolated from the license plate. Various approaches have been proposed in the literature. Nieuwoudt *et al.* [9] used region growing for segmentation of characters. Morel *et al.* [10] used a Partial Differential Equation (PDE) based technique for image segmentation. The proposed approach for segmentation is based on vertical projections on the extracted license plate and then counting the number of black pixels in each column.

Recognition of characters is the last phase in the LPR system. A wide variety of approaches have been considered for individual character recognition. Cowell *et al.* [11] discussed the recognition of individual Arabic and Latin characters. Their approach identifies the characters based on the number of black pixel rows and columns of the character, and comparison of those values to a set of templates or signatures in the database. Cowell *et al.* [12] used thinning of Arabic characters to extract essential structural information, which may be later used for the classification stage. Mei Yu *et al.* [6] and Naito *et al.* [5] used template matching. In the proposed system, recognition is done using template matching based on the Hamming distance between the characters.

The chapter is organized as follows: Section 2 provides an overview of the overall system, listing different stages in the LPR system. The image acquisition phase is explained in Section 3. Section 4 discusses the proposed method for license plate extraction. Section 5 gives a brief description of the segmentation of individual characters using vertical projection with pixel count. Section 6 deals with recognition of characters based on the Hamming distance of each character. Section 7 discusses experimental analysis and the chapter is concluded in Section 8.

## 2. Structure of a Typical LPR System

The system presented is designed to recognize license plates from the front and rear of the vehicle. Input to the system is an image containing the license plate acquired by a digital camera, and its output is the characters on the license plate. The system consists of four modules: image acquisition, license plate extraction, segmentation and recognition of individual characters. The structure of the system is shown in Figure 2.1.

The first task is to acquire the image of a vehicle containing the license plate using a digital camera. The second task then extracts the region that contains the license plate. The third task isolates the six characters (consisting of three letters and three numerals) and the last task identifies the individual characters.

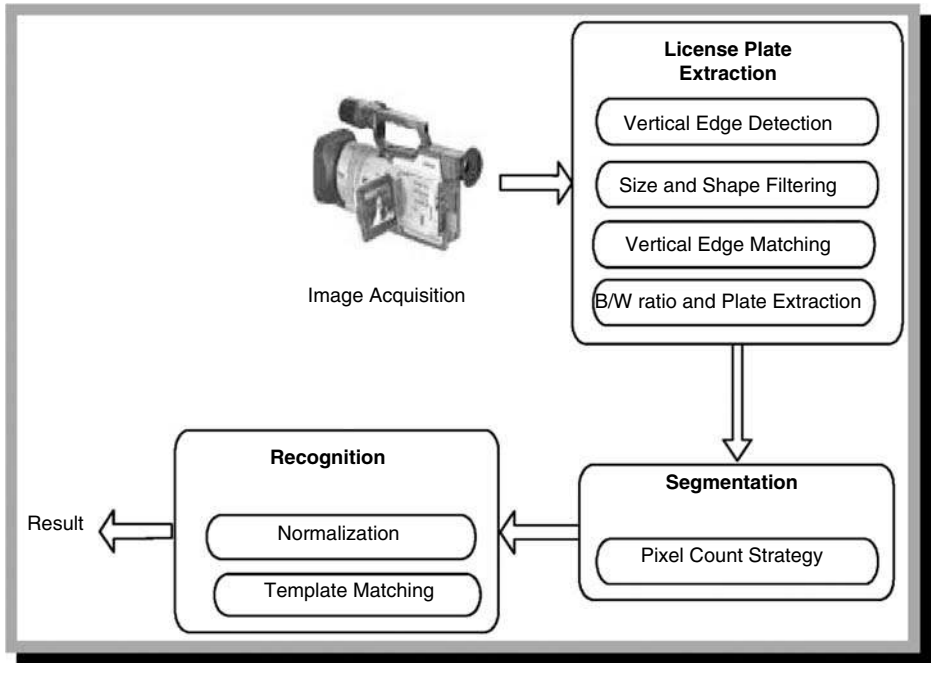



**Figure 2.1** The structure of a license plate recognition system.

## 3. Image Acquisition

This is the first phase in the LPR system. There are basically three ways of acquiring an image [11]. They are as follows:

- using a conventional analog camera and a scanner;
- using a digital camera;
- using a video camera and a frame grabber (capture card) to select a frame.

The first method, using a conventional analog camera, is clearly not appropriate for the LPR system since it is time consuming, tedious and impractical. The second method, i.e. using a digital camera, is more practical, cost effective and reliable. The third one uses a video camera with a frame grabber, which could be used in a real-life system to make the system automated, and is suitable for real-time processing.

In the proposed system, a high-resolution digital camera is used to acquire the image. This is shown in Figure 2.2(a). The acquired image is first converted to a grayscale image. Conversion to a grayscale facilitates the extraction of the license plate. The grayscale image is shown in Figure 2.2(b).

## 4. License Plate Extraction

License plate extraction is the key step in an LPR system, it influences the accuracy of the system significantly. The goal of this phase is, given an input image, to produce a number of candidate regions with high probability of containing a license plate. In the adopted approach, extraction of the license plate is divided into three steps which are explained in the following subsections.

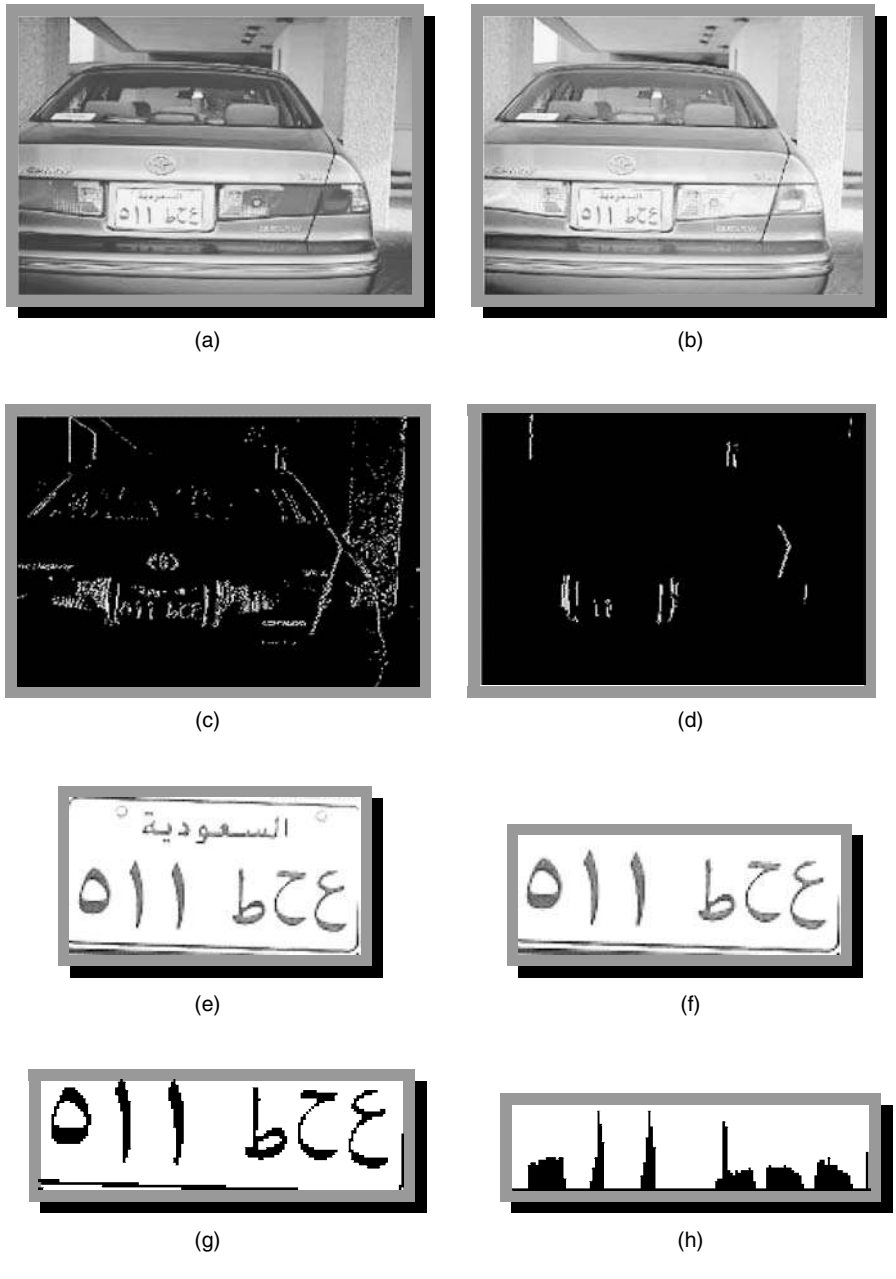



**Figure 2.2** Results showing different stages in a license plate recognition system (a) original image; (b) grayscale image; (c) vertical edges; (d) vertical edge regions; (e) extracted license plate; (f) cropped license plate; (g) binary image; (h) histogram depicting the vertical projections on (g).

## 4.1 Vertical Edge Detection

For the transformed grayscale image, its corresponding vertical edges are detected using Sobel or Prewitt edge detectors. The proposed system uses the Sobel edge detector because it shows better results. The threshold used by the edge detector is dynamic because the system takes an automatic value from the algorithm. The Sobel edge detector uses a $3 \times 3$ mask, which is applied on the input image to give the resultant edged image. The edge detection algorithm is not time consuming, since the algorithm used is a built-in feature of MATLAB6.1 [13]. Figure 2.2(c) shows an image with vertical edges. It is observed that most of the vehicles usually have more horizontal lines than vertical lines [6]. To reduce the complexity of the algorithm, the vertical edges are detected. If two of the vertical edges are detected correctly, the four corners of the license plate can then be located. This helps in extracting the license plate exactly from the input image, even if it is out of shape.

## 4.2 Filtering

Filtering is basically used to remove objects that do not satisfy some specific features. In the proposed approach, the seed-filling algorithm by Smith [14] is used to filter the unwanted objects. After filtering, the result is shown in Figure 2.2(d). The process of filtering starts by first detecting all the regions. Any group of white pixels is called a region if they are eight-connected pixels. In Figure 2.3, any of the eight pixels around the shaded pixel are called eight-connected pixels of the shaded pixel.

Once all the pixels of a region are marked, the region is checked for its size and shape. Through filtering, our target is to select the regions that can serve as possible license plate boundaries and discard the others by filling black colors in their place.

To achieve this target, any region of interest must be a straight line of specific length. So, for each region, the size and shape filter is applied. The first pixel of the region is marked as $P_0$ and the last is marked as $P_{\text{end}}$.

An overall filtering algorithm is presented as follows:

```
For each row:
1. Traverse from first column to last column (West to East).
2. If a white pixel is encountered and is not marked as part of any other
   region:
   (a) Select the pixel as P0.
   (b) Select the white pixels on East, SW, South and SE recursively until
       a white pixel having no eight-connected white pixel is found.
   (c) Mark the last white pixel as Pend.
   (d) if the slope and distance of line P0, Pend falls in the desired range:
       (i) mark all the pixels in the region as they are a part of a selected
           region.
       else:
       (i) fill all the pixels in the current region with black color,
           i.e., discard them.
4. End for.
```

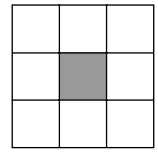


**Figure 2.3** Eight-connected components.

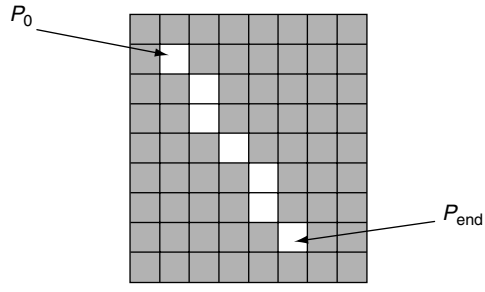



**Figure 2.4** Example of an eight-connected region.

The shape of the region is detected through the slope of the line joining $P_0$ and $P_{end}$. The size of the region is the distance between $P_0$ and $P_{end}$. If the distance falls within the thresholded value, it is taken as the region of interest. A threshold of 50 to 100 pixels is taken as the distance between $P_0$ and $P_{end}$ with an angle of deviation of around $\pm 15°$. The region in Figure 2.4 is discarded on the basis of the slope of the line $P_0$, $P_{end}$.

## *4.3 Vertical Edge Matching*

In this phase, the width to height ratio of the license plate is used to match the vertical edges for finding the region of where there is a probability of a license plate. The ratio of width to height of Saudi Arabian license plates is about 2:1. After the filtering step, the image in Figure 2.2(d) is extracted, having several vertical regions. Only two of the regions could be the possible boundary of the license plate. The task of vertical edge matching is to find out the correct pair of regions that include the license plate. To achieve this task, the width to height ratio of the rectangular area between two vertical regions is compared with the actual standard ratio of a license plate. The standard ratio is taken between 1.75 : 1 to 2.25 : 1. Figure 2.2(e) shows the extracted license plate for the matched vertical edges.

An algorithm for vertical edge matching is as follows:

```
1. For i = 1 to no. of extracted regions
2. For j = i + 1 to no. of extracted regions
   (a) if width to height ratio of region (i) and region (j) falls in the
       range 1.75:1 to 2.25:1
       (i) select the two regions in the list of possible license plate
           regions.
3. End for.
4. End for.
```

Figure 2.5 explains how the vertical edge matching algorithm computes the horizontal and vertical distances between two regions. In the figure, Region 1 and Region 2 are processed for the possibility that they form two vertical edges of a license plate. The width to height ratio of Region 1 and Region 2 certainly does not match with the specified ratio (i.e. between 1.75:1 to 2.25:1), so this pair cannot be the region of interest. In Figure 2.5, Region 2 and Region 3 are the possible pair of vertical edges of the license plate.

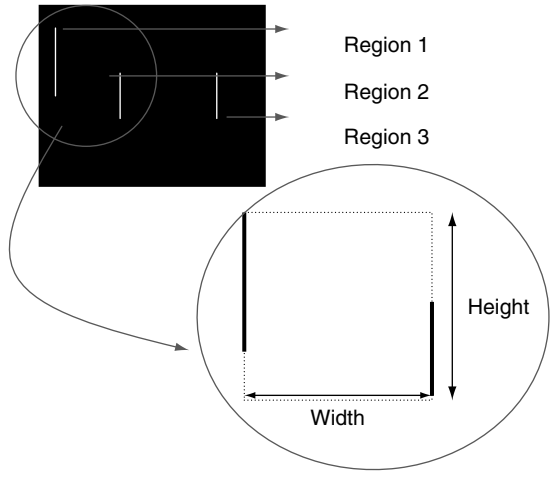



**Figure 2.5** Computation of horizontal and vertical distances between two extracted regions.

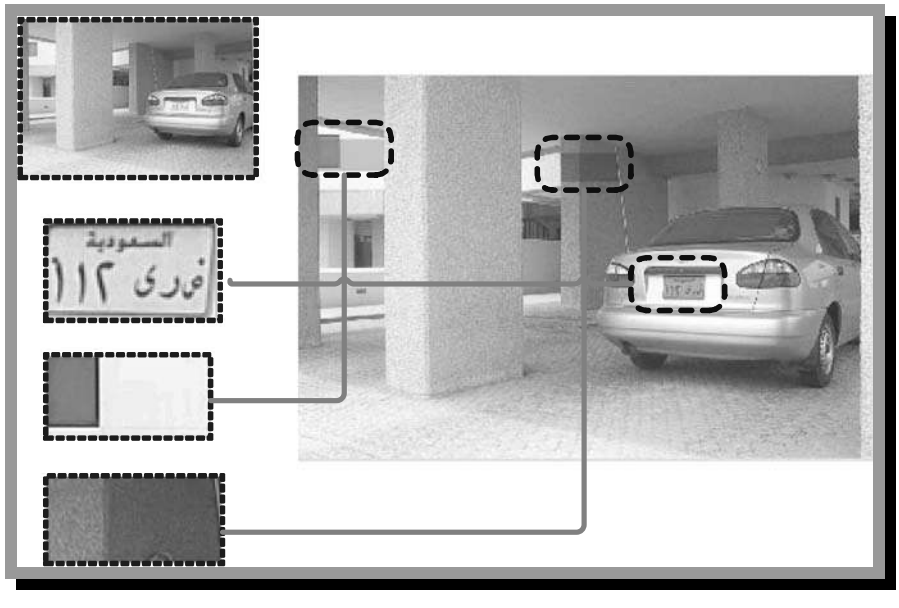


**Figure 2.6** Three extracted license plate regions of a vehicle satisfying the width to height constraint.

In some cases, there is a possibility of more than one pair of regions that satisfy the above threshold. This case is shown in Figure 2.6, where there are three regions satisfying the constraint of width to height ratio. To overcome this problem, the black to white ratio of these regions is taken, to get the probable license plate.

### *4.4 Black to White Ratio and Plate Extraction*

This phase is considered when more than one pair of probable license plate regions are obtained after the matching of vertical regions for their width to height ratio, as shown in Figure 2.6. All the coordinate points for every pair of matched regions are stored and the black to white ratios of the stored regions are calculated with respect to Figure 2.2(c). Since the characters on the license plate in Figure 2.2(c) contain white pixels, so the B/W ratio for the probable plate region is much less than the ratio of any of the extracted regions which do not contain a license plate. Therefore, if the pair is a possible license plate, then the ratio is within a specified threshold. The threshold is selected based on tests performed on a number of plates.

## 5. License Plate Segmentation

After the extraction phase, the license plate is segmented into individual characters. To ease the process of identifying the characters, it is preferable to divide the extracted plate into six images. This is done because the Saudi Arabian license plates consists of six characters, with three letters and three numerals.

For the segmentation phase, the proposed strategy used is based on vertical projections and counting the number of pixels in each column. In the proposed strategy, first the upper part (about 30 %) of the extracted license plate is cropped to remove السعودية. This is shown in Figure 2.2(f). Presence of bolts and logos in the license plates may prove cumbersome during the segmentation process. The Saudi Arabian license plate template does not contain any logos. Since the bolts are usually on the upper part of the plate, they are removed in the process of cropping. The image is then converted into binary form (as shown in Figure 2.2(g)) and vertical projections are made. Then, the number of black pixels in each column is counted and a histogram is plotted, as shown in Figure 2.2(h). The characters are segmented depending on the transition from a crest to its corresponding trough. In this strategy, some threshold is taken to avoid unnecessary segmentation by some unwanted black pixels.

## 6. Character Recognition

This is the last phase in the LPR system. This phase is divided into two stages:

1. Normalization of individual characters.
2. Recognition using template matching.

### *6.1 Normalization*

In this phase, first the extracted characters are refined to fit the characters into a window without white spaces on all four sides. Figure 2.7(a) shows the template for the extracted character 'ش' with respect to the above refinement. Now, each character image is normalized to a size of $40 \times 40$. Figure 2.7(b) shows the character 'ش' normalized to $40 \times 40$. The characters are normalized using a window to view port transformation. This mapping is used to map every pixel of the original image to the corresponding pixel in the normalized image. The image obtained is in variant to scaling because the characters were refined (i.e. all the white spaces on the top, bottom, left and right were removed, so as to fit the character exactly into the window).

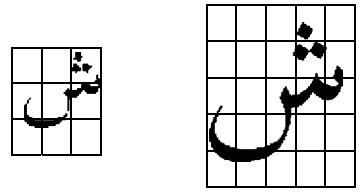



**Figure 2.7** (a) Original template for character, 'ش'; (b) normalized $40 \times 40$ template for character 'ش'.

## *6.2 Template Matching*

Template matching for character recognition is straightforward and is reliable. This method is more tolerant to noise than the structural analysis method. In this approach, the templates are normalized to $40 \times 40$ pixels and stored in the database. The extracted character, after normalization, is matched with all the characters in the database using the Hamming distance approach. This approach is shown in Equation (2.1).

$$\sum_{i=1}^{nrows} \sum_{j=1}^{ncols} \text{mismatch}_{i,j} \tag{2.1}$$

where

$$\text{mismatch}_{i,j} = \begin{cases} 1, \text{if original}_{i,j} \neq \text{extracted}_{i,j} \\ 0, \text{if original}_{i,j} = \text{extracted}_{i,j} \end{cases}$$

where *nrows* and *ncols* are the number of rows and columns in the original and extracted images. In our case, $nrows = ncols = 40$, as the image is normalized to $40 \times 40$. The mismatches for each of the extracted characters are found by comparison with the original characters in the database, and the character with the least mismatch value is taken as the recognized character.

In the case of Saudi Arabian license plates, containing three letters and three numerals, recognition is done by matching each extracted character to all the 38 (28 letters and ten numerals) characters in the database.

# 7. Experimental Analysis and Results

Experiments have been performed to test the above system. The system is designed in MATLAB6.1 [13] for recognition of Saudi Arabian license plates. The image to the system is a grayscale of size $640 \times 480$. The test images were taken under various illumination conditions (i.e. noon and in the evening).

The experiments for the above system were performed under different situations, i.e. the experiments were carried out for the following cases:

- license plates in normal shapes (as shown in Figure 2.8);
- license plates that are leaning with some angle of view (as shown in Figure 2.9);

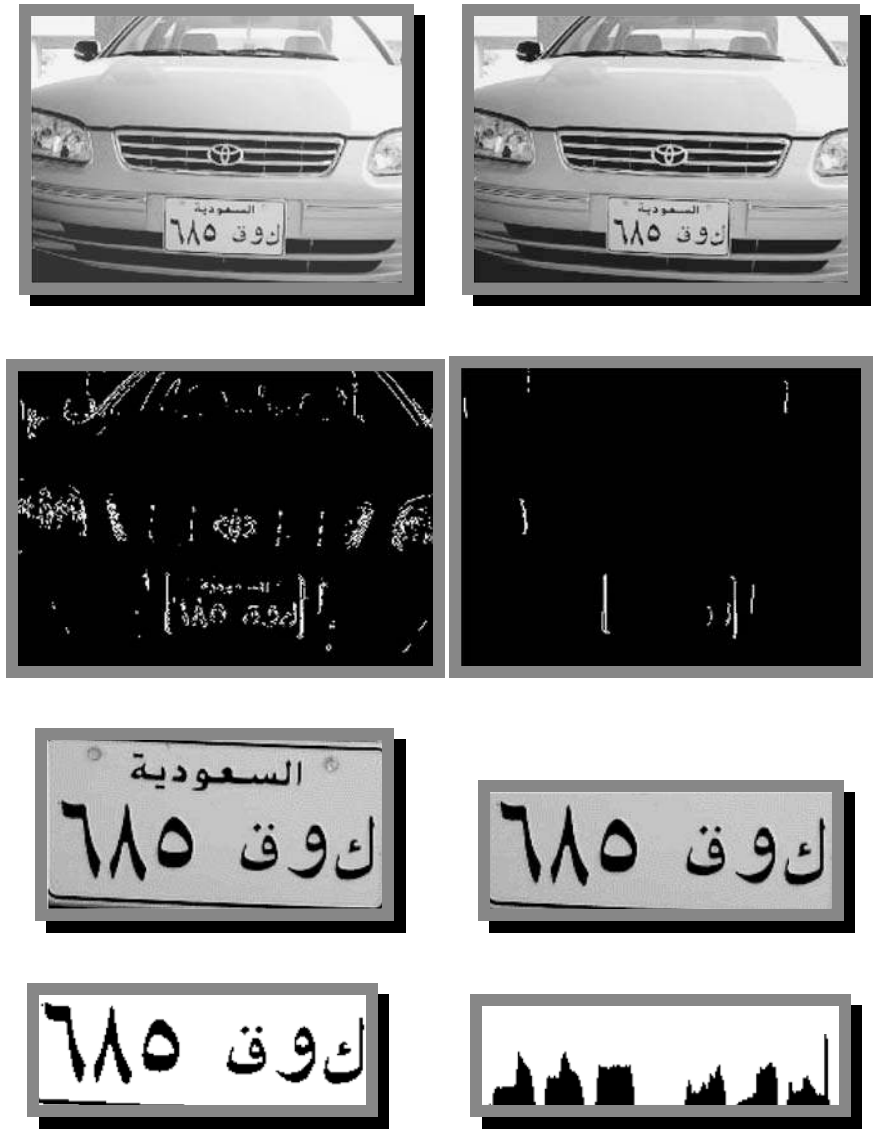



**Figure 2.8** The overall process of an LPR system with the front part of the car in view and the license plate in normal shape.

- license plates that have a similar color to the car (as shown in Figure 2.10);
- dirty license plates (as shown in Figure 2.11).

The whole system is working fine, achieving a high recognition rate. Table 2.1 gives results for the percentage recognition of license plate extraction, segmentation and character recognition. A failure results if a character of the license plate is incorrectly interpreted. This normally happens due to the bad

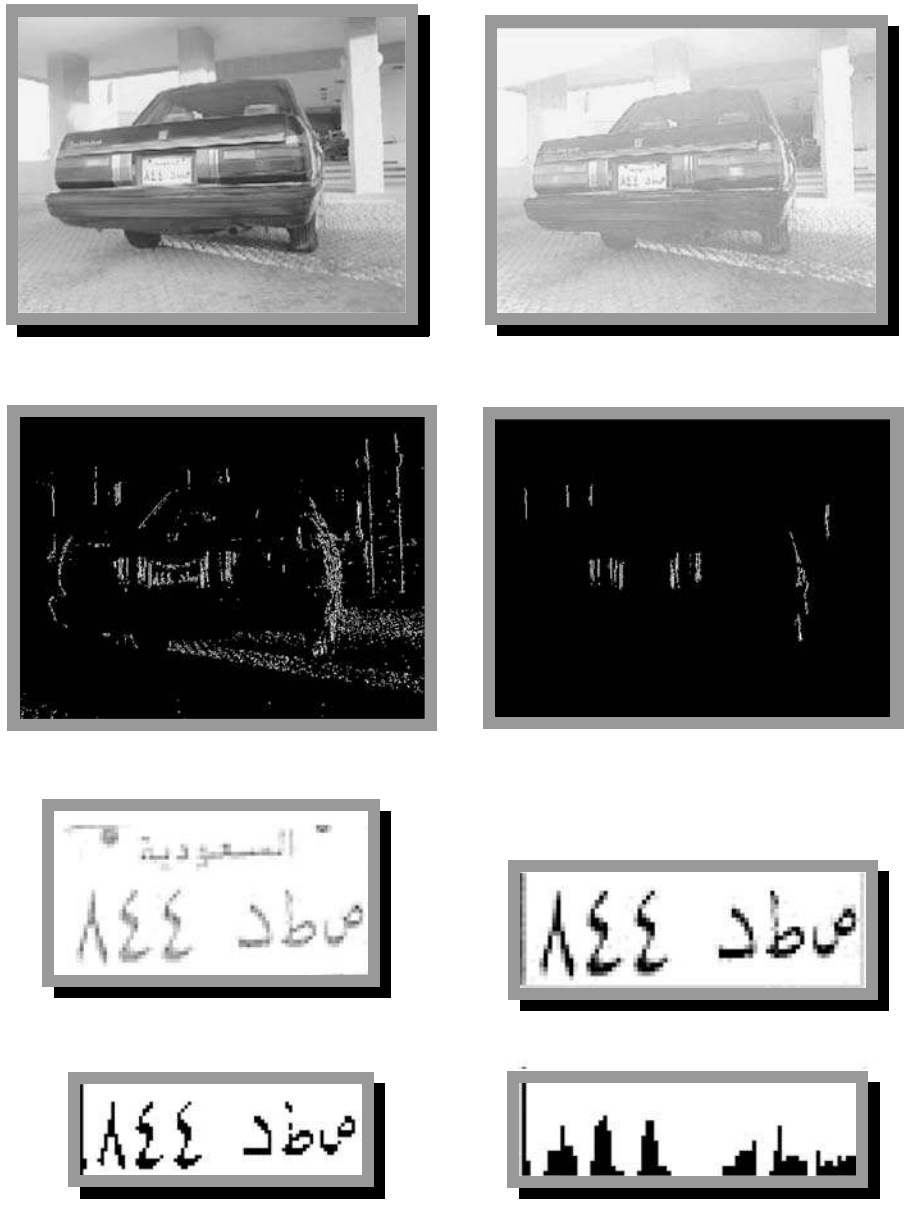



**Figure 2.9** The overall process of an LPR system with the rear part of the car in view and slanted towards the right, taken before sunset.

quality of the extracted plate. It is shown that the system has correctly recognized 581 license plates out of 610 test images. The system was implemented on PIII (700 MHz) using MATLAB6.1 [13].

The experimental results show that the shortcoming of the proposed system is mainly due to bad quality of the input image during the acquisition stage (i.e. the bad quality is due to the presence

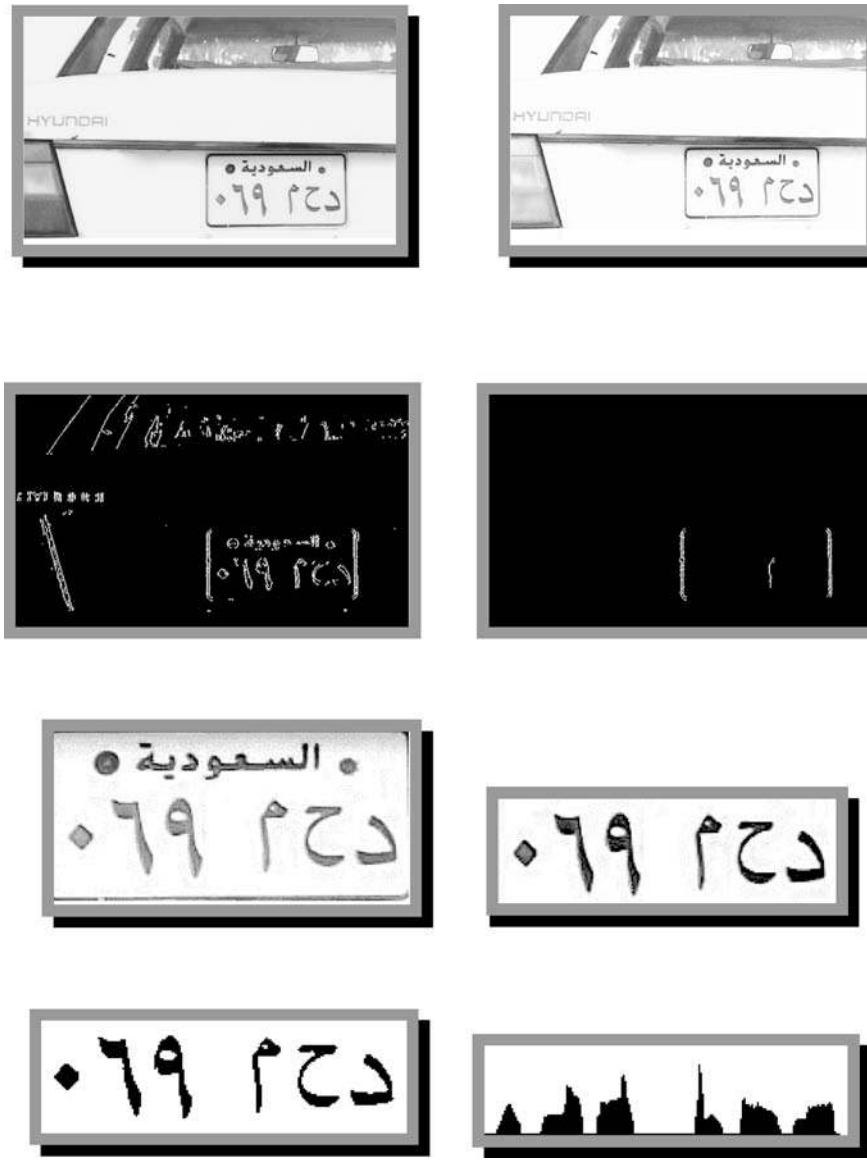



**Figure 2.10** The overall process of an LPR system showing a car having a similar color to the license plate body.

of dirty or unclear license plates), or unclear detection or extraction of the edges. The segmentation phase gives good results because the image is converted into binary form. In character recognition there is some form of misrecognition. This is due to the nuts and bolts appearing within the character. Otherwise, there is a standard font for all the Arabic characters used in the license plates, which does not cause any problem during the recognition phase.

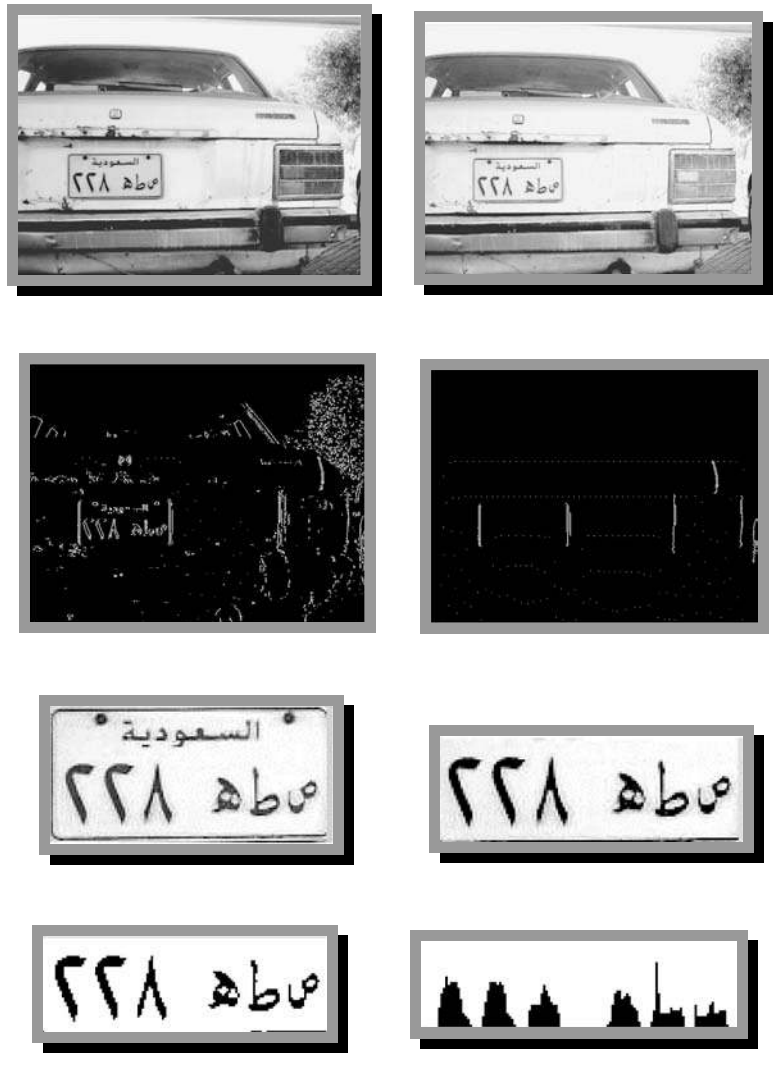



**Figure 2.11** The overall process of an LPR system showing a car and license plate with dust and scratches.

**Table 2.1** Recognition rate for license plate extraction, license plate segmentation and license plate recognition.

| | License plate extraction | License plate segmentation | License plate recognition |
|---|---|---|---|
| **Correct recognition** | 587/610 | 574/610 | 581/610 |
| **Percentage recognition** | 96.22 % | 94.04 % | 95.24 % |

## 8. Conclusion

Although there are many running systems for recognition of various plates, such as Singaporean, Korean and some European license plates, the proposed effort is the first of its kind for Saudi Arabian license plates. The license plate recognition involves image acquisition, license plate extraction, segmentation and recognition phases. Besides the use of the Arabic language, Saudi Arabian license plates have several unique features that are taken care of in the segmentation and recognition phases. The system has been tested over a large number of car images and has been proven to be 95 % accurate.

## References

[1] Kim, K. K., Kim, K. I., Kim, J. B. and Kim, H. J. "Learning based approach for license plate recognition," *Proceedings of IEEE Processing Society Workshop on Neural Networks for Signal Processing*, **2**, pp. 614–623, 2000.

[2] Bailey, D. G., Irecki, D., Lim, B. K. and Yang, L. "Test bed for number plate recognition applications," *Proceedings of the First IEEE International Workshop on Electronic Design, Test and Applications (DELTA'02)*, IEEE Computer Society, 2002.

[3] Hofman, Y. *License Plate Recognition – A Tutorial*, Hi-Tech Solutions, http://www.licenseplaterecognition.com/#whatis, 2004.

[4] Salgado, L., Menendez, J. M., Rendon, E. and Garcia, N. "Automatic car plate detection and recognition through intelligent vision engineering," *Proceedings of IEEE 33rd Annual International Carnahan Conference on Security Technology*, pp. 71–76, 1999.

[5] Naito, T., Tsukada, T., Yamada, K., Kozuka, K. and Yamamoto, S. "Robust license-plate recognition method for passing vehicles under outside environment." *IEEE Transactions on Vehicular Technology*, **49**(6), pp. 2309–2319, 2000.

[6] Yu, M. and Kim, Y. D. "An approach to Korean license plate recognition based on vertical edge matching," *IEEE International Conference on Systems, Man, and Cybernetics*, **4**, pp. 2975–2980, 2000.

[7] Hontani, H. and Koga, T. "Character extraction method without prior knowledge on size and information," *Proceedings of the IEEE International Vehicle Electronics Conference (IVEC'01)*, pp. 67–72, 2001.

[8] Park, S. H., Kim, K. I., Jung, K. and Kim, H. J. "Locating car license plates using neural networks," *IEE Electronics Letters*, **35**(17), pp. 1475–1477, 1999.

[9] Nieuwoudt, C. and van Heerden, R. "Automatic number plate segmentation and recognition," in *Seventh Annual South African workshop on Pattern Recognition*, pp. 88–93, IAPR, 1996.

[10] Morel, J. and Solemini, S. *Variational Methods in Image Segmentation*, Birkhauser, Boston, 1995.

[11] Cowell, J. and Hussain, F. "A fast recognition system for isolated Arabic characters," *Proceedings of the Sixth International Conference on Information and Visualisation*, IEEE Computer Society, London, England, pp. 650–654, 2002.

[12] Cowell, J. and Hussain, F. "Extracting features from Arabic characters," *Proceedings of the IASTED International Conference on Computer Graphics and Imaging*, Honolulu, Hawaii, USA, pp. 201–206, 2001.

[13] The MathWorks, Inc., *The matlab package*, http://www.mathworks.com/,1993–2003.

[14] Smith A. R. "Tint Fill," *Computer Graphics*, **13**(2), pp. 276–283, 1979.

# 3

# Algorithms for Extracting Textual Characters in Color Video

**Edward K. Wong**
**Minya Chen**
Department of Computer and Information Science, Polytechnic University, 5 Metrotech Center, Brooklyn, NY 11201, USA

*In this chapter, we present a new robust algorithm for extracting text in digitized color video. The algorithm first computes the maximum gradient difference to detect potential text line segments from horizontal scan lines of the video. Potential text line segments are then expanded or combined with potential text line segments from adjacent scan lines to form text blocks, which are then subject to filtering and refinement. Color information is then used to more precisely locate text pixels within the detected text blocks. The robustness of the algorithm is demonstrated by using a variety of color images digitized from broadcast television for testing. The algorithm performs well on JPEG images and on images corrupted with different types of noise. For video scenes with complex and highly textured backgrounds, we developed a technique to reduce false detections by utilizing multiframe edge information, thus increasing the precision of the algorithm.*

## 1. Introduction

With the rapid advances in digital technology, more and more databases are multimedia in nature, containing images and video in addition to the textual information. Many video databases today are manually indexed, based on textual annotations. The manual annotation process is often tedious and time consuming. It is therefore desirable to develop effective computer algorithms for automatic annotation and indexing of digital video. Using a computerized approach, indexing and retrieval are performed based on features extracted directly from the video, which directly capture or reflect the content of the video.

Currently, most automatic video systems extract global low-level features, such as color histograms, edge information, textures, etc., for annotations and indexing. There have also been some advances in using region information for annotations and indexing. Extraction of high-level generic objects from video for annotations and indexing purposes remains a challenging problem to researchers in the field,

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

and there has been limited success on using this approach. The difficulty lies in the fact that generic 3D objects appear in many different sizes, forms and colors in the video. Extraction of text as a special class of high-level object for video applications is a promising solution, because most text in video has certain common characteristics that make the development of robust algorithms possible. These common characteristics include: high contrast with the background, uniform color and intensity, horizontal alignment and stationary position in a sequence of consecutive video frames. Although there are exceptions, e.g. moving text and text embedded in video scenes, the vast majority of text possesses the above characteristics.

Text is an attractive feature for video annotations and indexing because it provides rich semantic information about the video. In broadcast television, text is often used to convey important information to the viewer. In sports, game scores and players' names are displayed from time to time on the screen. In news broadcasts, the location and characters of a news event are sometimes displayed. In weather broadcasts, temperatures of different cities and temperatures for a five-day forecast are displayed. In TV commercials, the product names, the companies selling the products, ordering information, etc. are often displayed. In addition to annotation and indexing, text is also useful for developing computerized methods for video skimming, browsing, summarization, abstraction and other video analysis tasks.

In this chapter, we describe the development and implementation of a new robust algorithm for extracting text in digitized color video. The algorithm detects potential text line segments from horizontal scan lines, which are then expanded and merged with potential text line segments from adjacent scan lines to form text blocks. The algorithm was designed for texts that are superimposed on the video, and with the characteristics described above. The algorithm is effective for text lines of all font sizes and styles, as long as they are not excessively small or large relative to the image frame. The implemented algorithm has fast execution time and is effective in detecting text in difficult cases, such as scenes with highly textured backgrounds, and scenes with small text. A unique characteristic of our algorithm is the use of a scan line approach, which allows fast filtering of scan line video data that does not contain text. In Section 2, we present some prior and related work. Section 3 describes the new text extraction algorithm. Section 4 describes experimental results. Section 5 describes a method to improve the precision of the algorithm in video scenes with complex and highly textured backgrounds by utilizing multiframe edge information. Lastly, Section 6 contains discussions and gives concluding remarks.

## 2. Prior and Related Work

Most of the earlier work on text detection has been on scanned images of documents or engineering drawings. These images are typically binary or can easily be converted to binary images using simple binarization techniques such as grayscale thresholding. Example works are [1–6]. In [1], text strings are separated from non-text graphics using connected component analysis and the Hough Transform. In [2], blocks containing text are identified based on a modified Docstrum plot. In [3], areas of text lines are extracted using a constrained run-length algorithm, and then classified based on texture features computed from the image. In [4], macro blocks of text are identified using connected component analysis. In [5], regions containing text are identified based on features extracted using two-dimensional Gabor filters. In [6], blocks of text are identified based on using smeared run-length codes and connected component analysis.

Not all of the text detection techniques developed for binary document images could be directly applied to color or video images. The main difficulty is that color and video images are rich in color content and have textured color backgrounds. Moreover, video images have low spatial resolution and may contain noise that makes processing difficult. More robust text extraction methods for color and video images, which contain small and large font text in complex color backgrounds, need to be developed.

In recent years, we have seen growing interest by researchers on detecting text in color and video images, due to increased interest in multimedia technology. In [7], a method based on multivalued

image decomposition and processing was presented. For full color images, color reduction using bit dropping and color clustering was used in generating the multivalued image. Connected component analysis (based on the block adjacency graph) is then used to find text lines in the multivalued image. In [8], scene images are segmented into regions by adaptive thresholding, and then observing the gray-level differences between adjacent regions. In [9], foreground images containing text are obtained from a color image by using a multiscale bicolor algorithm. In [10], color clustering and connected component analysis techniques were used to detect text in WWW images. In [11], an enhancement was made to the color-clustering algorithm in [10] by measuring similarity based on both RGB color and spatial proximity of pixels. In [12], a connected component method and a spatial variance method were developed to locate text on color images of CD covers and book covers. In [13], text is extracted from TV images based on using the two characteristics of text: uniform color and brightness, and 'clear edges.' This approach, however, may perform poorly when the video background is highly textured and contains many edges. In [14], text is extracted from video by first performing color clustering around color peaks in the histogram space, and then followed by text line detection using heuristics. In [15], coefficients computed from linear transforms (e.g. DCT) are used to find $8 \times 8$ blocks containing text. In [16], a hybrid wavelet/neural network segmenter is used to classify regions containing text. In [17], a generalized region labeling technique is used to find homogeneous regions for text detection. In [18], text is extracted by detecting edges, and by using limiting constraints in the width, height and area of the detected edges. In [19], caption texts for news video are found by searching for rectangular regions that contain elements with sharp borders in a sequence of frames. In [20], the directional and overall edge strength is first computed from the multiresolution representation of an image. A neural network is then applied at each resolution (scale) to generate a set of response images, which are then integrated to form a salience map for localizing text. In [21], text regions are first identified from an image by texture segmentation. Then a set of heuristics is used to find text strings within or near the segmented regions by using spatial cohesion of edges. In [22], a method was presented to extract text directly from JPEG images or MPEG video with a limited amount of decoding. Texture characteristics computed from DCT coefficients are used to identify $8 \times 8$ DCT blocks that contain text.

Text detection algorithms produce one of two types of output: rectangular boxes or regions that contain the text characters; or binary maps that explicitly contain text pixels. In the former, the rectangular boxes or regions contain both background and foreground (text) pixels. The output is useful for highlighting purposes but cannot be directly processed by Optical Character Recognition (OCR) software. In the latter, foreground text pixels can be grouped into connected components that can be directly processed by OCR software. Our algorithm is capable of producing both types of output.

## 3. Our New Text Extraction Algorithm

The main idea behind our algorithm is to first identify potential text line segments from individual horizontal scan lines based on the maximum gradient difference (to be explained below). Potential text line segments are then expanded or merged with potential text line segments from adjacent scan lines to form text blocks. False text blocks are filtered based on their geometric properties. The boundaries of the text blocks are then adjusted so that text pixels lying outside the initial text region are included. Color information is then used to more precisely locate text pixels within text blocks. This is achieved by using a bicolor clustering process within each text block. Next, non-text artifacts within text blocks are filtered based on their geometric properties. Finally, the contours of the detected text are smoothed using a pruning algorithm.

In our algorithm, the grayscale luminance values are first computed from the RGB or other color representations of the video. The algorithm consists of seven steps.

1. Identify potential text line segments.
2. Text block detection.
3. Text block filtering.

4. Boundary adjustments.
5. Bicolor clustering.
6. Artifact filtering.
7. Contour smoothing.

Steps 1–4 of our algorithm operate in the grayscale domain. Step 5 operates in the original color domain, but only within the spatial regions defined by the detected text blocks. Steps 6 and 7 operate on the binary maps within the detected text blocks. After Step 4, a bounding box for each text string in the image is generated. The output after Step 7 consists of connected components of binary text pixels, which can be directly processed by OCR software for recognition. Below is a high-level description of each step of the algorithm.

## *3.1 Step 1: Identify Potential Text Line Segments*

In the first step, each horizontal scan line of the image (Figure 3.1 for example) is processed to identify potential text line segments. A text line segment is a continuous one-pixel thick segment on a scan line that contains text pixels. Typically, a text line segment cuts across a character string and contains interleaving groups of text pixels and background pixels (see Figure 3.2 for an illustration.) The end points of a text line segment should be just outside the first and last characters of the character string.

In detecting scan line segments, the horizontal luminance gradient d$x$ is first computed for the scan line by using the mask $[-1, 1]$. Then, at each pixel location, the *Maximum Gradient Difference (MGD)* is computed as the difference between the maximum and minimum gradient values within a local window of size $n \times 1$, centered at the pixel. The parameter $n$ is dependent on the maximum text size we want to detect. A good choice for $n$ is a value that is slightly larger than the stroke width of the largest character we want to detect. The chosen value for $n$ would be good for smaller-sized characters as well. In our experiments, we chose $n = 21$. Typically, text regions have large MGD values and background regions have small MGD values. High positive and negative gradient values in text regions result from high-intensity contrast between the text and background regions. In the case of bright text on a dark background, positive gradients are due to transitions from background pixels to text pixels, and negative gradients are due to transitions from text pixels to background pixels. The reverse is true for dark intensity text on a bright background. Text regions have both large positive and negative gradients in a local region due to the even distribution of character strokes. This results in locally large MGD values. Figure 3.3 shows an example gradient profile computed from scan line number 80 of the


**Figure 3.1** Test image 'data13'.

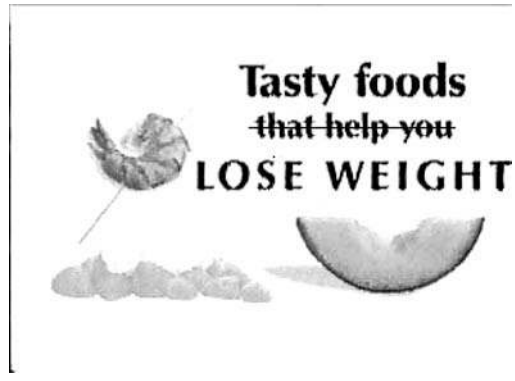



**Figure 3.2** Illustration of a 'scan line segment' (at $y = 80$ for test image 'data13').

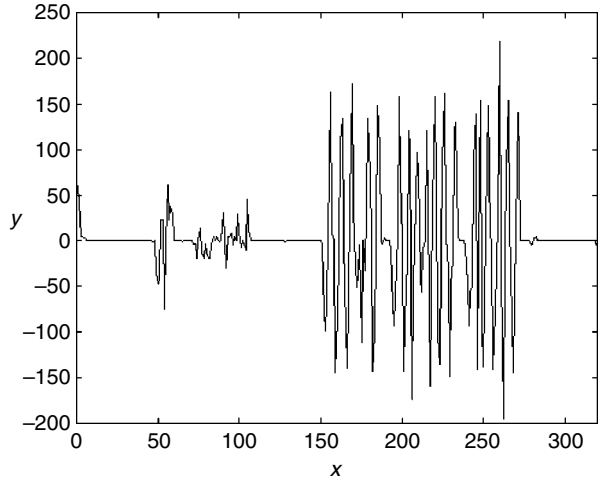



**Figure 3.3** Gradient profile for scan line $y = 80$ for test image 'data13'.

test image in Figure 3.1. Note that the scan line cuts across the 'shrimp' on the left of the image and the words 'that help you' on the right of the image. Large positive spikes on the right (from $x = 155$ to 270) are due to background-to-text transitions, and large negative spikes in the same interval are due to text-to-background transitions. The series of spikes on the left ($x = 50$ to 110) are due to the image of the 'shrimp.' Note that the magnitudes of the spikes for the text are significantly stronger than those of the 'shrimp.' For a segment containing text, there should be an equal number of background- to-text and text-to-background transitions, and the two types of transition should alternate. In practice, the number of background-to-text and text-to-background transitions might not be exactly the same due to processing errors, but they should be close in a text region.

We then threshold the computed MGD values to obtain one or more continuous segments on the scan line. For each continuous segment, the mean and variance of the horizontal distances between the background-to-text and text-to-background transitions on the gradient profile are computed.

A continuous segment is identified as a potential text line segment if these two conditions are satisfied: (i) the number of background-to-text and text-to-background transitions exceeds some threshold; and (ii) the mean and variance of the horizontal distances are within a certain range.

## *3.2 Step 2: Text Block Detection*

In the second step, potential text line segments are expanded or merged with text line segments from adjacent scan lines to form text blocks. For each potential text line segment, the mean and variance of its grayscale values are computed from the grayscale luminance image. This step of the algorithm runs in two passes: top-down and bottom-up. In the first pass, the group of pixels immediately below the pixels of each potential text line segment is considered. If the mean and variance of their grayscale values are close to those of the potential text line segment, they are merged with the potential text line segment to form an expanded text line segment. This process repeats for the group of pixels immediately below the newly expanded text line segment. It stops after a predefined number of iterations or when the expanded text line segment merges with another potential text line segment. In the second pass, the same process is applied in a bottom-up manner to each potential text line segment or expanded text line segment obtained in the first pass. The second pass considers pixels immediately above a potential text line segment or an expanded text line segment.

For images with poor text quality, Step 1 of the algorithm may not be able to detect all potential text line segments from a text string. But as long as enough potential text line segments are detected, the expand-and-merge process in Step 2 will be able to pick up the missing potential text line segments and form a continuous text block.

## *3.3 Step 3: Text Block Filtering*

The detected text blocks are then subject to a filtering process based on their area and height to width ratio. If the computed values fall outside some prespecified ranges, the text block is discarded. The purpose of this step is to eliminate regions that look like text, yet their geometric properties do not fit those of typical text blocks.

## *3.4 Step 4: Boundary Adjustments*

For each text block, we need to adjust its boundary to include text pixels that lie outside the boundary. For example, the bottom half of the vertical stroke for the lower case letter ‘p’ may fall below the baseline of a word it belongs to and fall outside of the detected text block. We compute the average MGD value of the text block and adjust the boundary at each of the four sides of the text block to include outside adjacent pixels that have MGD values that are close to that of the text block.

## *3.5 Step 5: Bicolor Clustering*

In Steps 1–4, grayscale luminance information was used to detect text blocks, which define rectangular regions where text pixels are contained. Step 5 uses the color information contained in a video to more precisely locate the foreground text pixels within the detected text block. We apply a bicolor clustering algorithm to achieve this. In bicolor clustering, we assume that there are only two colors: a foreground text color and a background color. This is a reasonable assumption since in the local region defined by a text block, there is little (if any) color variation in the background, and the text is usually of the same or similar color. The color histogram of the pixels within the text block is used to guide the selection of initial colors for the clustering process. From the color histogram, we pick two peak values

that are of a certain minimum distance apart in the color space as initial foreground and background colors. This method is robust against slowly varying background colors within the text block, since the colors for the background still form a cluster in the color space. Note that bicolor clustering cannot be effectively applied to the entire image frame as a whole, since text and background may have different colors in different parts of the image. The use of bicolor clustering locally within text blocks in our method results in better efficiency and accuracy than applying regular (multicolor) clustering over the entire image, as was done in [10].

## 3.6 Step 6: Artifact Filtering

In the artifact filtering step, non-text noisy artifacts within the text blocks are eliminated. The noisy artifacts could result from the presence of background texture or poor image quality. We first determine the connected components of text pixels within a text block by using a connected component labeling algorithm. Then we perform the following filtering procedures:

(a) If *text_block_height* is greater than some threshold $T1$, and the area of any connected component is greater than *(total_text_area)*/2, the entire text block is discarded.
(b) If the area of a connected component is less than some threshold $T2 =$ (*text_block_height/2*), it is regarded as noise and discarded.
(c) If a connected component touches one of the four sides of the text block, and its size is larger than a certain threshold $T3$, it is discarded.

In Step (a), *text_block_height* is the height of the detected text block, and *total_text_area* is the total number of pixels within the text block. Step (a) is for eliminating unreasonably large connected components other than text characters. This filtering process is applied only when the detected text block is sufficiently large, i.e. when its height exceeds some threshold $T1$. This is to prevent small text characters in small text blocks from being filtered away, as they are small in size and tend to be connected together because of poor resolution. Step (b) filters out excessively small connected components that are unlikely to be text. A good choice for the value of $T2$ is *text_block_height*/2. Step (c) is to get rid of large connected components that extend outside of the text block. These connected components are likely to be part of a larger non-text region that extends inside the text block.

## 3.7 Step 7: Contour Smoothing

In this final step, we smooth the contours of the detected text characters by pruning one-pixel thick side branches (or artifacts) from the contours. This is achieved by iteratively using the classical pruning structuring element pairs depicted in Figure 3.4. Details of this algorithm can be found in [23].

Note that in Step 1 of the algorithm, we compute MGD values to detect potential text line segments. This makes use of the characteristic that text should have both strong positive and negative horizontal

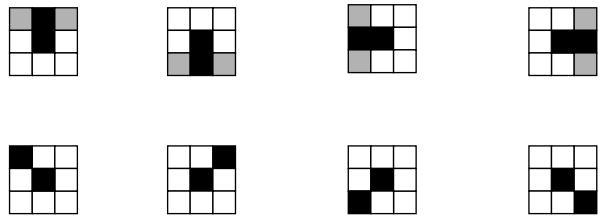


**Figure 3.4** Classical pruning structuring elements.

gradients within a local window. During the expand-and-merge process in the second step, we use the mean and variance of the gray-level values of the text line segments in deciding whether to merge them or not. This is based on the reasoning that text line segments belonging to the same text string should have similar statistics in their gray-level values. The use of two different types of measure ensures the robustness of the algorithm to detect text in complex backgrounds.

## 4. Experimental Results and Performance

We used a total of 225 color images for testing: one downloaded from the Internet, and 224 digitized from broadcast cable television. The Internet image is of size $360 \times 360$ pixels and the video images are of size $320 \times 240$ pixels. The test database consists of a variety of test cases, including images with large and small font text, dark text on light backgrounds, light text on dark backgrounds, text on highly textured backgrounds, text on slowly varying backgrounds, text of low resolution and poor quality, etc. The algorithm performs consistently well on a majority of the images. Figure 3.5 shows a test image with light text on a dark background. Note that this test image contains both large and small font text, and the characters of the word 'Yahoo!' are not perfectly aligned horizontally. Figure 3.6


**Figure 3.5** Test image 'data38'.

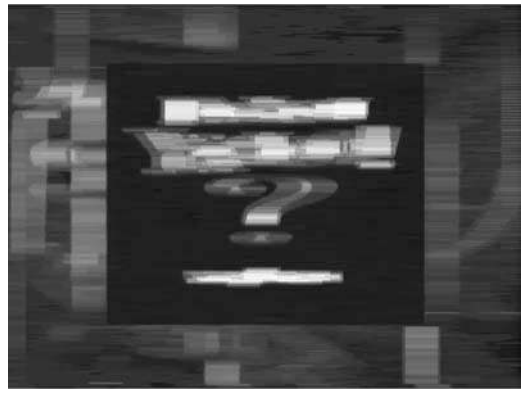


**Figure 3.6** Maximum Gradient Difference (MGD) for image 'data38'.

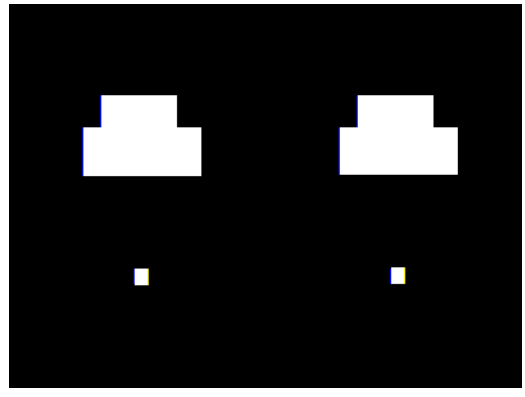


**Figure 3.7** Text blocks detected from test image 'data38'.

shows the result after computing the MGD of the image in Figure 3.5. Figure 3.7 shows the detected text blocks after Step 4 of the algorithm (boundary adjustment). In the figure, the text blocks for the words 'DO YOU' and 'YAHOO!' touch each other and they look like a single text block, but the algorithm actually detected two separate text blocks. Figure 3.8 shows the extracted text after Step 7 of the algorithm. Figure 3.1 showed a test image with dark text on a light colored background. Figure 3.9 shows the extracted text result. Figure 3.10 shows another test image with varying background in the text region. The second row of text contains small fonts that are barely recognizable by the human eye; yet, the algorithm is able to pick up the text as shown in Figure 3.11. Note that the characters are connected to each other in the output image due to poor resolution in the original image.

To evaluate performance, we define two measures: *recall* and *precision*. Recall is defined to be the total number of correct characters detected by the algorithm, divided by the total number of actual characters in the test sample set. By this definition, recall could also be called detection rate. Precision is defined to be the total number of correctly detected characters, divided by the total number of correctly detected characters plus the total number of false positives. Our definitions for recall and

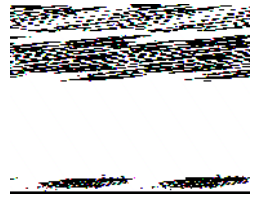


**Figure 3.8** Binary text extracted from test image 'data38'.

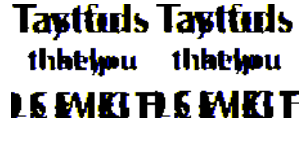


**Figure 3.9** Binary text extracted from test image 'data13'.


**Figure 3.10** Test image 'data41'.


**Figure 3.11** Binary text extracted from test image 'data41'.

precision are similar to those in [18], except that ours are defined for characters, and theirs were defined for text lines and frames. The actual number of characters was counted manually by visually inspecting all of the test images. Our algorithm achieves a recall or detection rate of 88.9 %, and a precision of 95.7 % on the set of 225 test images. Another way to evaluate performance is to compute the number of correctly detected text boxes that contain text, as has been done in some papers when the algorithm's outputs are locations of text boxes. We view character detection rate (or recall) as a stricter performance measure since the correct detection of a text box does not necessarily imply the correct detection of characters inside the text box. Our algorithm has an average execution time of about 1.2 seconds per image (of size $320 \times 240$ pixels) when run on a SUN UltraSpark 60 workstation.

We conducted experiments to evaluate the performance of the algorithm on images that went through JPEG compression and decompression. The purpose is to see whether our text extraction algorithm performs well when blocking effects are introduced by JPEG compression. Eleven images were selected from the test data set for testing. Figure 3.12 shows one of the test images after JPEG compression and decompression (the original is shown is Figure 3.5), and Figure 3.13 shows the text extraction result. Column three of Table 3.1 shows the recall and precision for the 11 images after JPEG compression and decompression. The rates are about the same as those of the original 11 images shown in column two of the same table. This shows that our algorithm performs well on JPEG compressed–decompressed images. Note that the rates shown in column two for the original images are not the same as the rates for the complete set of 225 images (88.9 % and 95.7 %) because the chosen 11 images comprise a smaller subset that does not include images with poor quality text. But for performance comparison with JPEG compressed and decompressed images, and later with noisy images, these 11 images serve the purpose.

We also conducted experiments to evaluate the performance of our algorithm on noisy images. Three types of noise were considered: Gaussian, salt and pepper and speckle. We added Gaussian noise to the same set of 11 test images to generate three sets of 11 noisy images with 30 dB, 20 dB and 10 dB Signal-to-Noise Ratios (SNRs). Figures 3.14 to 3.16 show the noisy images generated from test image 'data38' (shown in Figure 3.5) with SNRs equal to 30 dB, 20 dB and 10 dB, respectively.


**Figure 3.12** Test image 'data38' after JPEG compression–decompression.

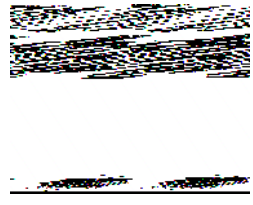


**Figure 3.13** Text extracted from test image 'data38' after JPEG compression–decompression.

**Table 3.1** Recall and precision for the original, JPEG and images with Gaussian noise.

| | Original | JPEG | 30 dB Gaussian | 20 dB Gaussian | 10 dB Gaussian |
|---|---|---|---|---|---|
| Recall | 0.93 | 0.94 | 0.93 | 0.90 | 0.74 |
| Precision | 0.96 | 0.97 | 0.97 | 0.97 | 0.98 |

Figures 3.17 to 3.19 show the test results for the three noisy images respectively. The precision and recall rates for the noisy images are listed in columns 4 to 6 of Table 3.1. From the results, we do not see degradation in performance for 30 dB SNR images. If fact, the precision is slightly higher because after adding noise, some false positives are no longer treated by the algorithm as text. The recall for 20 dB SNR images decreases slightly. Like 30 dB SNR images, the precision also slightly increases. For the very noisy 10 dB SNR images, recall decreases to 74 % and precision increases to 98 %. This shows that the algorithm is robust against Gaussian noise, with no significant degradation in recall for images with up to 20 dB SNR, and with no degradation in precision for images up to 10 dB SNR. For very noisy images with 10 dB SNR, recall decreases to 74 %, indicating that the algorithm can still detect a majority of the text. We also observed that precision slightly increases as SNR decreases in noisy images. Similarly, the performance statistics for images corrupted with salt and pepper noise and speckle noise are summarized in Table 3.2. It can be observed that for salt and pepper noise, the performance at 24 dB and 21 dB SNR is about the same as that of the original images. At 18 dB,


**Figure 3.14** Test image 'data38' with Gaussian noise ($SNR = 30$).


**Figure 3.15** Test image 'data38' with Gaussian noise ($SNR = 20$).


**Figure 3.16** Test image 'data38' with Gaussian noise ($SNR = 10$).

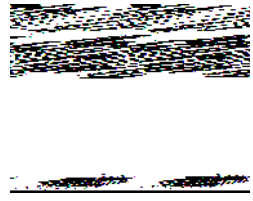


**Figure 3.17** Text extracted from test image 'data38' with Gaussian noise ($SNR = 30$).

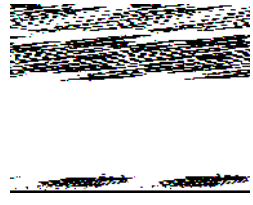


**Figure 3.18** Text extracted from test image 'data38' with Gaussian noise ($SNR = 20$).

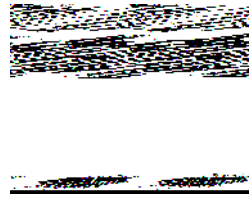


**Figure 3.19** Text extracted from test image 'data38' with Gaussian noise ($SNR = 10$).

**Table 3.2** Recall and precision for images with Salt And Pepper (SAP) and speckle noise.

| | 24 dB SAP | 21 dB SAP | 18 dB SAP | 24 dB Speckle | 16 dB Speckle | 15 dB Speckle |
|---|---|---|---|---|---|---|
| Recall | 0.93 | 0.93 | 0.83 | 0.93 | 0.91 | 0.72 |
| Precision | 0.97 | 0.95 | 0.90 | 0.95 | 0.95 | 0.97 |

the recall and precision drop to 83 % and 90 % respectively. For speckle noise, the performance is about the same as the original at 24 dB and 16 dB SNR. At 15 dB, the recall value drops to 72 %. To save space, we will not show the image results for salt and pepper noise or speckle noise here.

It is difficult to directly compare our experimental results with those of other text detection algorithms, since there does not exist a common evaluation procedure and test data set used by all researchers. A data set containing difficult images, e.g. texts on a low contrast or highly textured background, texts of small font size and low resolution, etc., could significantly lower the performance of a detection algorithm. Here, we cite some performance statistics from other published work for reference. The readers are referred to the original papers for the exact evaluative procedure and definitions of performance measures. In [7], a detection rate of 94.7 % was reported for video frames, and no false positive rate was reported. It was noted in [7] that this algorithm was designed to work on horizontal

text of relatively large size. In [11], a detection rate of 68.3 % was reported on a set of 482 Internet images, and a detection rate of 78.8 % was reported when a subset of these images that meets the algorithm's assumptions was used. No false positive rate was reported. The reported detection and false positive rates in [16] were 93.0 % and 9.2 %, respectively. The output from [16] consists of a set of rectangular blocks that contain text. In [17], high detection rates of 97.32 % to 100 % were reported on five video sequences. No false positive rate was reported. In [18], an average recall of 85.3 %, and a precision of 85.8 % were reported. The outputs from [11,17,18] consist of pixels belonging to text regions (as with our algorithm.) In [20], 95 % of text bounding boxes were labeled correctly, and 80 % of characters were segmented correctly. No false positive rate was reported. In [21], a detection rate of 55 % was reported for small text characters with area less than or equal to ten pixels, and a rate of 92 % was reported for characters with size larger than ten pixels. An overall false positive rate of 5.6 % was reported. In [22], detection and false positive rates of 99.17 % and 1.87 % were reported, respectively, for $8 \times 8$ DCT blocks that contain text pixels. Table 3.3 summarizes the detection and false positive rates for our algorithm and the various text detection algorithms. Note that we have used uniform performance measures of detection rate and false positive rate for all algorithms in the table. The performance measures of recall and precision used in this chapter and in [18] were converted to detection rate and false positive rate by the definition we gave earlier in this section. It should be noted that for many detection algorithms, detection rate could be increased at the expense of an increased false positive rate, by modifying certain parameter values used in the algorithms. The detection rate and false positive rate should therefore be considered at the same time when evaluating the performance of a detection algorithm. Table 3.3 also summarizes the execution time needed for the various text detection algorithms. Listed in the fourth, fifth and sixth columns are the computers used, the size of the image or video frame, and the corresponding execution time for one image frame. Note that our algorithm has comparable execution time with the algorithms in [16,17]. The execution time reported in [7] for a smaller image size of $160 \times 120$ is faster. The algorithm in [21] has a long execution time of ten seconds. The algorithm in [22] has a very fast execution time of 0.006 seconds. Further processing, however, is needed to more precisely locate text pixels based on the DCT blocks produced by the algorithm. Furthermore, the current implementation of the algorithm in [22] cannot extract text of large font size. Unlike our work, none of the above published work reported extensive experimental results for images corrupted with different types and degrees of noise.

**Table 3.3** Performance measures for various text detection algorithms.

| | Detection rate | False positive rate | Computer used | Image size | Execution time |
|---|---|---|---|---|---|
| Our algorithm | 88.9 % | 4.0 % | Sun UltraSparc 60 | $320 \times 240$ | 1.2 s |
| [7] | 94.7 % | NR[b] | Sun UltraSparc I | $160 \times 120$ | 0.09 s |
| [11] | 68.3 %[a]<br>78.8 %[a] | NR | NR | NR | NR |
| [16] | 93.0 % | 9.2 % | Sun UltraSparc I | $320 \times 240$ | 1 s |
| [17] | 97.3 %–100.0 % | NR | Pentium I PC | NR | 1.7 s |
| [18] | 85.3 % | 14.1 % | NR | NR | NR |
| [20] | 95 %80 %[a] | NR | NR | NR | NR |
| [21] | 55 %92 %[a] | 5.6 % | Pentium Pro PC | $320 \times 240$ | 10 s |
| [22] | 99.17 % | 1.87 % | Sun Sparc | $\sim 350 \times 240$ | $\sim 0.006$ s |

[a] See Section 4 for an explanation of entries with two detection rates

[b] NR in the above table indicates 'Not Reported'

## 5. Using Multiframe Edge Information to Improve Precision

In video scenes with complex and highly textured backgrounds, precision of the text extraction algorithm decreases due to false detections. In this section, we describe how we can use multiframe edge information to reduce false detections, thus increasing the precision of the algorithm.

The proposed technique works well when the text is stationary and there is some amount of movement in the background. For many video scenes with complex and highly textured backgrounds, we have observed that there is usually some amount of movement in the background; for example, the 'audience' in a basketball game. In a video with non-moving text, characters appear at the same spatial locations in consecutive frames for a minimum period of time, in order for the viewers to read the text. The proposed technique first applies Canny's edge operator to each frame in the frame sequence that contains the text, and then computes the magnitudes of the edge responses to measure the edge strength. This is followed by an averaging operation across all frames in the sequence to produce an average edge map. In the average edge map, the edge responses will remain high in text regions due to the stationary characteristic of non-moving text. The average edge strength, however, will be weakened in the background regions due to the movements present. We have found that even a small amount of movement in a background region would weaken its average edge strength. Computation of the average edge map requires that we know the location of the frame sequence containing a text string within a video. We are currently developing an algorithm that will automatically estimate the locations of the first and last frames for a text string within a video.

After computing the average edge map of a frame sequence containing a text string, Step 3 of the text extraction algorithm described in Section 3 is modified into two substeps 3(a) and 3(b). Step 3(a) – text block filtering based on geometric properties – is the same as Step 3 of the algorithm described in Section 3. Step 3(b) is a new step described below.

### *5.1 Step 3(b): Text Block Filtering Based on Multiframe Edge Strength*

For every candidate text region, look at the corresponding region in the average edge map computed for the frame sequence containing the text. If the average edge response for that region is sufficiently large and evenly distributed, then we keep the text region; otherwise, the candidate text region is eliminated. To measure whether the edge strength is sufficiently large, we set a threshold $T$ and count the percentage of pixels $C$ that has average edge strength greater than or equal to $T$. If the percentage $C$ is larger than a threshold, then the edge strength is sufficiently large. To measure even distribution, we vertically divide the candidate region into five equal-sized subregions and compute the percentage of pixels $c_i$ with edge strength greater than or equal to $T$ in each region. The edge response is considered to be evenly distributed if $c_i$ is larger than $C/15$ for all $i$s. Here the parameter $C/15$ was determined by experimentation.

Experimental results showed that by using multiframe edge information, we can significantly decrease the number of false detections in video scenes with complex and highly textured backgrounds, and increase the precision of the algorithm. Details of the experimental results can be found in [24].

## 6. Discussion and Concluding Remarks

We have developed a new robust algorithm for extracting text from color video. Given that the test data set contains a variety of difficult cases, including images with small fonts, poor resolution and complex textured backgrounds, we conclude that the newly developed algorithm performs well, with a respectable recall or detection rate of 88.9 %, and a precision of 95.7 % for the text characters. Good results were obtained for many difficult cases in the data set. Our algorithm produces outputs that consist of connected components of text character pixels that can be processed directly by OCR software. The new algorithm performs well on JPEG compressed–decompressed images, and on images

corrupted with Gaussian noise (up to 20 dB SNR), salt and pepper noise (up to 21 dB SNR) and speckle noise (up to 16 dB SNR) with no or little degradation in performance. Besides video, the developed method could also be used to extract text from other types of color image, including images downloaded from the Internet, images scanned from color documents and color images obtained with a digital camera.

A unique characteristic of our algorithm is the scan line approach, which allows fast filtering of scan lines without text when processing a continuous video input stream. When video data is read in a scan line by scan line fashion, only those scan lines containing potential text line segments, plus a few of the scan lines immediately preceding and following the current scan line need to be saved for further processing. The few extra scan lines immediately preceding and following the current scan line are needed for Steps 2 and 4 of the algorithm, when adjacent scan lines are examined for text line segment expansions and text block boundary adjustments. The number of extra scan lines needed depends on the maximum size of text to be detected, and could be determined experimentally.

For video scenes with complex and highly textured backgrounds, we described a method to increase the precision of the algorithm by utilizing multiframe edge information.

## References

[1] Fletcher, L. and Kasturi, R. "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**, pp. 910–918, 1988.

[2] Lovegrove, W. and Elliman, D. "Text block recognition from Tiff images" , *IEE Colloquium on Document Image Processing and Multimedia Environments*, 4/1–4/6, Savoy Place, London, 1995.

[3] Wahl, F. M., Wong, K. Y. and Casey, R. G. "Block segmentation and text extraction in mixed-mode documents," *Computer Vision, Graphics and Image Processing*, **20**, pp. 375–390, 1982.

[4] Lam, S. W., Wang, D. and Srihari, S. N. "Reading newspaper text," in *Proceedings of International Conference on Pattern Recognition*, pp. 703–705, 1990.

[5] Jain, K. and Bhattacharjee, S. "Text segmentation using Gabor filters for automatic document processing," *Machine Vision and Applications*, **5**, 169–184, 1992.

[6] Pavlidis, T. and Zhou, J. "Page segmentation and classification," *CVGIP: Graphic Models and Image Processing*, **54**(6), pp. 484–496, 1992.

[7] Jain, A. K. and Yu, B. "Automatic text location in images and video frames," *Pattern Recognition*, **31**(12), pp. 2055–2076, 1998.

[8] Ohya, J., Shio, A. and Akamatsu, S. "Recognizing characters in scene images," *IEEE Transactions on PAMI-16*, pp. 214–224, 1994.

[9] Haffner, P., Bottu, L., Howar, P. G., Simard, P., Bengio, Y. and Cun, Y. L. "High quality document image compression with DjVu," *Journal of Electronic Imaging, Special Issue on Image/Video Processing and Compression for Visual Communications*, July, 1998.

[10] Zhou, J. Y., and Lopresti, D. "Extracting text from WWW images," in *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, Ulm, Germany, pp. 248–252, 1997.

[11] Zhou, J. Y., Lopresti, D. and Tasdizen, T. "Finding text in color images," in *Proceedings Of the SPIE – Document Recognition V*, **3305**, pp. 130–139, 1998.

[12] Zhong, Y., Karu, K. and Jain, A. "Locating text in complex color images," *Pattern Recognition*, **28** (10), pp. 1523–1535, 1995.

[13] Ariki, Y. and Teranishi, T. "Indexing and classification of TV news articles based on telop recognition," *Fourth International Conference On Document Analysis and Recognition*, Ulm, Germany, pp. 422–427, 1997.

[14] Kim, H. K. "Efficient automatic text location method and content-based indexing and structuring of video database," *Journal of Visual Communication and Image Representation*, **7**(4), pp. 336–344, 1996.

[15] Chaddha, N. and Gupta, A. "Text segmentation using linear transforms," *Proceedings of Asilomar Conference on Circuits, Systems, and Computers*, pp. 1447–1451, 1996.

[16] Li, H. and Doermann, D. "Automatic identification of text in digital video key frames," *Proceedings of IEEE International Conference on Pattern Recognition*, pp. 129–132, 1998.

[17] Shim, J-C., Dorai, C. and Bolle, R. "Automatic text extraction from video for content-based annotation and retrieval," *Proceedings of IEEE International Conference on Pattern Recognition*, pp. 618–620, 1998.

[18] Agnihotri, L. and Dimitrova, N. "Text detection for video analysis," *Workshop on Content-based Access to Image and Video Libraries*, in conjunction with CVPR (Computer Vision and Pattern Recognition), Colorado, June 1999.

[19] Sato, T., Kanade, T., Hughes, E. K. and Smith, M. A. "Video OCR for digital news archive," *Proceedings of IEEE International Workshop on Content-based Access of Image and Video Databases*, pp. 52–60, 1998.

[20] Wernicke, A. and Lienhart, R. "On the segmentation of text in videos," *IEEE Proceedings of International Conference on Multimedia and Expo*, NY, August 2000.

[21] Wu, V., Manmatha, R. and Riseman, E. M. "Textfinder: An automatic system to detect and recognize text in images," *IEEE Transactions On PAMI*, **22**(11), pp. 1224–1229, 1999.

[22] Zhong, Y., Zhang, H. and Jain, A. K. "Automatic caption localization in compressed video," *IEEE Transactions on PAMI*, **22**(4), pp. 385–392, 2000.

[23] Dougherty, E. R. *An Introduction to Morphological Image Processing*, SPIE Press, Bellingham, WA, 1992.

[24] Chen, M. and Wong, E. K. "Text Extraction in Color Video Using Multi-frame Edge Information," in *Proceedings of International Conference on Computer Vision, Pattern Recognition and Image Processing* (in conjunction with Sixth Joint Conference On Information Sciences), March 8–14, 2002.

# 4

# Separation of Handwritten Touching Digits: A Multiagents Approach

**Ashraf Elnagar**
Department of Computer Science, University of Sharjah, P. O. Box 27272, Sharjah, U.A.E

**Reda Al-Hajj**
Department of Computer Science, University of Calgary, 2500 University Dr. NW, Calgary, Alberta, Canada T1N 2N2

*A new approach to separating single touching handwritten digit strings is presented. The image of the connected numerals is normalized, preprocessed and then thinned before feature points are detected. Potential segmentation points are determined, based on a decision line that is estimated from the deepest/highest valley/hill in the image, with one agent dedicated to each. The first agent decides on a candidate cut-point as the closest feature-point to the center of the deepest top-valley, if any. On the other hand, the second agent argues for a candidate cut-point as the closest feature point to the center of the highest bottom-hill, if any. After each of the two agents reports its candidate cut-point, the two agents negotiate to determine the actual cut-point based on a confidence value assigned to each of the candidate cut-points. A restoration step is applied after separating the digits. Experimental results produced a successful segmentation rate of 96 %, which compares favorably with those reported in the literature. However, neither of the two agents alone achieved a close success rate.*

## 1. Introduction

Character recognition in general, and handwritten character recognition in particular, has been an important research area for several decades [1–5]. Machine-typed characters have well-known, easy to detect and recognizable features [6–8]. On the other hand, the difficulty in recognizing handwritten characters is highly proportional to the quality of writing, which ranges from very poor to excellent. However, we cannot avoid having some connected digits, which cannot be recognized automatically

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

before they are separated. Therefore, segmentation of connected handwritten numerals is an important issue that should be attended to.

Segmentation is an essential component in any practical handwritten recognition system. This is because handwriting is unconstrained and depends on writers. It is commonly noted that whenever we write adjacent digits in our day-to-day lives we tend to connect them. Segmentation plays a pivotal role in numerous applications where digit strings occur naturally. For instance, financial establishments, such as banks and credit card firms, are in need of automated systems capable of reading and recognizing handwritten checks and/or receipt amounts. Another application could be seen in postal service departments to sort the incoming mail based on recognized handwritten postal zip codes. Separation can be encountered with other sets of characters, including Hindi numerals, Latin characters, Arabic characters, etc. One crucial application area is handling handwritten archival documents, including Ottoman and earlier Arabic documents, where adjacent characters were written as close to each other as possible. The target was not to leave even tiny spaces that would allow deliberate illegal modifications.

Based on numeral strings' lengths, segmentation methods can be broadly classified into two classes. The first one deals with separating digit strings of unknown length, as in the example of check values. The second class, however, deals with segmenting digit strings with specific length. A variety of applications fall into this class, such as systems that use zip codes and/or dates. Although knowing the length makes the problem simpler than the first class, it remains challenging.

We are proposing an algorithm for separating two touching digits. Our approach is summarized by the block diagram depicted in Figure 4.1. The proposed algorithm accepts a binary image as input, and then normalizing, preprocessing and thinning processes are applied to the image. Next, the segmentation process is carried out. Although thinning is computationally expensive, it is essential to obtaining a uniform stroke width that simplifies the detection of feature points. Besides, parallel thinning algorithms may be used to reduce computational time. We assume that the connected digits' image has reasonable quality and one single touching. Connected digits that are difficult to recognize by humans do not represent a good input for our proposed system. Different people usually write numerals differently. The same person may write digits in different ways based on his/her mood and/or health, which of course adds to the complexity of the segmentation algorithm. The basic idea is to detect feature points in the image and then determine the position of the decision line. The closest locus of feature points specifies potential cut-points, which are determined by two agents. While the first agent focuses on the top part of the thinned image, the other one works on the bottom side of the image. Each one sets, as a candidate cut-point, the closest feature point to the center of the deepest valley and highest hill, respectively. Coordination between the two agents leads to better results when compared to each one alone. Negotiation between the two agents is necessary to decide on the segmentation or cutoff point, which could be either one or a compromise between them. The decision is influenced by a degree of confidence in each candidate cut-point.

The rest of the chapter is organized as follows. Previous work is presented in Section 2. Digitizing and processing are described in Section 3. The segmentation algorithm details are introduced in Section 4. Experimental results are reported in Section 5. Finally, Section 6 includes the conclusions and future research directions.

## 2. Previous Work

A comprehensive survey on segmentation approaches is provided in [9]. An overview of various segmentation techniques for printed or handwritten characters can be found in [10,11]. Touching between two digits can take several forms such as: single-point touching (Figure 4.2), multiple touching along the same contour (Figure 4.3), smooth interference (Figure 4.4), touching with a ligature (Figure 4.5), and multiple touching. Robust segmentation algorithms are the ones which handle a variety

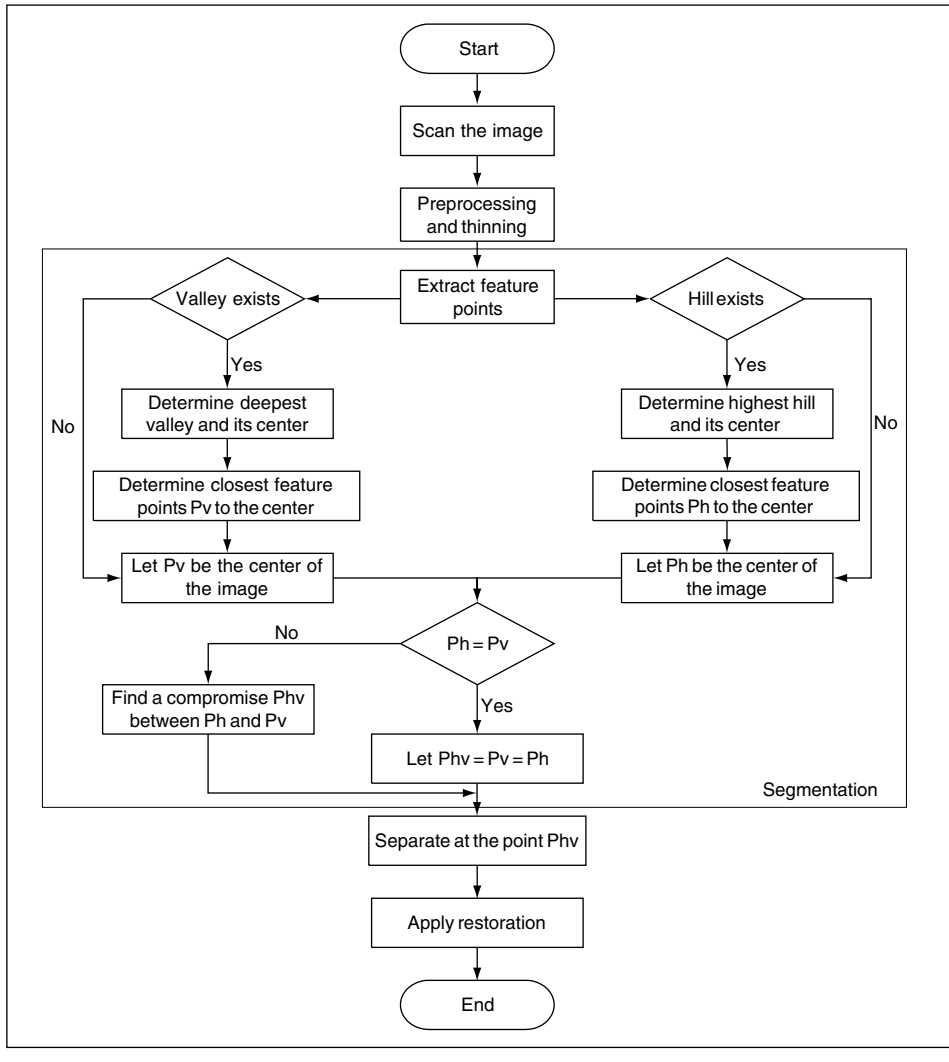



**Figure 4.1** Block diagram of the proposed algorithm.

of these touching scenarios. Segmentation algorithms can be classified into three categories: region-based, contour-based and recognition-based methods. Region-based algorithms identify background regions first and then some features are extracted, such as valleys, loops, etc. Top-down and bottom-up matching algorithms are used to identify such features, which are used to construct the segmentation path. Examples of work reported in this class may be found in [12–14]. However, such methods tend to become unpredictable when segmenting connected digits that share a long contour segment. For example, see Figure 4.3.

Contour-based methods [4,15,16] analyze the contours of connected digits for structure features such as high curvature points [17], vertically oriented edges derived from adjacent strokes [18], number of strokes crossed by a horizontal path [8], distance from the upper contour to the lower one [4], loops and

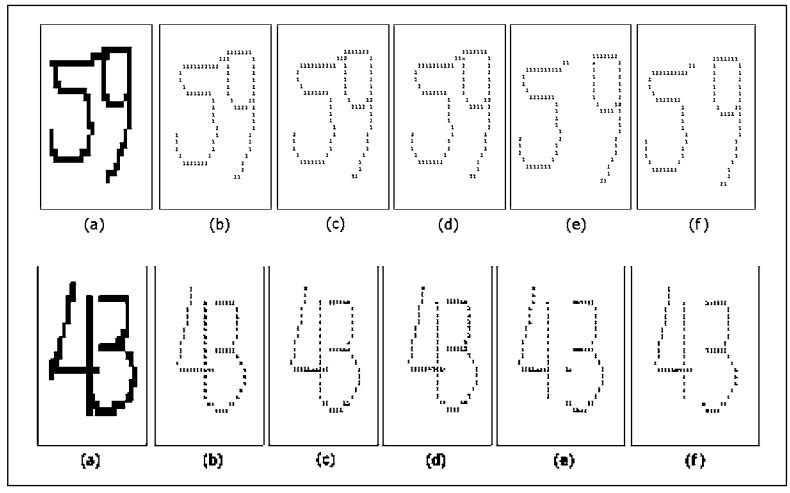



**Figure 4.2** Segmentation steps of numeral strings of Class 1. (a) Original image; (b) output after thinning; (c) extraction of feature points and noise reduction; (d) identifying segmentation points; (e) segmentation result; (f) restoration.

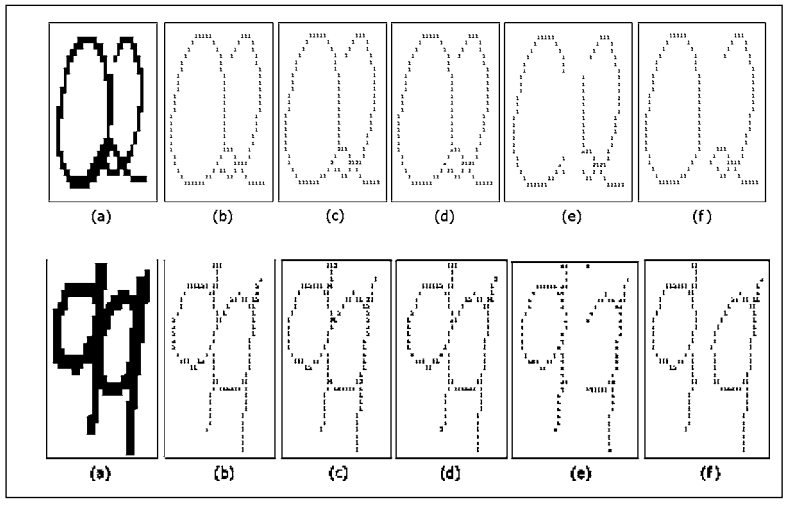



**Figure 4.3** Segmentation steps of two numeral strings from Class 2.

arcs [6], contour corners [17], and geometric measures [19]. However, such methods tend to become unstable when segmenting touched digits that are smoothly joined, have no touching point identified, (Figure 4.4) or have ligature in between (Figure 4.5).

The recognition-based approach involves a recognizer [1,20] and hence it is a time consuming process with the correctness rate highly dependent on the robustness of the recognizer. The work described in [21] handles the separation of single-touching handwritten digits. It simply goes back and

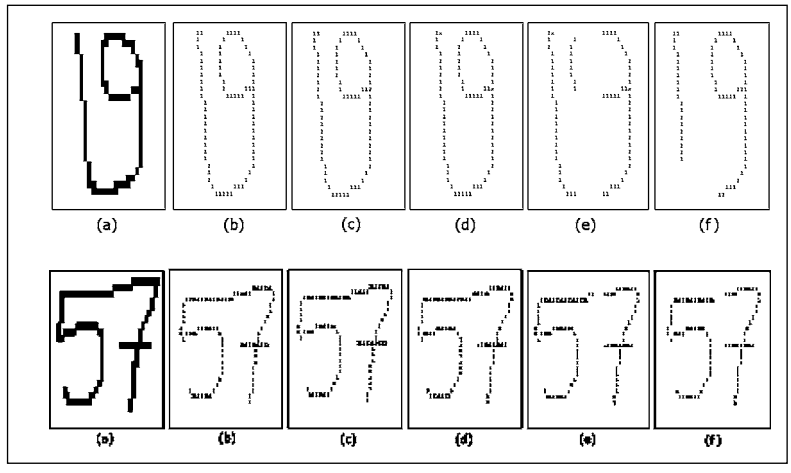



**Figure 4.4** Segmentation steps of two numeral strings from Class 3.

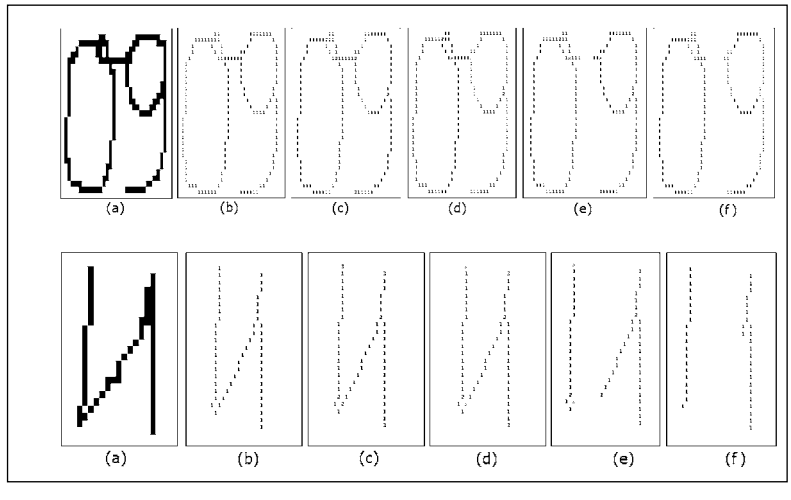



**Figure 4.5** Segmentation steps of two numeral strings from Class 4.

forth between selecting a candidate touching point and recognizing lateral numerals until the digits are recognized.

Finally, the work described in [22] employs both foreground and background alternatives to get a possible segmentation path. One approach for the construction of segmentation paths is discussed in [23]. However, improper segmentation may leave some of the separated characters with artifacts, for example, a character might end up losing a piece of its stroke to the adjacent character. In addition, such methods fail to segment touching digits with a large overlapping contour.

Our approach, which is thinning-based [24], addresses the above-mentioned shortcomings and successfully segments pairs of touching digits under different scenarios, as discussed in this chapter. Our approach reports two results, mainly correct and erroneous segmentation results.

## 3. Digitizing and Processing

Preprocessing of handwritten numerals is done prior to the application of the segmentation algorithm. Besides thinning, this usually involves scanning and normalizing the input image. The output is a binary image of the input numeral image, which has been transformed into a set of simple digital primitives (lines, arcs, etc.) that lie along their medial axes. This process deletes points of a region but it does not remove end points or connectivity, neither does it cause excessive erosion of the region.

Normalization, in general, includes several algorithms to correct the numeral's slant [2], to orient the numeral relative to a baseline, and to adjust its size to a preset standard. In our work, we adjust the size of the input character image to $30 \times 60$ pixels. Therefore, small-size numerals are enlarged and large ones are reduced.

As stated in [25], a good thinning algorithm should preserve the connectivity and reduce the width of the skeleton to unity. Furthermore, the skeleton of the thinned image should not be affected by rotation and translation, should be insensitive to boundary noise, and revolve around its medial axis. A thinning algorithm can be either parallel or sequential. In parallel algorithms, all thinning templates are applied simultaneously. On the other hand, sequential algorithms apply the templates iteratively to the input pattern. In order to reduce the time of processing, we use a parallel thinning algorithm based on the one described in [25]. Figures 4.2–4.5(b) show the resulting skeletons after the application of the thinning algorithm on the input images shown in Figures 4.2–4.5(a).

## 4. Segmentation Algorithm

Character segmentation is an operation that aims to decompose the two-touching-digits image into two subimages. Each resulting subimage contains a digit. Our proposed segmentation algorithm is based on background and foreground features, with agents, and consists of four steps: extraction of feature points and noise reduction, identifying cutoff points, negotiation and restoring the two digits. For example, check the output of Figure 4.2 to Figure 4.5 to study the output of each phase. Spatial domain methods are used to process the input image at each step. Such methods can be simply expressed as:

$$g(x, y) = T(f(x, y)) \tag{4.1}$$

where $f(x, y)$ is the input image, $T$ is an operator and $g(x, y)$ is the output image. Operators are defined as small 2D masks or filters ($3 \times 3$ and $5 \times 5$), referred to as templates too. The value of the coefficients of a certain mask determines the nature of the operator. Figure 4.6 is an example of such operators. In general, segmentation is a time consuming process. However, the use of templates in a proper way can help to fully parallelize this process and therefore cut down tremendously on the computation time cost.

### *4.1 Extraction of Feature Points*

Feature points are extracted from the thinned image. We differentiate between three types of feature point, namely end, branch and cross points. An end point is the start or end of a line segment or an arc. A branch point connects three branches like a capital 'T' rotated by different angles. A cross point connects four branches; it is like a '+' sign, again rotated by different angles.

Feature points are extracted by matching each pixel in the image against all template images of feature points. To obtain all possible templates, each template in Figure 4.6 is rotated by multiples of $(\pi/2)$. Once feature points are identified, they are stored for further manipulation and analysis.

Noise reduction is the process of cleaning up the image from any redundant feature point, which mainly takes the form of a ligature (a curve joining an end point with a branch or a cross point;

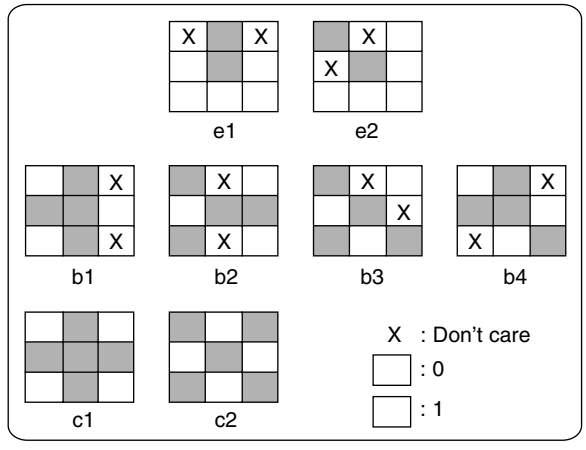



**Figure 4.6** Templates for extracting feature points. End points, branch points and cross points are extracted using templates [e1, e2], [b1, b2, b3, b4] and [c1, c2], respectively.

or a small loop joining two branch points). Such redundant features are eliminated by using specific templates which test for end points and other feature points in a $3 \times 3$ neighborhood. Examples of noise reduction and feature extraction are shown in all figures from Figure 4.2 to Figure 4.5(c). Each end point is marked with '3', whereas branch and cross points are designated by '2'.

In this chapter, we are mainly dealing with single touching numeral strings, which can be classified into the following classes:

1. Single point touches, where the pair of digits are sharing one point. Such a point is a branch or cross feature point (Figure 4.2).
2. Single segment touches, where the pair of digits are sharing part of a stroke (or contour). Such a stroke is identified by a combination of branch and/or cross feature points (Figure 4.3).
3. Smooth touches, where the two digits are sharing, smoothly, one stroke. In this case, no feature points can be identified in the touching area because it is smooth (Figure 4.4).
4. Single touches with ligature, where the two digits are connected together by an extra segment (ligature). Such a stroke may be identified with feature points (Figure 4.5).

Although the proposed algorithm is intended for single touching numeral strings, it can still handle multiple touching numeral strings if the touching points are very close.

## *4.2 The Employed Agents*

Two agents have been employed in the process of separating connected numerals. While the first (top-valley agent) focuses on the top part of the numeral and locates the deepest valley, the second (bottom-hill agent) processes the bottom part of the numeral to locate the highest hill. Both are presented next.

### 4.2.1 Top-Valley Agent

In this section, we present the agent that locates a candidate cut-point by considering characteristics of the upper side of the thinned image. The basic idea is that the segmentation path lies somewhere

close to the deepest top-valley between two connected digits, if there are any. In other words, the closest feature point to the center of the deepest valley lies somewhere along the segmentation path. The success of this agent depends on the availability of at least one valley from the upper side of the image. In the case where no such valley exists, this agent predicts the segmentation path to include the closest feature point to the center of the image. It is identified as candidate cut-point with a low degree of confidence. This will probably be resolved during the negotiation with the other agent, which may report to the negotiation process a candidate cut-point with a higher degree of confidence. We elaborate more on the degree of confidence later in this section.

The top-valley-based agent works as follows. It traces the contour of the thinned image of a handwritten numeral string, starting at the upper leftmost end point, to locate all existing valleys, if any. While doing this, it always keeps track of the deepest encountered valley. The depth of a valley is calculated as the difference between the position of two pixels; the top-most pixel, at which the contour changes direction downwards, and the bottom-most pixel, after which the contour changes direction upwards. Of course, the two highest points on the two sides of a valley are considered in calculating its depth, as formalized below.

There are three important pixels related to each valley; the top pixels on both sides and the deep pixel at the bottom, denoted as $A$, $B$ and $C$, respectively. Each point has two coordinates $x$ and $y$, with the bottom left end point of the image as the origin, the horizontal axis is the $x$-axis, and the vertical axis is the $y$-axis. Then, the depth $D$ of a valley is calculated as follows:

$$D = \min(y_A, y_B) - y_C \tag{4.2}$$

where min(.) is a function that returns the minimum of its arguments. It is important to have $\text{abs}(y_A - y_B) \leq T$, where abs(.) returns the absolute value and $T$ is a threshold chosen as $(2H/3)$, with $H$ denoting the average height of the thinned image.

The center of a valley is calculated as the average distance between each of the corresponding opposite points $L$ and $R$ along the two sides of a valley, from point $C$ up to point $M = \min(A, B)$. Formally,

$$x_{\text{center}} = \text{average}\left(\sum_C^M (x_R - x_L)\right) \tag{4.3}$$

$$y_{\text{center}} = y_C + \frac{D}{2} \tag{4.4}$$

The closest feature point $F$ to the center of the deepest valley is the point with $\min(\text{abs}(x_F - x_{\text{center}}))$ of all feature points. However, if more than one feature point qualify according to this rule, then the one with $\min(\text{abs}(y_F - y_{\text{center}}))$ is selected as a candidate cut-point to be brought up in the negotiation process. The selected candidate cut-point is marked by replacing its feature point marker ('2' or '3') by 'y', and its degree of confidence is calculated as follows:

$$\text{Confidence}(x) = \frac{D}{\min(\text{abs}(x_F - x_{\text{center}}))} \tag{4.5}$$

where $D$ is either the depth of a valley or the height of a hill, depending on the involved agent.

The success of the top-valley-based agent depends on the degree of confidence in the reported candidate cut-point. A low degree of confidence does not lead to successful segmentation of connected digits. As the degree of confidence increases, the segmentation of connected digits becomes more successful. Experimental results showed that segmentation based on a low degree of confidence leads to two distorted parts, which cannot be restored into recognizable digits. This example illustrates the importance of the negotiation process between multiple agents in resolving the conflict.

### 4.2.2 Bottom-Hill Agent

In this section, we present the bottom-hill-based agent. The idea is simple. The original image is inverted (upside down) and the same methodology as the top-valley agent is applied to locate a candidate cut-point. Hence, almost the same problems encountered with the top-valley-based agent can be faced with the bottom-hill-based agent. The center, $\text{center}_\text{v}$, of the deepest valley in the inverted image is reflected into the normal image as the center, $\text{center}_\text{h}$, of the highest bottom-hill. It is formally done by applying the following two formulae:

$$x_{\text{center}_\text{h}} = x_{\text{center}_\text{v}} \tag{4.6}$$

$$y_{\text{center}_\text{h}} = H - y_{\text{center}_\text{v}} \tag{4.7}$$

The selected candidate cut-point is marked by replacing its feature point marker ('2' or '3') by 'x', and its degree of confidence is determined.

As is the case with the top-valley-based agent, the success of the bottom-hill-based agent depends on the degree of confidence in the reported candidate cutoff point.

### 4.2.3 Negotiation and Conflict Resolution

After each agent derives its input required for the negotiation process, namely the two candidate cut-points marked as 'x' and 'y' on the thinned connected image, the two agents cooperate to resolve the conflict, if any. Although every agent concentrates on its reported cut-point with the highest confidence, each agent keeps track of the confidence of the other candidate cut-point sorted in descending order by confidence. The latter candidate cut-points are utilized only in cases where the other agent does not agree on the cut-point with the highest confidence. In this case, a cut-point is selected as an actual cut-point only if the sum of its confidence values reported by both agents is maximal.

Every agent starts by declaring its detected candidate cut-point with the highest confidence. There are three possible cases to consider. The two agents may report:

- the same candidate cut-point;
- two candidate cut-points with the same $x$ coordinate and different $y$ coordinates; or
- two candidate cut-points with different $x$ coordinates and different $y$ coordinates.

The last case is the worst, where negotiation and conflict resolution is employed. The first two cases increase the degree of confidence in the candidate cut-point and the image is separated vertically along the reported $x$ coordinate. However, the degree of confidence of each agent in its proposed candidate cut-points is important.

As the degree of confidence increases, the agent gets higher priority during the negotiation process. If the depth of the reported valley or the height of the reported hill is zero, then the degree of confidence also reaches zero, giving little chance to the corresponding agent during the negotiation process.

What remains are touching cases that cause conflict. The two agents negotiate to resolve this conflict based on the calculated degrees of confidence in each of the two reported candidate cut-points.

1. Let $F_\text{v}$ and $F_\text{h}$ be the two reported candidate cut-points.
2. If $\text{Confidence}(F_\text{v}) = \text{Confidence}(F_\text{h})$ then the actual cut-point is reported as having

$$x_{\text{actual}} = \frac{x_{F_\text{v}} + x_{F_\text{h}}}{2} \tag{4.8}$$

3. ElseIf Confidence($F_v$) > Confidence($F_h$) then the actual cut-point is reported as having

$$x_{\text{actual}} = x_{F_v} \times \frac{1 \mp \dfrac{\text{Confidence}(F_h)}{\text{Confidence}(F_v)}}{2} \tag{4.9}$$

4. ElseIf Confidence($F_h$) > Confidence($F_v$) then the actual cut-point is reported as having

$$x_{\text{actual}} = x_{F_h} \times \frac{1 \mp \dfrac{\text{Confidence}(F_v)}{\text{Confidence}(F_h)}}{2} \tag{4.10}$$

Notice the $\mp$ sign above, where the degree of confidence related fraction is either added or subtracted. It is added in the case where the other candidate cut-point ($F_v$ or $F_h$) lies on the right-hand side of the candidate cut-point ($F_h$ or $F_v$) of the $x$ coordinate used in the right-hand side of the equation; it is subtracted otherwise.

The negotiation and conflict resolution process has been successfully applied to different images with single touching, multiple touching and long touching segments and overlapping. Examples will be introduced in Section 5.

### 4.2.4 Restoration

Segmenting the image into two parts by a vertical line through the already selected actual cutoff point will lead to several situations. The first is to have a single connected component for each numeral, where some parts could be shared between the two numerals (Algorithm Restore is used). Another result is to have a single connected part on one side and a single connected part with some small isolated strokes on the other side. This requires restoration to be applied after moving the extra strokes to the other side. The last scenario is to have two or more isolated parts on one side and a single connected part on the other side. In such a case, slices of the connected part are moved to the other side until some feature point(s) is (are) encountered on the connected side, or the other side becomes a single connected component. Then, Algorithm Restore is applied.

Algorithm Restore investigates the possibility of restoring each part of a segmented image into a recognizable digit by moving necessary strokes between the two parts. However, Algorithm Restore utilizes what we call artificial end points. An end point, $e$, is said to be artificial if and only if:

1. $e$ lies in one of the two parts of a segmented image;
2. $e$ faces the other part; and
3. $e$ is not marked as '3'.

**Algorithm Restore**

```
Input: The two parts obtained after initial segmentation.
Output: The two parts (hopefully digits) after restoration.
Algorithm:

1. Consider each of the two parts after initial segmentation;
2. Locate artificial end points in each part;
```

```
3. If there exists a single artificial end point, e, in any of the two
   parts then:
   • there is a corresponding artificial end point, e1, in the other part
     such that abs (ye − ye1) ≤ 1;
   • locate stroke s connecting e to the first encountered feature point
     f along the contour of e's part;
   • locate stroke s1 connecting e1 to the first encountered feature
     point f1 along the contour of e1's part;
       If f is an end feature point (marked '3') then
         Move s to e1's part;
       ElseIf f1 is an end feature point then
         Move s1 to e's part;
       ElseIf (Length (s) > Length (s1)) && (Length (s1) < threshold) then
         Move s1 to e's part;
       ElseIf (Length(s) < threshold) then
         Move s1 to e1's part;
       EndIf
   Else
   • Consider artificial end points in each part in top-down order;
   • For each two consecutive artificial end points, e and j, in the same
     part do
     ◦ Fill the gap between e and j by moving a corresponding stroke from
       the other part;
     ◦ If there are, not necessarily in the same part, two or more
       feature points m and n between e and j and neither m nor n is marked
       '3' then
           The stroke that lies between m and n (inclusive) must be
           duplicated in both parts;
         EndIf
     EndFor
   EndIf .
```

Noise may be present between connected digits like bridging strokes. Our proposed segmentation algorithm successfully eliminates extra strokes by applying some rules. In other words, there are cases where there are some extra strokes located on the right-most side of the left part, or on the left-most side of the right part. Such strokes are identified as follows.

Generate the (horizontal) histogram and then eliminate any stroke located on the right-most side of the left part or on the left-most side of the right part with a small number of 1s per column compared to the maximum number of 1s in a single column along the whole considered part, where this is the only stroke across the height of its part. These types of extra stroke were not considered in [22], which considered only special extra strokes and mainly those involving digits '0', '8', '6' and '9'. Our approach is more general to cover many other cases.

## 5. Experimental Results

Experiments were conducted using our own 4095 images written by 150 different people. The images are all handwritten numeral strings with possible touchings from different categories, including single touching, multiple touching, long touching and overlapping. All our images were scanned on a 600 dpi scanner and stored as Bitmap files. Some examples are shown in Figure 4.7.

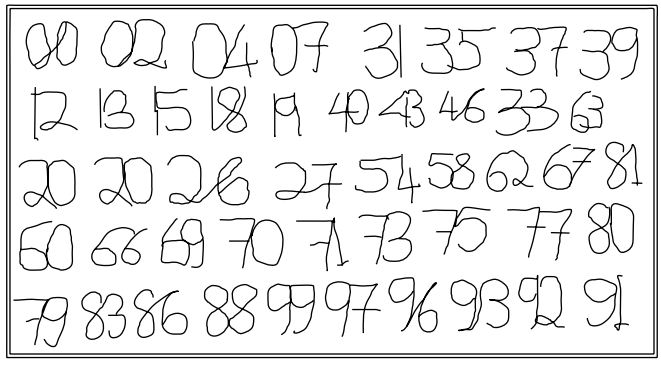


**Figure 4.7** Some examples of the numeral strings from the test database.

The segmentation results are verified by a recognition system [26]. The performance of the proposed segmentation algorithm relies on the recognition and manual checking of the rejected and/or misrecognized digits. However, recognition does not affect the decision making of the segmentation process. It is just used for verifying the results produced by the proposed segmentation algorithm. Examples of the segmentation algorithm are depicted in Figures 4.8–4.11. Some of these samples are unsuccessfully segmented by other proposed segmentation algorithms in the literature. Examples of unsuccessful segmentation results obtained by the proposed algorithm are depicted in Figure 4.12. This is due to the overlap between the two numeral digits. In particular, when the overlap results in a hole as illustrated in the same figure.

It is worth noting that although the proposed algorithm is intended for separating single touching digits, it has the potential to separate multiple touching digits, as shown in Figure 4.13, provided that the touching points are close.

Table 4.1 shows a performance evaluation comparison among different segmentation algorithms, of which some use recognition as a deciding factor in the segmentation process. The overall performance of our algorithm is 96 %, which compares favorably with existing ones in the literature.

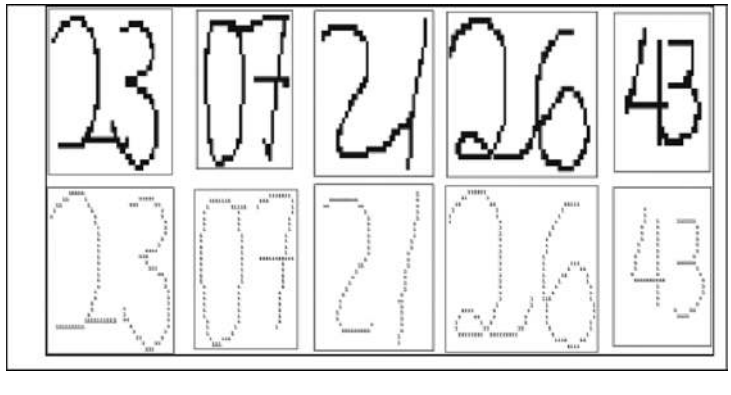


**Figure 4.8** Examples of segmentation results from Class 1.

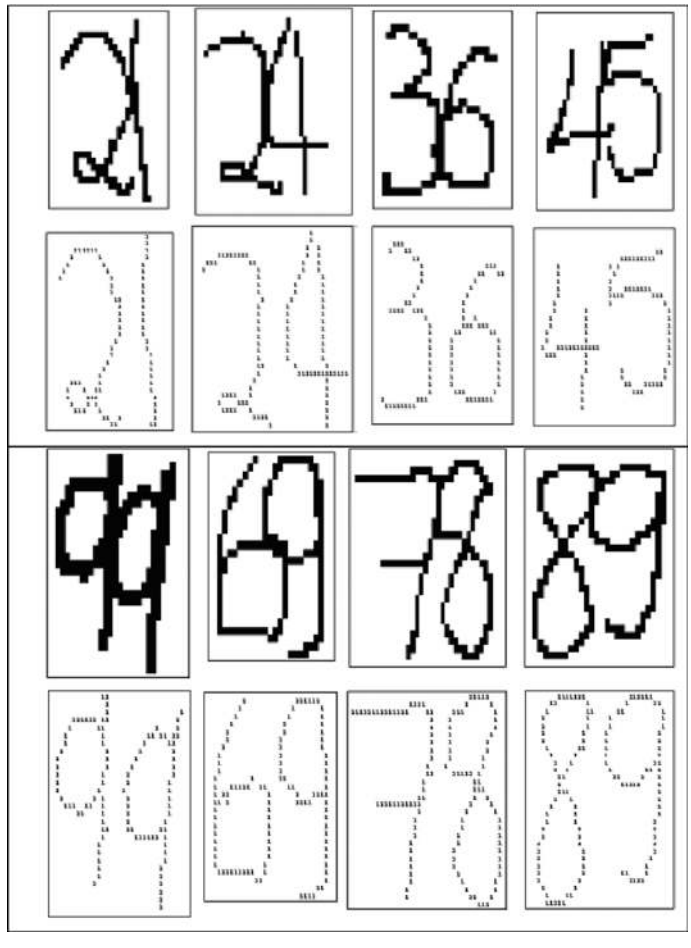


**Figure 4.9** Examples of segmentation results from Class 2.

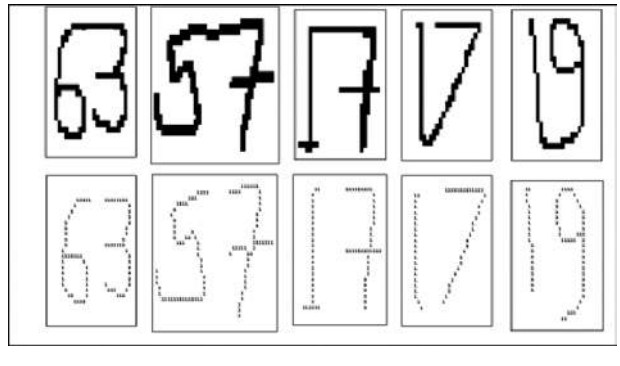


**Figure 4.10** Examples of segmentation results from Class 3.

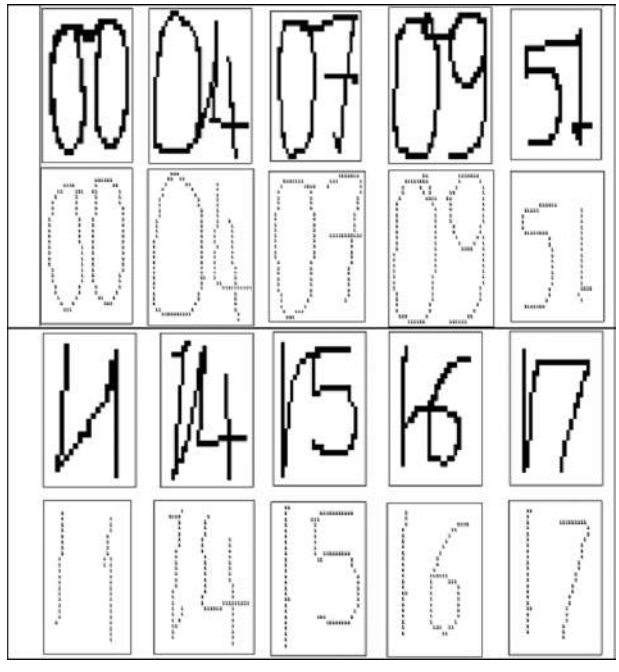


**Figure 4.11** Examples of segmentation results from Class 4.

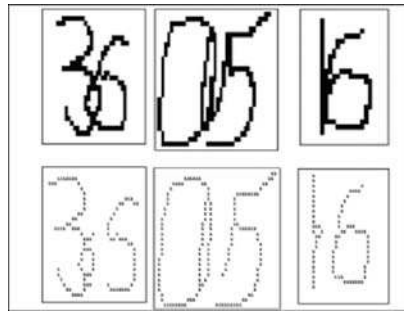


**Figure 4.12** Examples of unsuccessful segmentation.

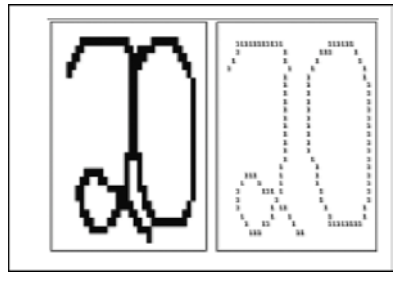


**Figure 4.13** Example of successful multiple touching segmentation.

**Table 4.1** A comparative performance evaluation.

| Algorithm | Success segmentation rate |
|---|---|
| [20] | 75.9 % |
| [17] | 89.0 % |
| [15] | 95.1 % |
| [11] | 92.5 % |
| [16] | 85.7 % |
| Proposed algorithm | 96.2 % |

## 6. Conclusions and Future Work

We elaborated on the importance of utilizing multiple agents in recognizing touching in handwritten numeral strings. We mainly introduced two agents that we have developed to work on the thinned image. Every agent looks at the image from a different perspective and detects a candidate cut-point, which may lead to correct segmentation based on a computed degree of confidence. In general, the derived two candidate cut-points are not the same. The difference requires the two agents to negotiate and resolve the conflict and decide on the actual cut-point. The experiments conducted were successful and encouraging. We realized that each of the two agents was not able to achieve on its own the same recognition rate achieved after the cooperation and negotiation strategy was applied. Finally, our approach successfully removes extra strokes, if any.

We are planning to test more agents that will help to increase the correct rate. Adding more agents will improve the result without affecting the performance because agents work in parallel. Another area that we are concentrating on is to apply the same approach to other sets of characters, including Latin and Arabic.

## References

[1] Bae, J. H., *et al.* "Segmentation of Touching Characters using an MLP," *Pattern Recognition Letters*, **19**, pp.701–709, 1998.

[2] Kim, G. and Govindaraju, V. "Handwritten word recognition for real-time applications," *Proceedings of the International Conference on Document Analysis and Recognition*, **1**, pp. 24–27, 1995.

[3] Kim, H. J. and Kim, P. K. "On-line recognition of cursive Korean characters using a set of extended primitive segments and fuzzy functions," *Pattern Recognition Letters*, **17**, pp. 19–28, 1996.

[4] Fujisawa, H., Nakano, Y. and Kurino, K. "Segmentation methods for character recognition: from segmentation to document structure analysis," *Proceedings of IEEE*, **80**(7), 1079–1092, 1992.

[5] Lee, E. and Choi, H. I. "Recognition of Touching Characters Using Partial Prototypes," *Proceedings of ICCPOL*, March 1999.

[6] Nishida, H. and Mori, S. "Model-based-split-and-merge method for recognition and segmentation of character strings," *Advanced Structural Pattern Recognition*, **5**, pp. 301–309, 1992.

[7] Siy, P. and Chen, C. "Fuzzy logic for handwritten numeral character recognition," *IEEE SMC*, **4**, pp. 570–575, 1974.

[8] Shridhar, M. and Badreldin, A. "Context-directed segmentation algorithm for hand-written numeral strings," *Image Vision Computing*, **5**(1), pp. 3–8, 1987.

[9] Casey, R. and Lecolinet, E. "A survey of methods and strategies in character segmentation," *IEEE PAMI*, **18**(7), pp. 690–706, 1996.

[10] Lu, Y. "Machine printed character segmentation – an overview," *Pattern Recognition*, **28**(1), pp. 67–80, 1995.

[11] Lu, Y. and Shridhar, M. "Character segmentation in handwritten words – an overview," *Pattern Recognition*, **29**(1), pp. 77–96, 1996.

[12] Shridhar, M. and Badreldin, A. "Recognition of Isolated and Simply Connected Handwritten Numerals," *Pattern Recognition*, **19**, pp. 1–12, 1986.

[13] Cheriet, M., Huang, Y. S. and Seun, C. Y. "Background Region-Based Algorithm for the Segmentation of Connected Digits," *Proceedings of ICPR*, pp. 619–622, 1992.

[14] Lu, Z., Chi, Z. and Shi, P. "A Background Thinning-Based Approach for Separating and Recognizing Connected Handwritten Digit Strings," *Pattern Recognition*, **32**(6), pp. 921–933, 1999.

[15] Chi, Z., Suters, M. and Yan, H. "Separation of Single and Double Touching Handwritten Numeral Strings," *Optical Engineering*, **34**, pp. 159–165, 1995.

[16] Shi, Z. and Govindaraju, V. "Segmentation and Recognition of Connected Handwritten Numeral Strings," *Pattern Recognition*, **30**, pp. 1501–1504, 1997.

[17] Strathy, N., Suen, C. and Krzya, A. "Segmentation of handwritten digits using contour features," *Proceedings of the International Conference on Document Analysis and Recognition*, **2**, pp. 577–580, 1993.

[18] Kimura, F. and Shridhar, M. "Segmentation-recognition algorithm for handwritten numeral strings," *Machine Vision Application*, **5**, pp. 199–210, 1992.

[19] Zhao, B., Suters, M. and Yan, H. "Connected handwritten digit separation by optimal contour partition," *Proceedings of the International Conference on Digital Image Computing Techniques and Applications*, **2**, pp. 786–793, 1993.

[20] Fenrich, R. "Segmentation of Automatically Located Handwritten Numeric Strings," in Impedovo, S. and Simon, J.C. (Eds) *From Pixels to Features III: Frontiers in Handwriting Recognition*, Elsevier, Amsterdam, pp.47–59, 1992.

[21] Yu, D. G. and Yan, H. "Separation of Single-Touching Handwritten Numeral Strings Based on Structural Features," *Pattern Recognition*, **31**, pp. 1835–1847, 1998.

[22] Chen, Y. K. and Wang, J. F. "Segmentation of Single or Multiple Touching Handwritten Numeral Strings Using Background and Foreground Analysis," *IEEE-PAMI*, **22**(11), pp. 1304–1317, 2000.

[23] Hu, J., Yu, D. and Yan, Y. H. "Construction of partitioning paths for touching handwritten characters," *Pattern Recognition Letters*, **20**, pp. 293–303, 1999.

[24] Elnagar, A. and Alhajj, R. "Segmentation of Connected Handwritten Numerals," *Pattern Recognition*, **36**, pp. 625–634, 2003.

[25] Jang, B. and Chin, R. "One-Pass Parallel Thinning: Analysis, Properties, and Quantitative Evaluation," *IEEE-PAMI*, **14**(11), pp. 1129–1140, 1992.

[26] Elnagar, A., Al-Kharousi, F. and Harous, S. "Handwritten Arabic and Hindi numerals recognition based on a decision tree classifier," *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, **2**, pp. 983–988, 1997.

# 5

# Prototype-based Handwriting Recognition Using Shape and Execution Prototypes

**Miguel L. Bote-Lorenzo**
**Eduardo Gómez-Sánchez**
**Yannis A. Dimitriadis**
Department of Signal Theory, Communications and Telematics Engineering
School of Telecommunications Engineering
University of Valladolid, Spain

*A novel prototype-based handwriting recognition system is explained in detail, covering all stages. First, a segmentation method based on handwriting generation theories is illustrated. A small set of informative features representing the character is then constructed. Then, a prototype extraction method is carried out, in two stages. First, using Fuzzy ARTMAP, we group character instances according to classification criteria. Then, an algorithm refines these groups and computes the prototypes. This method is illustrated with examples using digits, lower-case and upper-case letters from the UNIPEN international database. When these prototypes are used to initialize the recognizer, an average recognition rate of 90.15 % is achieved, which is comparable to that reached by human readers.*

## 1. Introduction

Online handwriting recognition has been a hot research topic for a long time [1,2], but lately it has become more important due to the expansion of pen-based devices, such as cell phones, Personal Digital Assistants (PDAS) and pocket PCs, in which handwriting is the most ergonomic way for humans to interact with machines. However, though simple for humans, automatic character recognition is regarded as one of the most difficult problems in the field of pattern recognition. This is because handwritten characters present an intrinsic variability that stems from the complexity of the handwriting generation process [3]. More specifically, variability is due to the existence of more than one shape

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

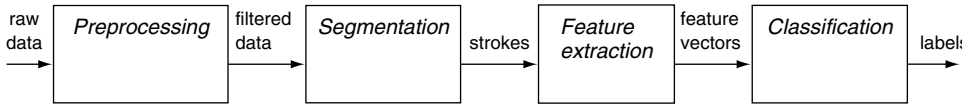



**Figure 5.1** Tasks commonly performed in handwriting recognition systems.

to represent most characters, the possibility of tracing a character shape in several different ways (e.g. upwards vs. downwards, clockwise vs. counterclockwise), and the noise introduced by writers when drawing the shape. Authors mentally choose a shape for the character they are going to write, then plan how they will trace it (e.g. upwards, clockwise, etc.) and finally move their hand in response to a sequence of very simple impulses [3,4].

Therefore, online handwriting recognition systems that take this fact into account, usually consist of a number of stages, depicted in Figure 5.1, that try to invert the handwriting recognition process [2]. *Filtering and preprocessing* of data is performed in order to remove noise from the electronic device capturing the pen movement, or to perform some coordinate transformation that reduces the impact of escalation, rotation or translation of the handwritten path along the digitizing pad. Afterwards, since the actual character path traced results from the superimposition of fast, simple hand movements, the character can be broken into the strokes, which are pieces of handwriting generated by these movements [3], through a *character segmentation* process [4]. Then, a *feature extraction* process is convenient to find out a representation of the strokes in terms of informative features, such as pad coordinates, length, angles, trace velocity, etc., that will allow the classifier to distinguish one stroke from another, and thereafter one character from another. Finally, the *classification* stage is a general pattern recognition problem, for which many different techniques can be suitable [5]. There are many examples in the literature, from statistical [6] or syntactic recognition [7], to the use of neural networks [8] or neuro-fuzzy systems [9]. Though each approach has its advantages, a broad number of techniques perform prototype-based classification [10–12], which is closer to the handwriting generation theories, i.e. recognizing a character consists of making a guess on which prototype the writer had in mind when he generated the actual instance to be labeled [11].

In this chapter we will describe a prototype-based classifier that is based on the well-known LVQ algorithm [13], but initialized with the prototypes extracted through a two-stage process that we presented in [14]. Though many other initialization techniques are available, we shall show that our prototype extraction method has several advantages: it determines the number of prototypes from the complexity of the data and extracts prototypes that are human-recognizable, thus enabling uses of the prototypes other than initializing a classifier.

This chapter will continue with a brief explanation of the relationship between the handwriting generation process and how handwriting recognition can be performed, in Section 2. Then, Section 3 describes how characters are segmented into strokes, using a method we proposed in [15] that is inspired by handwriting generation theories. In addition, the feature extraction process carried out in our study will be explained. Section 4 describes in detail the prototype extraction method that we proposed in [14] and evaluates its suitability in terms of the number of prototypes generated and their visual appearance. Section 5 will use these prototype collections to implement a prototype-based classifier that will be evaluated on unseen real data. Finally, Section 6 will present the main conclusions of this work.

## 2. A Handwriting Generation Process Model

Usually, each character concept may be written using several different shapes. For example, the letter concept {a} can be written in different ways, such as an upper case, a block printed or a cursive variant. The digit concept {0} may in turn be drawn using a normal zero or a slashed zero. Each of the shape

models that a writer can choose to represent a character concept is called a *character allograph* [7,16], i.e. an allograph may be considered as the shape model or shape prototype that the writer tries to imitate in order to represent the character concept.

Once the *choice of the shape model* or allograph of the character concept to be represented has been made, there is the matter of how the writer will trace that shape. Character allographs may be drawn in several ways; for example, the drawing of the normal zero allograph can be started upwards or downwards and may be done clockwise or counterclockwise. Before starting to write, the author must *choose an execution plan* defining the order and sequence of the allograph's strokes [3,17]. Thus, we may give the label *character shape and execution prototype* (or execution prototype for brief) to the model that provides information about the shape model that must be drawn, as well as about the order of the strokes the shape may be divided into. In the *handwriting execution* process, the writer will draw a *character instance*, trying to make it as similar as possible to the model given by the character shape and execution prototype.

Conversely, *handwriting recognition* consists of labeling character instances in order to recover the character concept from the instance. Therefore, the *execution prototype extraction method* described here would allow identification of the different execution prototypes of a character concept, as illustrated in Figure 5.2.

Three sources of variation in handwriting can thus be identified in this handwriting generation model: allograph variation, execution plan variation and instance variability. *Allograph variation* refers to the large amount of different shapes used by individuals to represent character concepts. The shapes used by the writer depend mainly on his education at primary school and personal preferences. *Execution plan variation* is related to the different possible ways of drawing a given shape. Again, execution plans used by writers depend on education, as well as on the context of neighboring letters. Finally, *instance variability* holds for a given writer, and refers to the 'noise' (e.g. different slants and sizes, movement noise) introduced by the author when trying to reproduce a character execution prototype.

To suit a prototype-based classifier, a character shape and execution prototype extraction method should retain both allograph and execution plan variation, and reject instance variability, i.e. the extraction method must try to find a representative prototype for each group of character instances that has been drawn following a given allograph and execution plan. In such a task, knowledge about characters' shape and execution existing in a large number of character instances is condensed in a collection of prototypes that can be used in the classifier.

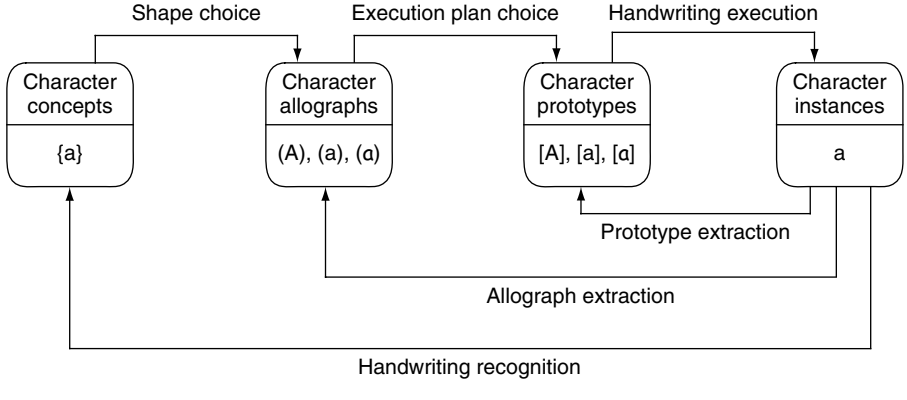



**Figure 5.2** A conceptual model of the handwriting generation process that shows how handwriting recognition is related through allographs and execution prototypes.

This chapter will describe the automatic shape and execution prototype extraction method that we proposed in [14], serving several purposes. First, it can help to formalize the *study of the different handwriting styles* used by humans according to several factors like age or culture [18]. In addition, the study of execution plans is a very important topic in *handwriting generation research* [1,19]. Finally, although human recognizable shape and execution prototypes are not *a priori* thought of for character recognition, the knowledge they enclose may be used to initialize *online handwriting recognition* architectures [11]. Therefore, the method described here has been designed to meet three objectives: a *low number* of prototypes, *human recognizable* prototypes and *knowledge condensation*.

## 3. The First Stages of the Handwriting Recognition System

As stated in the introduction, a common handwriting recognition system consists of a preprocessing stage, in which raw data is filtered, scaled or otherwise prepared; a segmentation stage, to divide characters into strokes; a feature extraction stage, in which an informative feature set is built to represent characters; and the actual classification stage.

In the research presented here, the preprocessing was quite simple: an FIR low-pass filter was applied to the velocity signal, with a cutoff frequency of 10 Hz [20] and a window length of $f/10$, where $f$ is the sampling frequency of the digitizer pad. After this preprocessing, the segmentation takes place using the method we proposed in [15], also described in Section 3.1. Characters were then represented by a few features, selected as explained in Section 3.2.

### *3.1 Character Segmentation*

The objective of segmentation is to divide complex handwriting pieces into simpler ones, in order to reduce input pattern variability and thus simplify the classifier structure. In this work, data are isolated characters composed of one or more components (a component is the piece of handwriting between each *pen-down* and its successive *pen-up*), which are to be segmented into strokes. Segmentation methods can be derived from handwriting generation theories, as well as heuristic studies of the geometric features in handwritten specimens.

According to [3,4], handwriting is generated by a sequence of superimposed simple movements, each with a delta-lognormal velocity profile. Some examples of such profiles are shown in Figure 5.3.

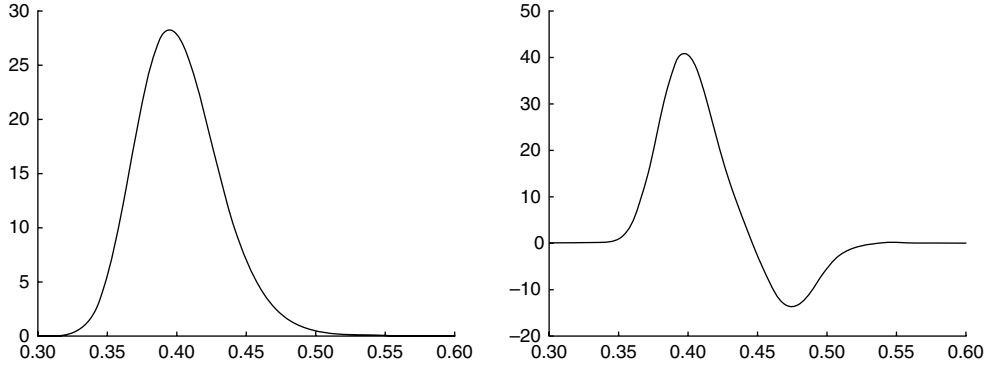



**Figure 5.3** Some typical delta-lognormal velocity profiles (velocity in cm/s, time in s). These profiles describe movements produced by the synergy of an agonist and an antagonist system synchronously activated with a pair of impulse commands, each described by a lognormal impulse response [3].

Segmentation procedures should then divide the velocity profile of a component into simpler delta-lognormal profiles, each corresponding to one stroke.

It can be observed that the first part (with the form 'c') of the trajectory of several characters is similar, e.g. in 'a', 'c', 'd', 'g', 'o' and 'q'. It should therefore be reasonable to expect that any character could be built from a finite base of strokes [21]. Then, segmentation should aim to discover such a stroke base, assuming that character prototypes correspond to sequences of strokes picked up from the aforementioned base.

Besides the extensive use of this generic approach in oriental character recognition, biology-inspired segmentation methods based on minima in velocity profiles have been proposed in the literature [10,22]. These algorithms require preprocessing of real data, and therefore the quantification of the signal, inaccurate sampling rates or human trembling may introduce irregularities.

The character instances used in the experiments described here have been segmented using a method inspired by handwriting generation theories [3,4]. This method tries to eliminate perturbations in the velocity signal with minimum modifications of the character geometry. After the preprocessing stage, angle $\theta(t)$, angular velocity $\theta'(t)$ and linear velocity $v(t)$ signals are obtained according to:

$$\theta(t) = \arctan\frac{v_x(t)}{v_y(t)} \approx \arctan\frac{\Delta y(t)}{\Delta x(t)} \tag{5.1}$$

$$\theta'(t) \approx \Delta\theta(t) \tag{5.2}$$

$$v(t) = \sqrt{v_x(t)^2 + v_y(t)^2} \approx \sqrt{(\Delta x(t))^2 + (\Delta y(t))^2} \tag{5.3}$$

This information is jointly exploited in order to obtain the segmentation points, as follows:

- Initial candidates are determined according to singularities of the angle signal, which are crosses with 0, $90°$ and $-90°$. Crosses with 0 will determine peaks (due to sudden changes in orientation), loops (in which there is a transition to a different angular region), and crosses with horizontal baselines. Also, crosses with $90°$ or $-90°$ can be used to detect crosses with vertical baselines. These candidates can be seen in Figure 5.4(a) and Figure 5.4(b) for a sample of letter 'a'.
- In this step, we look for extrema of the angular velocity signal in the vicinity of the candidate points chosen in the previous step. This second criterion is not used independently because of its extreme sensibility to possible perturbations that might have passed through the low-pass filter stage. The output of this stage is shown in Figure 5.4(c) and Figure 5.4(d) for the same character as above.
- Finally, minima in linear velocity next to remaining candidates determine the final set of segmentation points. The use of this information, which is theoretically dual to extrema in angular velocity, permits us to reduce the number of local singularities that were previously accepted as segmentation points. In Figures 5.4(e) and 5.4(f), the final segmentation points are shown for our example.
- Besides this basic segmentation procedure, we can use any available *pen-up* information in order to delete small strokes that are due to a too early contact of the pen with the pad or a too late withdrawal, at the beginning or at the end of the component respectively. Information on the linear velocity profile can also be used to delete small pulses (ornamental strokes), as shown in Figures 5.4(g) and 5.4(h).

In [15] we showed how this segmentation method is computationally simple, consistent (similar characters are segmented into similar strokes), and produces a reduced number of strokes (which is desirable in order to achieve a small number of character prototypes).

## *3.2 Feature Extraction*

Feature extraction or codification is necessary in order to cluster strokes within our effort to build a base of character primitives and classify stroke sequences that correspond to complete characters.

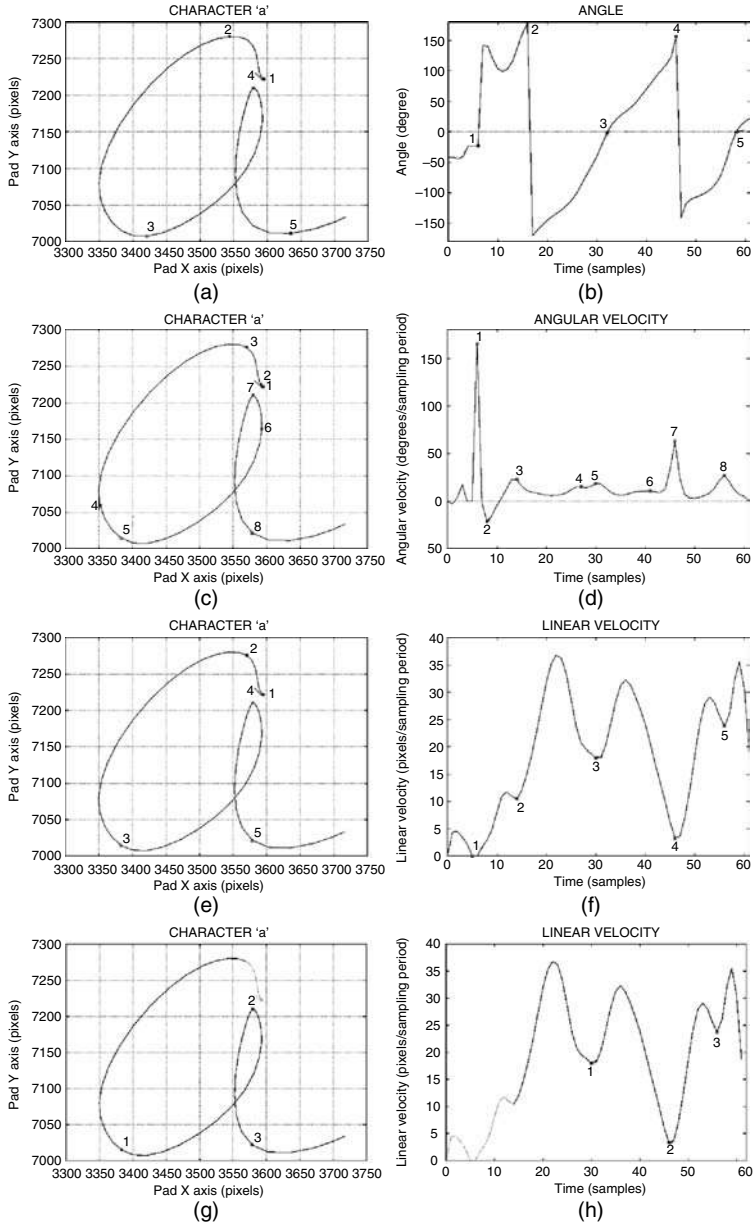



**Figure 5.4** Steps of the segmentation method based on biological models. Character 'a' with segmentation points and the associated signal information is shown. (a,b) Singularities in angle signal are selected as segmentation candidate points; (c,d) extrema in angular velocity are found in the vicinity of candidates; (e,f) minima in linear velocity next to candidates are selected as final segmentation points; (g,h) pen-up information is used to delete some strokes.

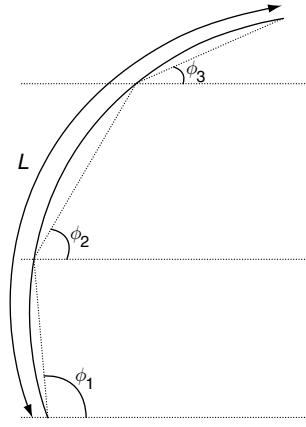



**Figure 5.5** Some geometrical features extracted from a stroke are the sine and cosine of $\phi_1$, $\phi_2$ and $\phi_3$, that code its curvature, and length L, that codes its size.

Then, features should be discriminant, geometrically significant and consistent for each character class. To select a set of them, we considered a large number of candidate features used in the literature [10] or suggested by our knowledge about the problem [9]. The feature selection was handled through the well-known Sequential Forward Selection (SFS) [23], which implicitly assumes that they are used for classification. We used the classifier proposed later in Section 5.1 to assess the quality of these features.

After the selection procedure, each stroke contributed ten features to the feature set of the character, as follows:

- mean $x$-coordinate value and mean $y$-coordinate value, that inform about the position of the stroke within the box containing the character;
- sine and cosine of the three phases ($\phi_1$, $\phi_2$ and $\phi_3$) defined by three segments approaching the stroke, as shown in Figure 5.5, that inform of the curvature of the stroke;
- length of the stroke ($L$), also shown in Figure 5.5, that informs of the size of the stroke;
- a sequential feature showing whether the stroke starts and/or ends a component or is inside the component, which is relevant when characters are made up of several components (e.g. 't' or 'i').

In addition, the character's height to width ratio was added to the feature set. Therefore, a character's feature set has $10n+1$ features, where $n$ is the number of strokes resulting from its segmentation.

It is noteworthy that since these features condense geometrical information about the character, characters can be reconstructed from feature vectors. It is important to have human recognizable prototypes, as will be shown later.

## 4. The Execution of the Prototype Extraction Method

As stated above, character instances are represented using feature vectors, and if these features are discriminant, the vectors will spread along the feature space forming separate clouds. Each cloud is in fact a group of character instances around a prototype. This prototype is *a priori* unknown and can in turn be considered the generating execution prototype in our case of handwriting. The more one character instance resembles the prototype, the smaller the distance between the prototype and the instance feature vector will be.

Thus, the aim of our prototype extraction method is first to find groups of character instances generated by the same prototype, and then to compute the latter. The mean of the feature vectors belonging to the same group is employed to calculate the prototype. This assures the minimization of the distance between the prototype vector and the set of instance vectors employed for its calculation [23].

As described in Section 3.2, feature vectors associated to character instances are built by concatenating the fixed-length feature vectors individually computed for each stroke extracted from the instance. The concatenation is made according to the order they were drawn. The use of this concatenation is an important point in our system because it assures that instance feature vectors store the information about the execution plan that has been followed to write the character; i.e. the order in which the strokes were drawn is always known, thus describing the execution plan. After this stage, prototype extraction proceeds. For this task, the technique employed is based on the idea of performing the grouping in two stages [14].

First, instances are grouped in sets according to classification criteria, i.e. all instances within a group represent the same character concept (they have the same label). The number of created groups depends only on classification needs, i.e. the proximity of instances of different character concepts. Therefore, each of these groups may contain instances generated following one or more different character execution prototypes, since this fact does not affect classification.

A refinement of the grouping is then performed, but based on the search of clouds of instances within the already existing groups, instead of on classification criteria. These two steps are shown in Figure 5.6. Figure 5.7 shows an example of the prototype extraction method performance.

## *4.1 Grouping Training Samples*

A rough division of the feature vectors is performed in the initial grouping stage. This division is made according to classification criteria and tries to gather under the same group sets of prototypes having the same label that may or may not have been drawn using the same prototype. The labels of the training data are thus employed to make this first division, in the *grouping stage*. In the next stage, the prototype extraction method will try to separate the clusters generated by different prototypes within each group.

For the grouping stage we have used Fuzzy ARTMAP, a supervised neural network architecture that follows the principles of Adaptive Resonance Theory (ART) [24]. In this network, two Fuzzy ART unsupervised modules, one grouping feature vectors ($ART_a$) and another classifying the label ($ART_b$), are joined by an inter-ART map. $ART_a$ divides the input space using hyperboxes according to the data label. Fuzzy ARTMAP is a popular neuro-fuzzy scheme that has shown its validity in other handwriting recognition systems [9,15].

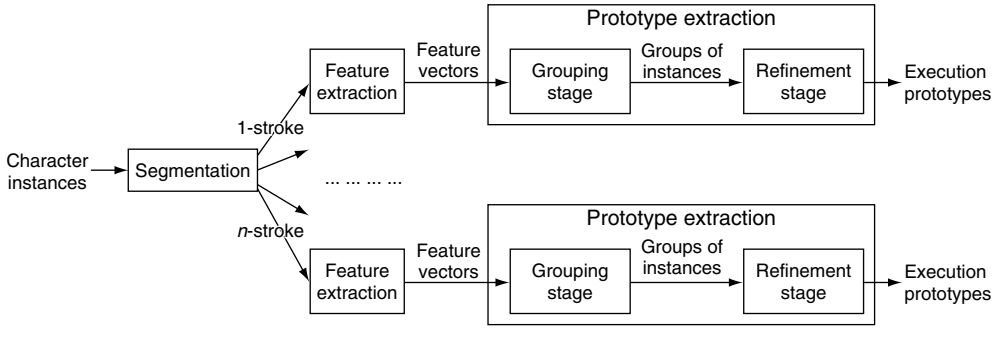



**Figure 5.6** Block diagram of the execution prototype extraction method.

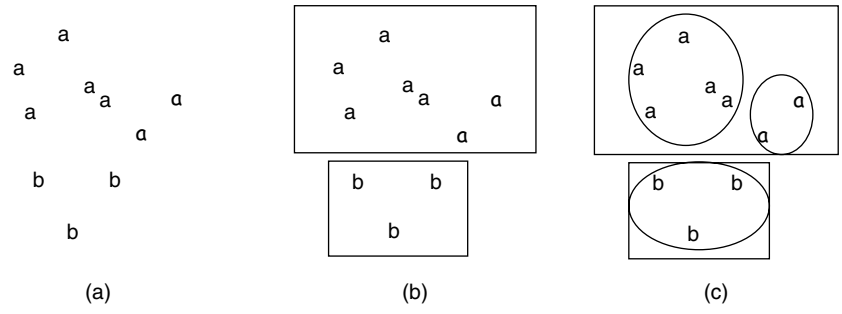



**Figure 5.7** An illustrative example of the prototype extraction method (note that in the real extraction experiments, the dimensionality of the feature space is much higher). The character instances in (a) belonging to two character concepts {a} and {b}, are then clustered according to label and distance criteria in (b), while in the second phase, we extract the prototypes in (c) using the refinement phase.

Since the strokes' number of instances is not fixed, feature vectors are processed by a different neural network depending on the number of strokes of each character (i.e. a different network must be used for 1-stroke characters, 2-stroke characters, etc.) as shown in Figure 5.6. After training, the *n*-stroke network will have created a number of hyperboxes dividing the *n*-stroke input feature space, as can be seen in Figure 5.7. All instances within a given hyperbox have the same label, unless also contained by a smaller hyperbox. Sometimes zero-volume hyperboxes containing just one feature vector are found. These hyperboxes are usually produced by noisy samples, although they may appear because of the Fuzzy ARTMAP training algorithm itself. Zero-volume hyperboxes are rejected prior to finding out the final prototypes in the next stage. The output groups are thus made by the feature vectors activating the same hyperbox. This assures that vectors belonging to the same group have the same label and are close in the input space.

## *4.2 Refinement of the Prototypes*

The previous stage produces groups of instance vectors according to classification criteria. This is performed by a series of Fuzzy ARTMAP neural networks that divide the input space using labeled hyperboxes, thus creating groups of character instances having the same label. However, these groups may enclose instances stemming from different execution prototypes, as shown in Figure 5.7. To correct this, a refinement stage proceeds, redistributing the instances within previously determined hyperboxes and identifying the generating prototypes. This second stage allows generation of a low number of prototypes, easily recognizable by human readers, that represent the knowledge acquired by the neural networks.

This procedure incrementally builds the prototype lexicon from the existing hyperboxes. For each iteration, a number of *permanent prototypes* exist, as well as a *candidate prototype* that can vary until it becomes permanent.

Initially, only one permanent prototype exists: it is calculated as the mean of the vectors grouped in the smallest hyperbox, i.e. this will be the most specific prototype.

Afterwards, the next hyperbox in ascending order of size is selected, so that its patterns can be used to create new prototypes. In order to do this, one vector among them is randomly selected as a candidate prototype. Then, all the patterns closer to this candidate than to any other are said to have been generated by the current candidate prototype, that in turn can be recalculated as the mean of all these vectors. Because of the random choice of the initial candidate, its new value can differ significantly.

Therefore, all patterns in the processed hyperbox are newly assigned to the closest prototype, permanent or candidate, and another more precise value for the candidate prototype is calculated. Eventually, this process stops when the variation between two successive values for the candidate prototype is under a predefined threshold. When this occurs, the prototype becomes permanent and it is added to the prototype lexicon.

However, it may well happen that not all vectors in the processed hyperbox are closer to the newly generated prototype. If some vectors remain, they are closer to another already existing prototype that may have the correct label associated. If, however, they are closer to a prototype with a different class label, it means that a new prototype should be created. This new prototype can be seen as the generator of another 'cloud' of character instances that significantly differ from others describing the same character concept.

To calculate this new prototype, a vector is randomly selected among the remaining patterns of the hyperbox being processed. The prototype is then refined through the same iterative process described above, until it becomes a permanent prototype. The process continues, so that even more than two prototypes could actually be extracted from a given hyperbox.

When all the patterns in a given hyperbox are finally processed, the next largest hyperbox is taken. The procedure continues until all existing hyperboxes have been processed. Therefore, this algorithm starts from very specific prototypes, continuing to more general ones, while also incrementing the generality of those already existing.

In addition, processing the groups of feature vectors proceeds in ascending order of size of their associated hyperboxes, dealing first with compact hyperboxes that contain only one cloud of vectors (i.e. one prototype). Since larger hyperboxes are processed later, they may contain some of the previous hyperboxes, and thus already generated prototypes. The detection of instances belonging to the same group and lying on both sides of a previously computed allograph, lets us determine the existence of different prototypes within the group.

A step-by-step description of this algorithm can be found in Table 5.1.

## *4.3 Experimental Evaluation of the Prototype Extraction Method*

The prototype extraction method proposed can be experimentally evaluated to comply with the objectives of a low number of prototypes, human recognizable prototypes and knowledge condensation. Such evaluation will be performed using the UNIPEN [25] training data sets, using versions 2 and 7 (`train_r01_v02` and `train_r01_v07`). Since the UNIPEN project has already collected over five million characters generated by 40 organizations in a common human-readable format, the use of the character database provided by UNIPEN ensures a large amount of data representing a wide range of allographs, author independence and comparability to other systems.

The extraction method was tested independently on digits, upper-case letters and lower-case letters. The number of labels used in each set was 10, 26 and 26 respectively, according to the English alphabet. Every set was divided into two subsets of the same size, guaranteeing the presence of samples by any writer in both of them. The first subset was used to perform the prototype extraction experiments and later to train the character recognition systems. The second subset was only used for handwriting recognition tests. Table 5.2 shows the distribution of data. Characters were segmented using the method described in Section 3.1 and then a feature vector was collected to represent each character, as explained in Section 3.2.

The number of Fuzzy ARTMAP networks employed in the prototype extraction system was set to six for digit and upper-case letter experiments, and to seven for lower-case letters. Characters having a larger number of strokes were considered as segmentation errors due to the preprocessing stage. However, the number of incorrectly segmented characters was never more than 0.2 % of the total amount of data used for the experiments. The training of all networks was made setting design parameters $\alpha = 0.001$, $\rho = 0$ and $\beta = 1$, so that large (i.e. general) hyperboxes would be created if

**Table 5.1** Summary of the refinement algorithm.

| | |
|---|---|
| ***Notation*** | The hyperboxes created in the grouping refinement stage are denoted $R_1, R_2, \ldots, R_N$, and it is assumed that $\|R_1\| < \|R_2\|, \ldots, < \|R_N\|$.<br>The hyperbox being processed is denoted by $r$.<br>The vectors associated to hyperbox $R_r$ are denoted $\iota_r = \{\boldsymbol{v}_1^r, \boldsymbol{v}_2^r, \ldots, \boldsymbol{v}_{Mr}^r\}$.<br>The collection of prototypes is denoted by $T$. |
| ***Initialization*** | Take $r = 1$ (the smallest hyperbox), compute the first prototype $\boldsymbol{a}_1$ as the mean of the vectors related to $R_1$. Make $T = \{\boldsymbol{a}_1\}$. |
| ***Step 1*** | Select the next category in size, i.e. $r = r + 1$. Mark all vectors in $\iota_r$ as not related to any prototype. |
| ***Step 2*** | Select any random vector not related to any prototype from $\iota_r$, and let it be the candidate prototype $\boldsymbol{a}_c$. |
| ***Step 3*** | Compute the distances from every vector $\boldsymbol{v}_i^r$ to $\boldsymbol{a}_c$ and all other prototypes in $T$.<br>• If vector $\boldsymbol{v}_i^r$ is closer to $\boldsymbol{a}_c$ than to any other prototype, or vector $\boldsymbol{v}_i^r$ is closer to some permanent prototype $\boldsymbol{a}_j$ than to $\boldsymbol{a}_c$ but $\|\|\boldsymbol{a}_c - \boldsymbol{v}_i^r\|\| < \|\|\boldsymbol{a}_c - \boldsymbol{a}_j\|\|$, then mark vector $\boldsymbol{v}_i^r$ as related to prototype $\boldsymbol{a}_c$.<br>• If vector $\boldsymbol{v}_i^r$ is closer to some permanent prototype $\boldsymbol{a}_j$ than to $\boldsymbol{a}_c$, prototype $\boldsymbol{a}_j$ has the same associated label as hyperbox $R_r$ and $\|\|\boldsymbol{a}_c - \boldsymbol{v}_i^r\|\| > \|\|\boldsymbol{a}_c - \boldsymbol{a}_j\|\|$, then mark vector $\boldsymbol{v}_i^r$ as related to prototype $\boldsymbol{a}_j$.<br>• If vector $\boldsymbol{v}_i^r$ is closer to some permanent prototype $\boldsymbol{a}_k$ than to $\boldsymbol{a}_c$, prototype $\boldsymbol{a}_k$ has a different associated label to hyperbox $R_r$ and $\|\|\boldsymbol{a}_c - \boldsymbol{v}_i^r\|\| > \|\|\boldsymbol{a}_c - \boldsymbol{a}_j\|\|$, then do not relate vector $\boldsymbol{v}_i^r$ to any prototype. |
| ***Step 4*** | Compute a new value for prototype $\boldsymbol{a}_c$ as the mean of all vectors in $\iota_r$ associated to it.<br>• If the prototype has changed too much ($\|\|\boldsymbol{a}_c - \boldsymbol{a}_c^{\text{old}}\|\| > \theta$) mark all vectors in $\iota_r$ as not associated to any prototype and go to ***step 3***.<br>• If the prototype has changed very slightly ($\|\|\boldsymbol{a}_c - \boldsymbol{a}_c^{\text{old}}\|\| < \theta$), add $\boldsymbol{a}_c$ to $T$ and remove all vectors associated to $\boldsymbol{a}_c$ from $\iota_r$.<br>  • If $\iota_r$ still contains vectors not related to any prototype, go to ***step 2*** to process them.<br>  • Otherwise, go to ***step 1*** to process the next category. |

possible [26]. Training patterns were presented to the networks until null prediction error on the training data was reached. The threshold parameter of the refinement algorithm was set to 0.00001.

### 4.3.1 An Example of Prototype Extraction

Figure 5.8(a) shows an example of the prototypes extracted for the character concept {Ø} (zero) from the UNIPEN version 2 data set. Prototypes in Figure 5.8(a) are sorted by the number of strokes; the first row contains one-stroke prototypes, the second contains two-stroke prototypes, and so on. The system extracted 16 prototypes within which were the two allographs expected *a priori* (the normal zero (0)

**Table 5.2** Available data from UNIPEN database is distributed in subsets for prototype extraction and learning, and test.

| | Version 2 | | | Version 7 | | |
|---|---|---|---|---|---|---|
| | Digits | Upper-case letters | Lower-case letters | Digits | Upper-case letters | Lower-case letters |
| Prototype extraction/ learning | 1916 | 2109 | 6100 | 7245 | 12105 | 23710 |
| Test | 1917 | 2109 | 6101 | 7242 | 12104 | 23714 |
| Number of labels | 10 | 26 | 26 | 10 | 26 | 26 |

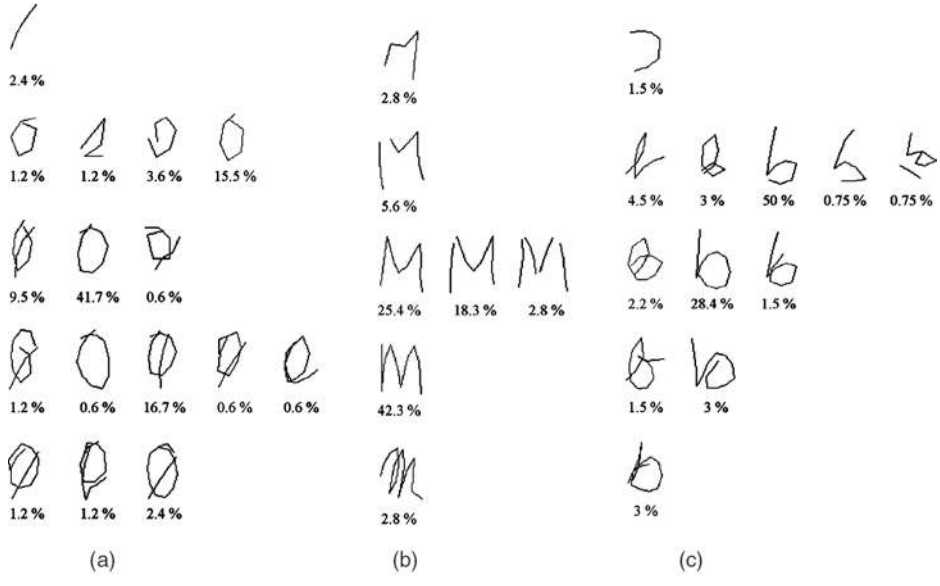



**Figure 5.8** Reconstruction of the prototypes extracted for (a) the '0' digit; (b) the upper-case letter 'M'; and (c) the lower-case letter 'b' of the version 2 UNIPEN data learning set. Prototypes in each row correspond to the same number of strokes. The percentage of training samples represented by each prototype is also shown.

and the slashed zero (Ø)), as well as several different execution plans for both of them. For example, the first and the last two-stroke zeros have been made following the same model (0) but different execution plans: both drawings started upwards but were made clockwise and counterclockwise respectively. This agrees with the studies on handwriting generation in which the possibility of designing different motor plans for the same allograph is stated [1].

Most of the extracted prototypes are easily recognizable at first glance. Exceptionally, the one-stroke prototype is hardly recognizable. This is clearly due to a wrong segmentation performed in the preprocessing stage. Nevertheless, this prototype has been generated from just four character instances, which represents 2.3 % of the zero characters found in the training data, a percentage low enough to

be considered not too significant. In addition, it must be said that the example shown and discussed here is one of the worst cases found among version 2 experimental results on digits.

It can also be seen in Figure 5.8 that the same execution prototype is found with different numbers of strokes (e.g. the fourth two-stroke and the second three-stroke zeros). Therefore, the segmentation method should be revised, since ideally all instances generated from a given prototype should have been segmented into the same number of strokes.

On the other hand, very similar prototypes are sometimes identified having been segmented into an equal number of strokes. However, it is also appreciated in Figure 5.8(a) that these 'repeated' prototypes are generated by a reduced number of instances, while prototypes representing a much larger number of instances (i.e. more general prototypes) can be found (e.g. the first, third and fourth four-stroke slashed zeros represent the same prototype, but they have been generated from 1, 28 and 1 character instances respectively). Figures 5.8(b) and 5.8(c) show the execution prototypes extracted for the upper-case letter 'M' and the lower-case letter 'b', confirming the issues discussed above.

The category proliferation problem present in Fuzzy ARTMAP architecture [26] could be considered as one of the main reasons of the existence of repeated prototypes. This problem can produce the separation of instances generated by the same prototype in many hyperboxes that will turn into the appearance of repeated prototypes produced by few character instances.

The chosen feature set may be responsible for the appearance of repeated prototypes as well. The features used to describe character instances may not include enough information to avoid the superposition in the input space of instances drawn following different prototypes. When this happens, the grouping stage splits prototypes, thus producing repeated models. The improvement of the proposed prototype extraction methods could overcome these difficulties.

Some more reasons can be considered to cause system performance degradation. Erroneously labeled data have been found in the UNIPEN data sets, since not all characters have been checked by human recognizers. Also, ambiguous data that even a human would hardly recognize are quite frequently found. Both factors can produce the appearance of incorrect prototypes, usually generated by just one character instance.

### 4.3.2 Evaluation of the Number of Prototypes

Table 5.3 shows that a reasonable number of prototypes are retrieved in all cases when compared to the number of character instances employed. It can also be observed that the total number of prototypes extracted for version 7 sets is larger than for those of version 2, although the compression rate (number of prototypes per label) has been significantly enhanced. Ideally, the system should retrieve the same number of prototypes, regardless of the number of instance vectors, as long as they are representative of the underlying problem.

**Table 5.3** Number of prototypes extracted from each data set compared to the original number of instances.

| | Version 2 | | | Version 7 | | |
|---|---|---|---|---|---|---|
| | Digits | Upper-case letters | Lower-case letters | Digits | Upper-case letters | Lower-case letters |
| Number of instance vectors | 1916 | 2109 | 6100 | 7245 | 12 105 | 23 710 |
| Number of prototypes | 108 | 186 | 558 | 278 | 723 | 1577 |
| Prototypes per label ratio | 10.8 | 7.2 | 21.5 | 27.8 | 27.8 | 60.7 |

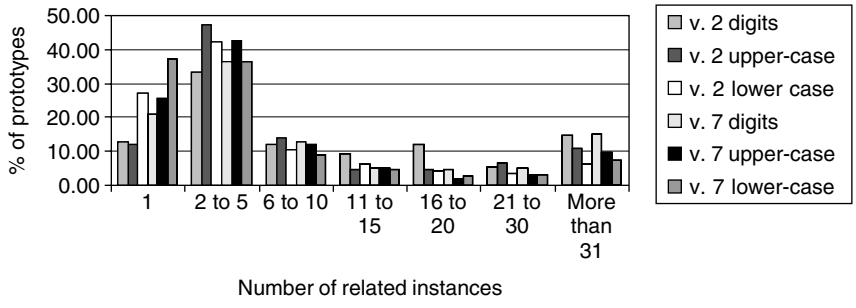



**Figure 5.9** Distribution of the extracted prototypes according to the number of character instances related to them.

The distribution of the extracted prototypes according to the number of character instances employed for their generation can be seen in Figure 5.9. This figure states that significant amounts of the extracted prototypes for every test set were generated by either one character instance or by a group of two to five character instances. The reasons for the appearance of this phenomenon were discussed in the previous section. As has also been mentioned, these are usually repeated prototypes that represent the same allograph and execution plan as other extracted prototypes associated with a much larger number of instances. Later, we will study the possibility of discarding these prototypes according to the number of instances that they represent.

On the contrary, a low number of highly general prototypes have been extracted for each data set, therefore allowing the definition of a prototype dictionary. In the next section, it will be shown that these general prototypes are easily recognizable by humans.

The distribution of the number of prototypes extracted per character concept can also be seen in Figure 5.10. Figure 5.10(a) shows that for most digits and upper-case letters in version 2, the number of prototypes extracted is between one and ten. For lower-case letters, the distribution has moved to the right, showing that 11 to 20 prototypes are needed to represent most character concepts. This result is due to the larger size of the third data set and the greater variability of lower-case characters. The distributions found for version 7 data are shown in Figure 5.10(b). Again, the greater instance variability and the larger amount of data employed turns into an increase of the number of extracted prototypes per character.

### 4.3.3 Evaluation of the Prototype Recognition by Humans

One of the main goals of our system is the extraction of *human recognizable* prototypes. In order to measure this quality, the extracted prototypes for version 2 data sets were used for a human recognition test. For this purpose, three volunteers not related to this research were employed. The reconstructions of the extracted prototypes were shown in a console, one pattern at a time, and the volunteers were asked to label it. The average recognition results are shown in Table 5.4.

It is especially remarkable that the recognition rates for prototypes representing a low number of instances are significantly below the corresponding average recognition rate. This supports the idea that these groups of prototypes are mainly made up of repeated prototypes and exceptions. On the other hand, the more general extracted prototypes show, in almost all cases, much higher recognition rates than the average. This result is also consistent with the ideas exposed previously in Sections 4.3.1 and 4.3.2: the system is able to retrieve a low number of general prototypes that are easily recognizable by humans.

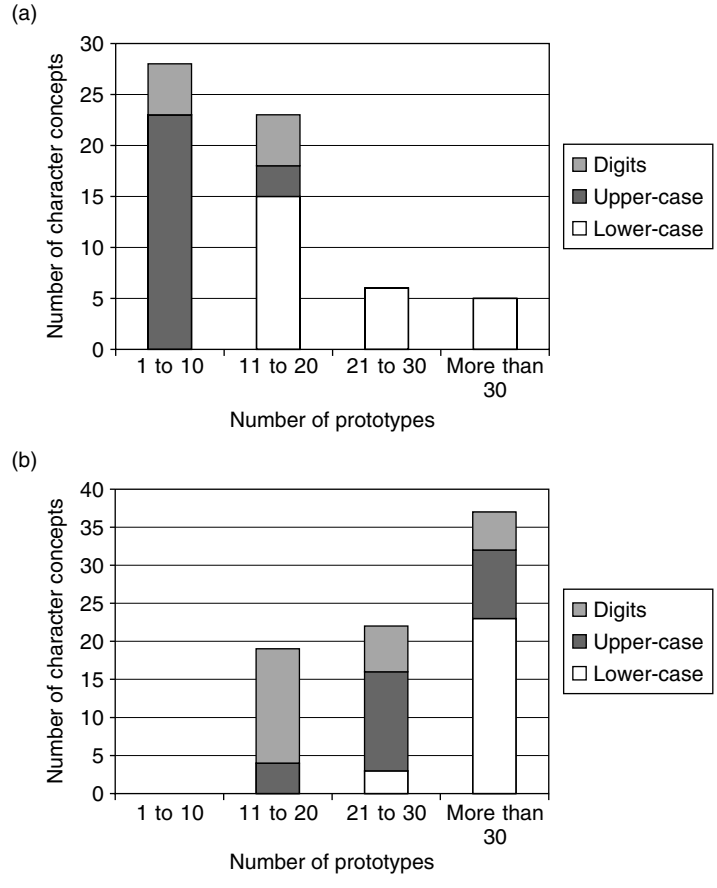



**Figure 5.10** Distribution of the number of prototypes extracted per character concept in (a) UNIPEN version 2; (b) UNIPEN version 7.

**Table 5.4** Recognition rates for the human recognition test of prototypes extracted from UNIPEN v.2 data, according to the number of related character instances. Rates below the average are shown in italics.

| | 1 | 2 ∼ 5 | 6 ∼ 10 | 11 ∼ 15 | 16 ∼ 20 | 21 ∼ 30 | 31+ | Mean |
|---|---|---|---|---|---|---|---|---|
| Digits | *71.43* | *64.81* | 89.75 | *76.67* | *76.92* | 100.00 | 93.75 | **77.47** |
| Upper-case | *57.58* | *72.73* | *67.95* | 100.00 | 100.00 | 91.67 | 95.00 | **76.53** |
| Lower case | *43.49* | *48.02* | 60.45 | *72.38* | 69.56 | 75.44 | 90.48 | **54.12** |

Another interesting point of study may be the comparison of prototype recognition results with the recognition rates achieved by humans when they are shown character instances instead of prototypes. This experiment was made in [15] using the character instances of version 2 UNIPEN digits, upper-case and lower-case letters. In order to make a fair comparison, a *weighted prototype recognition rate* is

**Table 5.5** Human recognition rates on character instances of UNIPEN v.2 data (reported in [15]), and weighted recognition rates using the prototypes extracted in this work.

| | Digits | Upper case | Lower case |
|---|---|---|---|
| Character instances rate | 96.17 | 94.35 | 78.79 |
| Prototype weighted rate | 88.74 | 89.41 | 74.63 |

used, so that correctly or incorrectly identifying a prototype contributes to the final recognition rate proportionally to the number of character instances that the prototype represents. This is computed by the expression:

$$\text{Weighted rate} = \frac{a_1 \cdot p_1 + \cdots + a_i \cdot p_i + \cdots + a_N \cdot p_N}{p_1 + \cdots + p_i + \cdots + p_N} \tag{5.4}$$

where $a_i = 1$ if the $i$th prototype has been correctly identified, and 0 if it has been incorrectly identified; $p_i$ is the number of character instances represented by the $i$th prototype and $N$ is the number of extracted prototypes.

The results of this comparison in Table 5.5 indicate that the recognition rates calculated using Equation (5.4) are close to those achieved on the actual instances. This states again that the extracted prototypes are easily recognizable. Incorrect performance of the prototype extraction system, the use of less visual information describing the prototype than describing character instances (i.e. prototypes are drawn performing a reconstruction from their feature vectors, while character instance shape is drawn according to the information recorded from the digitizer), and errors introduced in the reconstruction phase are considered to be the main factors in prototype rate falls.

## 5. Prototype-based Classification

The method to extract shape and execution prototypes described in Section 4 is intended to condense the knowledge about characters' shape and execution provided by large amounts of data, i.e. the extraction method tries to represent every group of character instances generated following a given allograph and execution plan by one prototype. If this is made correctly, the collection of extracted models will be made up of prototypes that should help to build prototype-based classifiers.

In this section, we will show that our prototype extraction system achieves knowledge condensation by using the extracted prototypes to initialize a handwriting recognizer. If these prototypes are representative of the training data, good recognition rates should be obtained.

### *5.1 The Prototype-based Classifier Architecture*

Since the prototypes, as constructed here, are models close to the group of character instances they are computed from (i.e. training patterns), it seems reasonable to apply recognizers based on the comparison of distances between prototypes and test instances (i.e. test patterns). For this purpose, Learning Vector Quantization (LVQ) codebooks [13] can be used, while other initialization methods can be used for comparison purposes. LVQ is a supervised version of vector quantization and generates code vectors to produce near-optimal decision borders between the classes, even in the sense of classical Bayesian decision theory [27]. The handwriting recognizer thus employed consists of a series of LVQ codebooks. Different codebooks are employed to classify the characters according to their number of strokes.

The prototypes extracted by five different methods were used to initialize the LVQ codebooks: method (1) is the prototype extraction method proposed in this chapter. Methods (2) and (3) are two methods called *propinit* and *eveninit*, proposed in [13] as the standard initialization methods for the LVQ, that choose initial codebook entries randomly from the training data set, making the number of entries allocated to each class proportional (*propinit*) or equal (*eveninit*). Both methods try to assure that the chosen entries lie within the class edges, testing it automatically by *k-NN* classification. Method (4) is *k*-means clustering [23], which is also widely used for LVQ initialization [28,29] and obtains prototypes by clustering the training data of each class (characters having the same label and number of strokes) independently. Finally, method (5) is the centroid hierarchical clustering method [23,30], one of the most popular hierarchical clustering algorithms [30]. This is used in the same way as *k*-means clustering.

The first advantage of the proposed extraction method comes out when setting the different parameters for the comparison experiments: the number of initial entries must be fixed *a priori* for the *propinit, eveninit*, *k*-means and hierarchical initialization methods, while there is not such a need in the extraction algorithm presented in this chapter. Consequently, in order to make comparisons as fair as possible, the number of initial vectors for a given codebook to be generated by the *propinit* and *eveninit* methods was set to the number of prototypes extracted by the algorithm proposed here for the corresponding number of strokes. In addition, the number of prototypes to be computed with *k*-means and hierarchical clustering algorithms was fixed to the number of prototypes extracted by the method proposed here, for the same number of strokes *and* the same label. In all cases, the OLVQ1 algorithm [13] was employed to carry out the training. It must be mentioned that either the basic algorithm LVQ1, the LVQ2.1 or the LVQ3 [31] may be used to refine the codebook vectors trained by the OLVQ1 in an attempt to improve recognition accuracy.

## *5.2 Experimental Evaluation of the Prototype Initialization*

Two different experiments have been made using the five aforementioned initialization methods with the different data sets. First, the system was tested without any kind of training. This would show that indeed the prototypes retain the problem's essential information (i.e. allograph and execution plan variation), which can be used for classification without further refinement. Second, the test was carried out after training the LVQ recognizer. The chosen training lengths were always 40 times the total number of codebook vectors. The initial value of the parameter $\alpha$ was set to 0.3 for all codebook entries. The *k-NN* classifications for *propinit* and *eveninit* initializations were made using $k = 3$. These values correspond to those proposed in the documentation of the LVQ software package [13]. The achieved results are shown in Table 5.6.

The recognition rates yielded in the first experiment show a very poor performance of the *propinit* and *eveninit* initialization methods because, given their random nature, they do not assure the existence of an initial entry in every cloud found in the training data, nor its placement in the middle of the cloud. On the contrary, the other three methods give much better results, especially the *k*-means method, which shows the best rates, followed by our extraction method. Thus supporting the idea that the extracted prototypes retain the problem's essential information.

However, the entries computed by *k*-means and hierarchical clustering methods do not try to represent clusters of instances having the same allograph and execution plan, as the prototypes extracted from Fuzzy ARTMAP boxes do, but just groups of characters with the same label. In addition, the clustering methods tend to create prototypes in strongly represented clusters (i.e. clusters with a large number of instance vectors) and not in poorly represented clusters, while the proposed extraction method is able to compute prototypes for every cluster found in the training data, no matter their number of instances. This idea is also supported by the recognition rates achieved after training the system: once the OLVQ1 algorithm refines the prototypes' positions according to classification criteria, our prototype extraction

**Table 5.6** Results of recognition experiments using different initialization methods, with or without further training of the system. Entries in bold show the highest recognition rates for each experiment.

| Train | Initialization | Version 2 | | | Version 7 | | |
|---|---|---|---|---|---|---|---|
| | | Digits | Upper-case letters | Lower-case letters | Digits | Upper-case letters | Lower-case letters |
| No | *Prototype* | **92.80** | 86.96 | 83.87 | 91.12 | 87.28 | 83.53 |
| No | *Propinit* | 75.85 | 70.65 | 67.30 | 83.97 | 75.78 | 75.50 |
| No | *Eveninit* | 75.74 | 58.04 | 65.25 | 79.51 | 70.86 | 70.63 |
| No | *k-means* | 90.40 | 85.90 | **84.51** | **93.22** | **87.58** | **87.54** |
| No | *Hierarchical* | 90.87 | **88.00** | 79.45 | 89.74 | 82.51 | 77.33 |
| Yes | *Prototype* | **93.84** | **87.81** | **86.76** | **95.04** | **89.68** | **88.28** |
| Yes | *Propinit* | 88.47 | 78.38 | 76.71 | 89.23 | 80.92 | 83.49 |
| Yes | *Eveninit* | 85.08 | 73.11 | 75.40 | 89.42 | 80.02 | 82.28 |
| Yes | *k-means* | 93.32 | 87.58 | 86.34 | 94.61 | 89.05 | 88.24 |
| Yes | *Hierarchical* | 91.91 | 86.86 | 84.43 | 93.17 | 88.15 | 86.84 |

method achieves the best results. This is due to the existence of test instances belonging to unusual prototypes that are captured by the proposed method but not by the others.

Therefore, the proposed extraction method is considered to be the best of the five initialization methods for the following reasons:

1. There is no need to fix *a priori* the number of prototypes to be extracted.
2. The best recognition rates are yielded for all the data sets.

## 5.3 Prototype Pruning to Increase Knowledge Condensation

A new experiment can be made in order to both have a measure of the knowledge condensation performed by the extracted prototypes and to try to decrease their number. This experiment consists of successively removing the prototypes having the smallest number of character instances related to them. This way, we can have an idea of the importance of the prototypes and the knowledge they provide. In this case, the recognition system is initialized using the remaining prototypes and then trained following the criteria mentioned previously. The experiment was made for version 7 lower-case letters, which showed the worst numeric results in prototype extraction and form the most difficult case from the classification point of view.

Removing the prototypes representing ten or less instances strongly reduces the number of models, from 1577 to 297, while the recognition rate decreases from 88.28 % to 81.66 %. This result shows that the number of instances related to a prototype can be taken as a measure of the quantity of knowledge represented by the given allograph. This is consistent with related works for Fuzzy ARTMAP's rule pruning [26]. The new distribution of extracted prototypes per character concept can also be seen in Figure 5.11(b). It is noteworthy that the distribution has significantly moved to the left (see Figure 5.10(a)), while a good recognition rate is preserved.

As a result, we can state that the number of character instances related to a prototype can be used as an index to selectively remove prototypes, thus alleviating the problem of prototype proliferation detected in Sections 4.3.1 and 4.3.2, while increasing the knowledge condensation. In addition, prototypes related to a large number of instances are usually more easily recognized by humans.

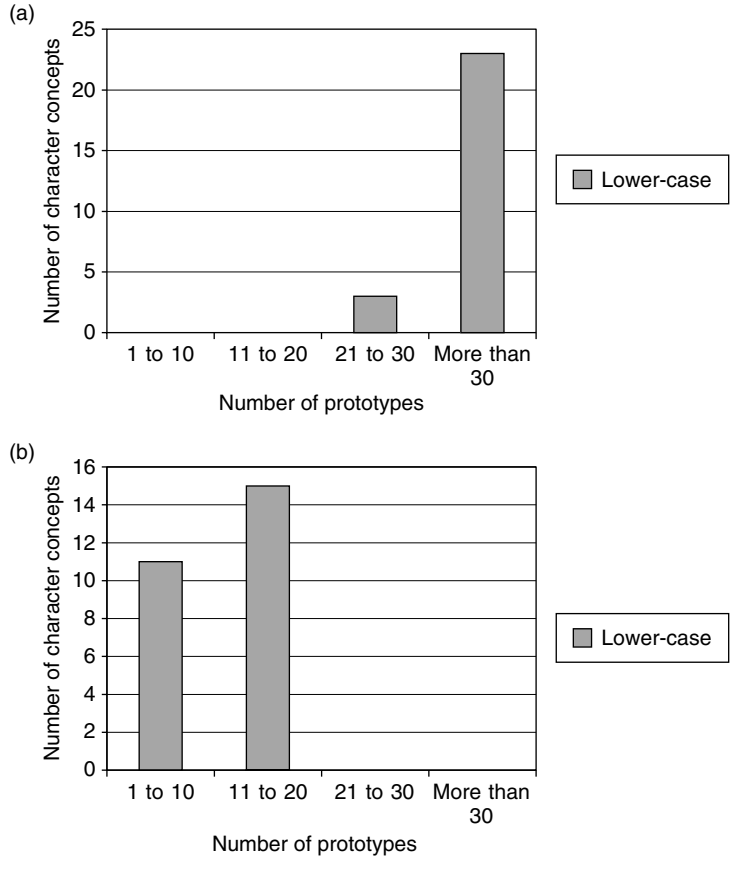



**Figure 5.11** Distribution of the number of extracted prototypes per character concept in UNIPEN version 7 lower-case letters. (a) initial set of prototypes (extracted from Figure 5.9(b)); and (b) removing prototypes representing ten or less instances.

## *5.4 Discussion and Comparison to Related Work*

The comparison of our recognition results with those found in the literature of some other researchers working with the UNIPEN database is not straightforward, due to the use of different experimental conditions. Fair comparisons can only be made when the same release of UNIPEN data is employed; training and test data sets are generated the same way; and data are preprocessed using the same techniques; otherwise, results can only be taken as indicative. This is the case of the results found in [32], in which 96.8 %, 93.6 % and 85.9 % recognition rates are reported for digits, upper-case and lower-case letters respectively for the 6th training release of UNIPEN data after removing those that were found to be mislabeled (4 % of the total); [33] reports 97.0 and 85.6 % for isolated digits and lower-case letters using the 7th UNIPEN release for training and the 2nd development UNIPEN release for test. These numbers confirm the good performance of our recognition system.

The recognition rates of the system proposed here can be fairly compared with those achieved by the neuro-fuzzy classifier studied in [15]. In this chapter, recognition experiments were carried out using similar version 2 UNIPEN data sets. The results achieved are shown in Table 5.7. It can be seen

**Table 5.7** Comparison of the recognition results of the LVQ system initialized using prototypes proposed in this chapter with the two recognizers presented in [15], a Fuzzy-ARTMAP based system, the asymptotic performance of the 1-NN rule and human recognition rates reported in [9]. Entries in bold show the highest recognition rates for each experiment.

| | Version 2 | | | Version 7 | | |
|---|---|---|---|---|---|---|
| | Digits | Upper-case letters | Lower-case letters | Digits | Upper-case letters | Lower-case letters |
| LVQ system | **93.84** | 87.81 | **86.76** | **95.04** | **89.68** | **88.28** |
| System 1 proposed in [15] | 85.39 | 66.67 | 59.57 | — | — | — |
| System 2 proposed in [15] | 82.52 | 76.39 | 58.92 | — | — | — |
| Fuzzy-ARTMAP based system | 93.75 | **89.76** | 83.93 | 92.20 | 85.04 | 82.85 |
| 1-NN rule | 96.04 | 92.13 | 88.48 | 96.52 | 91.11 | — |
| Human recognition | 96.17 | 94.35 | 78.79 | — | — | — |

that the LVQ system based on prototype initialization clearly exceeds all the results achieved by the other system.

In [9] it is also possible to find experiments on handwriting recognition using version 7 UNIPEN digit data sets. The best rate achieved by the two recognition architectures proposed there is 82.36 % of correct predictions. Again, the system presented in this chapter improves this result.

In order to have a more accurate idea of the LVQ system's performance, one more comparison can be made using the same test data with a recognizer based on the already trained Fuzzy ARTMAP networks used for the first grouping stage. The results of both recognizers are also shown in Table 5.7. The LVQ-based system performs better in all cases except for version 2 upper-case letters. As is shown in [34], the high recognition rate achieved by the Fuzzy ARTMAP architecture is due to the appearance of an extraordinarily large number of categories after the training phase.

Considering that the LVQ algorithm performs a 1-*NN* classification during the test phase, it is interesting to notice the asymptotic performance of the 1-*NN* classifier which was proved in [35] to be bounded by twice the Bayesian error rate. In order to approach this asymptotic performance, 1-*NN* classification of the test characters was made using all the training patterns of every set except for version 7 lower-case letters due to the excessive size of this set, which was not affordable in terms of computation. The rates yielded with 1-*NN* classification are also shown in Table 5.7. It is noticeable that the results of our recognition system are quite near to the computed asymptotic performance. This is especially remarkable for version 7 digits and upper-case letters, where the differences are below 1.5 %.

Another reasonable performance limit of our system can be obtained by comparing the achieved recognition rates to those of humans. Thus, the expected average number of unrecognizable data can be estimated for the different test sets. This experiment was carried out in [15] for the version 2 UNIPEN data. The comparison of the LVQ-based system rates and human recognition performance is also shown in Table 5.7. It is quite surprising to notice that the LVQ recognizer performs better than humans do in lower-case recognition. This can be due to different facts. First, humans did not spend too much time on studying the shapes of the training data, although they have a previous knowledge already acquired. In addition, humans get tired after some hours on the computer, and thus their recognizing performance can degrade with time. Finally, humans do not exploit movement information, while the recognizer does, as seen in the prototype samples shown above, which can help to distinguish certain characters.

Finally, it can be said that the main sources of misclassification in the LVQ-based recognizer are common to the prototype extraction method, i.e. erroneously labeled data, ambiguous data, segmentation errors and an insufficient feature set. These problems affect the recognizer in two ways: first, the error sources previously mentioned may cause the appearance of incorrect prototypes that would generate

erroneous codebook entries. In addition, the presentation of erroneous patterns during the training phase may cause a deficient learning. The improvement of these aspects in the prototype extraction method should turn into a decrease of the number of codebook vectors used and an increase in recognition accuracy.

## 6. Conclusions

The prototype-based handwritting recognition system presented in this chapter achieves better recognition rates than those extracted from the literature for similar UNIPEN datasets, showing that the prototypes extracted condense the knowledge existing in the training data, retaining both allograph and execution variation while rejecting instance variability. In addition, it has been shown that the number of character instances that have generated a prototype can be employed as an index of the importance of prototypes that can help to reduce the number of extracted prototypes.

These benefits stem from the method to extract the prototypes: groups of training patterns are identified by the system in two stages. In the first one, Fuzzy ARTMAP neural networks are employed to perform a grouping stage according to classification criteria. In the second, a refinement of previous groups is made, and prototypes are finally extracted. This ensures that prototypes are as general as possible, but that all clouds of input patterns are represented. This way, a low number of easily recognizable prototypes is extracted, making it affordable to build a lexicon, though reducing this number would be a desirable objective. The study of prototype recognition performed by humans stated that the more general prototypes were easy to recognize, while a few repeated prototypes were harder to label.

Besides their importance in initializing the classifier, the prototypes extracted can serve other purposes as well. First, they may help to tackle the study of handwriting styles. In addition, establishing the relationship between character instances, allographs and execution plans may also help to comprehend handwriting generation.

## Acknowledgments

This chapter is a derivative work of an article previously published by the authors in [14]. The authors would like to fully acknowledge Elsevier Science for the use of this material.

## References

[1] Plamondon, R. and Srihari, S. N. "On-line and off-line handwriting recognition: a comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22** (1), pp. 63–84, 2000.

[2] Tappert, C. C., Suen, C. Y. and Wakahara, T. "The state of the art in on-line handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12** (8), pp. 787–808, 1990.

[3] Plamondon, R. "A kinematic theory of rapid human movements part I: movement representation and generation," *Biological Cybernetics*, **72** (4), pp. 295–307, 1995.

[4] Plamondon, R. "A kinematic theory of rapid human movements part II: movement time and control," *Biological Cybernetics*, **72** (4), pp. 309–320, 1995.

[5] Jain, A. K., Duin, R. P. W. and Mao, J. "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22** (1), pp. 4–37, 2000.

[6] Bellagarda, E. J., Nahamoo, D. and Nathan, K. S. "A fast statistical mixture algorithm for on-line handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16** (12), pp. 1227–1233, 1994.

[7] Parizeau, M. and Plamondon, R. "A fuzzy-syntactic approach to allograph modeling for cursive script recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17** (7), pp. 702–712, 1995.

[8] Dimitriadis, Y. A. and López Coronado, J. "Towards an ART- based mathematical editor that uses on-line handwritten symbol recognition," *Pattern Recognition*, **28**(6), pp. 807– 822, 1995.

[9] Gómez-Sánchez, E., Gago González, J.Á., *et al.* "Experimental study of a novel neuro-fuzzy system for on-line handwritten UNIPEN digit recognition," *Pattern Recognition Letters*, **19** (3), pp. 357–364, 1998.

[10] Morasso, P., Barberis, L., *et al.* "Recognition experiments of cursive dynamic handwriting with self-organizing networks," *Pattern Recognition*, **26** (3), pp. 451–460, 1993.
[11] Teulings, H. L. and Schomaker, L. "Unsupervised learning of prototype allographs in cursive script recognition," in Impedovo, S. and Simon, J. C. (Eds) *From Pixels to Features III: Frontiers in Handwriting Recognition*, Elsevier Science Publishers B. V., pp. 61–75, 1992.
[12] Vuurpijl, L. and Schomaker, L. "Two-stage character classification: a combined approach of clustering and support vector classifiers," *Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition*, Amsterdam, pp. 423–432, 2000.
[13] Kohonen, T. Kangas, J. *et al. LVQ-PAK: The Learning Vector Quantization Program Package*, Helsinki University of Technology, Finland, 1995.
[14] Bote-Lorenzo, M. L., Dimitriadis, Y. A. and Gómez- Sánchez, E. "Automatic extraction of human-recognizable shape and execution prototypes of handwritten characters," *Pattern Recognition*, **36** (7), pp. 1605–1617, 2001.
[15] Gómez-Sánchez, E., Dimitriadis, Y. A., *et al.* "On- line character analysis and recognition with fuzzy neural networks," *Intelligent Automation and Soft Computing*, **7** (3), pp. 163–175, 2001.
[16] Duneau, L. and Dorizzi, B. "Incremental building of an allograph lexicon," in Faure, C., Kenss, P. *et al.* (Eds) *Advances in handwriting and drawing: a multidisciplinary approach*, Europia, Paris, France, pp. 39–53, 1994.
[17] Plamondon, R. and Maarse, F. J. "An evaluation of motor models of handwriting," *IEEE Transactions on Systems, Man and Cybernetics*, **19** (5), pp. 1060–1072, 1989.
[18] Wann, J., Wing, A. M. and Sövic, N. *Development of Graphic Skills: Research Perspectives and Educational Implications*, Academic Press, London, UK, 1991.
[19] Simner, M. L. "The grammar of action and children's printing," *Developmental Psychology*, **27** (6), pp. 866–871, 1981.
[20] Teulings, H. L. and Maarse, F. L. "Digital recording and processing on handwritting movements," *Human Movement Science*, **3**, pp. 193–217, 1984.
[21] Kerrick, D. D. and Bovik, A. C. "Microprocessor-based recognition of hand-printed characters from a tablet input," *Pattern Recognition*, **21** (5), pp. 525–537, 1998.
[22] Schomaker, L. "Using stroke- or character-based self-organizing maps in the recognition of on-line, connected cursive script," *Pattern Recognition*, **26** (3), pp. 443–450, 1993.
[23] Devijver, P. A. and Kittler, J. *Pattern Recognition: A Statistical Approach*, Prentice-Hall International, London, UK, 1982.
[24] Carpenter, G. A., Grossberg, S., *et al.* "Fuzzy ARTMAP: a neural network architecture for supervised learning of analog multidimensional maps," *IEEE Transactions on Neural Networks*, **3** (5), pp. 698–713, 1992.
[25] Guyon, I., Schomaker, L., *et al.* "UNIPEN project of on-line data exchange and recognizer benchmarks," *Proceedings of the 12th International Conference on Pattern Recognition*, Jerusalem, Israel, pp. 9–13, 1994.
[26] Carpenter, G. A. and Tan, H. A. "Rule extraction: from neural architecture to symbolic representation," *Connection Science*, **7** (1), pp. 3–27, 1995.
[27] Kohonen, T. "The self-organizing map," *Proceedings of the IEEE*, **78** (9), pp. 1464–1480, 1990.
[28] Huang, Y. S., Chiang, C. C., *et al.* "Prototype optimization for nearest-neighbor classification," *Pattern Recognition*, **35** (6), pp. 1237–1245, 2002.
[29] Liu, C.-L. and Nakagawa, M. "Evaluation of prototype learning algorithms for nearest-neighbor classifier in application to handwritten character recognition," *Pattern Recognition*, **34** (3), pp. 601–615, 2001.
[30] Gordon, A. D. "Hierarchical classification," in Arabie, P., Hubert, P. J. and De Soete, G. (Eds) *Clustering and Classification*, World Scientific, River Edge, NJ, USA, pp. 65–121, 1999.
[31] Kohonen, T., Kangas, J. *et al. SOM-PAK: The Self- Organizing Map Program Package*, Helsinki University of Technology, Finland, 1995.
[32] Hu, J., Lim, S. G. and Brown, M. K. "Writer independent on-line handwriting recognition using an HMM approach," *Pattern Recognition*, **33** (1), pp. 133–147, 2000.
[33] Parizeau, M., Lemieux, A. and Gagné, C. "Character recognition experiments using Unipen data," *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR 2001*, Seattle, USA, pp. 481–485, 2001.
[34] Bote-Lorenzo, M. L. *On-line recognition and allograph extraction of isolated handwritten characters using neural networks*, MSc. thesis, School of Telecommunications Engineering, University of Valladolid, 2001.
[35] Duda, R. O. and Hart, P. E. *Pattern Classification and Scene Analysis*, John Wiley & Sons, Inc., New York, USA, 1973.

# 6

# Logo Detection in Document Images with Complex Backgrounds

**Tuan D. Pham**
**Jinsong Yang**
School of Computing and Information Technology, Nathan Campus, Griffith University
QLD 4111, Australia

*We propose an approach for detecting logos in document images with complex backgrounds. The detection works with documents that contain non-logo images and are subjected to noise, translation, scaling and rotation. The methods are based on the mountain clustering function, geostatistics and neural networks. The proposed logo detection system is tested with many logos embedded in document images, and the results demonstrate the effectiveness of the approach. It is also more favorable when compared with other existing methods for logo detection. The learning algorithm described herewith can be useful for solving general problems in image categorization.*

## 1. Introduction

As a component of a fully automated logo recognition system, the detection of logos contained in document images is carried out first in order to determine the existence of a logo that will then be classified to best match a logo in the database. In comparison with the research areas of logo or trademark retrieval [1–9], and classification [10–15], logo detection has been rarely reported in the literature of document imaging. In the published literature of logo detection, we have found but a single work by Seiden *et al.* [16] who developed a detection system by segmenting the document image into smaller images that consist of several document segments, such as small and large texts, picture and logo. The segmentation proposed in [16] is based on a top-down, hierarchical $X$–$Y$ tree structure [17]. Sixteen statistical features are extracted from these segments and a set of rules is then derived using the ID3 algorithm [18] to classify whether an unknown region is likely to contain a logo or not.

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

This detection algorithm is also independent of any specific logo database, as well as the location of the logo.

A new method is introduced in this chapter to detect the presence of a logo in a document image for which the layout may be a page of fax, a bill, a letter or a form with different types of printed and written texts. The detection is unconstrained and can deal with complex backgrounds on document images. In other words, first, the presence of a logo can be detected under scaling, rotation, translation and noise; secondly, the document may contain non-logo images that make the classification task difficult. To fix ideas, the detection procedure consists of two phases: the initial detection is to identify potential logos that include logo and non-logo images, and if there exist any potential logos; then, the verification of the identity of a logo is carried out by classifying all potential logos based on their image contents against the logo database. The following sections will discuss the implementations of the mountain function for detecting potential logos, and geostatistics for extracting content-based image features that will be used as inputs for neural networks to verify the identities of logos in document images.

## 2. Detection of Potential Logos

The detection is formulated based on the principle that the spatial density of the foreground pixels (we define foreground pixels as the black pixels) within a given windowed image that contains a logo, or an image that is not a logo, is greater than those of other textual regions.

We seek to calculate the density of the foreground pixels in the vicinity of each pixel within a window size by considering each pixel as a potential cluster center of the windowed image and computing its spatial density as follows. First we apply an image segmentation using a threshold algorithm such as Otsu's method [19] that is developed for grayscale images, to binarize the document image into foreground and background pixels.

Let $I$ be a document image of size $M \times N$, and $\omega \subset I$ a window of size $m \times n$, which is chosen to be an approximation of a logo area, $k$ be the location of a pixel in $\omega$, $1 \leq k \leq (m \times n)$, and $p$ the midpoint of $\omega$. Such a function for computing the density of the foreground pixels around a given point $k \in \omega$ is the mountain function $M(p)$ which is defined as [20]:

$$M(p) = \sum_{k \in \omega, p \neq k} \delta(k) \exp[-\alpha D(p,k)] \quad a < x(p) \leq b, \; a < y(p) \leq d \tag{6.1}$$

where $\alpha$ is a positive constant, $D(p,k)$ is a measure of distance between $p$ and the pixel located at $k$, $x(p)$ and $y(p)$ are the horizontal and vertical pixel coordinates of $p$ respectively, $a = \text{round}(m/2)$, where round$(\cdot)$ is a round-off function, $b = M - \text{round}(m/2)$, $c = \text{round}(n/2)$, and $d = N - \text{round}(n/2)$. The function $\delta(k)$ is defined as:

$$\delta(k) = \begin{cases} 1 : f_k = \text{foreground} \\ 0 : f_k = \text{background} \end{cases} \tag{6.2}$$

A typical distance measure expressed in Equation (6.1) is defined by:

$$d(p,k) = [x(p) - x(k)]^2 + [y(p) - y(k)]^2 \tag{6.3}$$

The reason for using the mountain function instead of simply counting the number of foreground pixels in the windowed region is that the foreground pixels of the logo region are more compact than those of non-logo regions. Therefore, using Equation (6.1), a region of pixels which are closely grouped together as a cluster tends to have greater spatial density than that of scattered pixels. For example, the number of foreground pixels of a textual region can be the same or greater than that of

a region having a fine logo; however, using the mountain function, the results can be reversed with respect to the measure of spatial density. We will illustrate this effect in the experimental section by comparing the mountain function defined in Equation (6.1) and the counting of foreground pixels within a window $\omega$, denoted as $C(\omega)$, which is given by:

$$C(\omega) = \sum_{k \in \omega} \delta(k) \tag{6.4}$$

where $\delta(k)$ has been previously defined in Equation (6.2).
Finally, the window $\omega^*$ is detected as the region that contains a logo if:

$$\omega^* = \arg\max_{p} M(p) \geq \delta \tag{6.5}$$

where $\delta$ is a threshold value that can be estimated easily based on the training data for logo and non-logo images.

Furthermore, we can see that the use of the mountain function $M(p)$ is preferred to the pixel-counting function $C(\omega)$ because the former, based on the focal point $p^* \in \omega^*$, can approximately locate the central pixel coordinates of a logo, which is then easily utilized to form a bounding box for clipping the whole detected logo. By using the mountain function to estimate the pixel densities, we can detect potential logos that will then be verified by matching their image content-based features against those of the logo database. This verification task is to be carried out by neural networks, where the image-content-based features are determined by a geostatistical method known as the semivariogram function.

## 3. Verification of Potential Logos

### *3.1 Feature Extraction by Geostatistics*

The theory of geostatistics [21] states that when a variable is distributed in space, it is said to be *regionalized*. A regionalized variable is a function that takes a value at point $p$ of coordinates $(p_x, p_y, p_z)$ in three-dimensional space and consists of two conflicting characteristics in both local erratic and average spatially structured behaviors. The first behavior yields to the concept of a random variable; whereas the second behavior requires a functional representation [22]. In other words, at a local point $p_1$, $F(p_1)$ is a random variable; and for each pair of points separated by a spatial distance $h$, the corresponding random variables $F(p_1)$ and $F(p_1 + h)$ are not independent but related by the spatial structure of the initial regionalized variable.

By the hypothesis of stationarity [22], if the distribution of $F(p)$ has a mathematical expectation for the first-order moment, then this expectation is a function of $p$ and is expressed by:

$$E\{F(p)\} = \mu(p) \tag{6.6}$$

The three second-order moments considered in geostatistics are as follows.

1. The variance of the random variable $F(p)$:

$$\mathrm{Var}\{F(p)\} = E\left\{[F(p) - \mu(p)]^2\right\} \tag{6.7}$$

2. The covariance:

$$C(p_1, p_2) = E\{[F(p_1) - \mu(p_2)][F(p_2) - \mu(p_2)]\} \tag{6.8}$$

3. The variogram function:

$$2\gamma(p_1, p_2) = \text{Var}\{F(p_1) - F(p_2)\} \tag{6.9}$$

which is defined as the variance of the increment $F(p_1) - F(p_2)$. The function $\gamma(p_1, p_2)$ is therefore called the semivariogram.

The random function considered in geostatistics is imposed with the four degrees of stationarity known as *strict stationarity*, *second-order stationarity*, the *intrinsic hypothesis* and *quasi-stationarity*. Strict stationarity requires the spatial law of a random function that is defined as all distribution functions for all possible points in a region of interest, and is invariant under translation. In mathematical terms, any two $k$-component vectorial random variables $\{F(p_1), F(p_2), \ldots, F(p_k)\}$ and $\{F(p_1 + h), F(p_2 + h), \ldots, F(p_k + h)\}$ are identical in the spatial law, whatever the translation $h$.

Second-order stationarity possesses the following properties:

1. The expectation $E\{F(p)\} = \mu(p)$ does not depend on $p$, and is invariant across the region of interest.
2. The covariance depends only on separation distance $h$:

$$C(h) = E\{F(p+h)F(p)\} - \mu^2, \forall p \tag{6.10}$$

where $h$ is a vector of coordinates in one- to three-dimensional space. If the covariance $C(h)$ is stationary, the variance and the variogram are also stationary:

$$\text{Var}\{F(p)\} = E\left\{[F(p) - \mu]^2\right\} = C(0), \forall p \tag{6.11}$$

$$\begin{aligned}
\gamma(h) &= \frac{1}{2}E\left\{[F(p+h) - F(p)]^2\right\} \\
&= \frac{1}{2}E\left\{F(p)^2\right\} + \frac{1}{2}E\left\{F(p)^2\right\} - E\{F(p+h)F(p)\} && (6.12)\\
&= E\left\{F(p)^2\right\} - E\{F(p+h)F(p)\} && (6.13)\\
&= E\left\{F(p)^2\right\} - \mu^2 - [E\{F(p+h)F(p)\} - \mu^2] && (6.14)\\
&= C(0) - C(h) && (6.15)
\end{aligned}$$

The intrinsic hypothesis of a random function $F(p)$ requires that the expected values of the first moment and the variogram are invariant with respect to $p$. That is, the increment $[F(p+h) - F(p)]$ has a finite variance which does not depend on $p$:

$$\begin{aligned}
\text{Var}[F(p+h) - F(p)] &= E\{[F(p+h) - F(p)]^2\} \\
&= \gamma(h), \forall p
\end{aligned} \tag{6.16}$$

Quasi-stationarity is defined as a local stationarity when the maximum distance $|h| = \sqrt{h_x^2 + h_y^2 + h_z^2} \leq b$. This is a case where two random variables $F(p_k)$ and $F(p_k + h)$ cannot be considered as coming from the same homogeneous region if $|h| > b$.

Let $f(p) \in \Re$ be a realization of the random variable or function $F(p)$, and $f(p+h)$ be another realization of $F(p)$, separated by the vector $h$. Based on Equation (6.9), the variability between $f(p)$ and $f(p+h)$ is characterized by the variogram function:

$$2\gamma(p, h) = E\left\{[F(p) - F(p+h)]^2\right\} \tag{6.17}$$

which is a function of both point $p$ and vector $h$, and its estimation requires several realizations of the pair of random variables $[F(p) - F(p+h)]$.

In many applications, only one realization $[f(p), f(p+h)]$ can be available, that is the actual measure of the values at point $p$ and $p+h$. However, based on the intrinsic hypothesis, the variogram $2\gamma(p, h)$ is reduced to the dependency of only the modulus and direction of $h$, and does not depend on the location $p$. The semivariogram $\gamma(h)$ is then constructed using the actual data as follows.

$$\gamma(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} [f(P_i) - f(P_i + h)]^2 \tag{6.18}$$

where $N(h)$ is the number of experimental pairs $[f(P_i) - f(P_i + h)]$ of data separated by $h$. Based on this notion, the function $\gamma(h)$ is said to be the experimental semivariogram.

There are several mathematical versions for modeling the theoretical semivariograms [23–24] that allow the computation of a variogram value for any possible distance $h$. However, in order to fit an experimental variogram into any of the above theoretical variogram models, manual fitting and subjective judgments are usually required to ensure the validity of the model. To overcome this problem for the task of automatic recognition, we have chosen to use neural networks to learn the variogram functions from their experimental data.

### *3.2 Neural Network-based Classifier*

Neural networks have been well known for their capabilities for function approximation and are applied herein for approximating the variogram functions from discrete values of the experimental variograms. The multilayer feed-forward neural network consists of one input layer, one hidden layer and one output layer. The input layer receives ten input nodes which are the first ten values of the experimental semivariograms, $\gamma(h = 1), \gamma(h = 2), \ldots, \gamma(h = 10)$ as defined in Equation (6.18). There are twenty nodes, which are chosen arbitrarily, in the hidden layer. The output layer has two nodes that represent the two strength values in the range [0,1] for logo and non-logo images. Thus, when given a logo image for training, the neural network is to respond with the values of 1 and 0 for the nodes of logo and non-logo images respectively. Likewise, when given a non-logo image for training, the neural network is to produce the values of 0 and 1 for the logo and non-logo output nodes respectively. The logistic sigmoid transfer function is selected because it interprets the network outputs as posterior probabilities that can produce powerful results in terms of discrimination.

## 4. Experimental Results

We extract the semivariogram values of 105 logos obtained from the University of Maryland (UMD) logo database (ftp://ftp.cfar.umd.edu/pub/documents/contrib/databases/UMDlogo database.tar) and 40 non-logo images. Figure 6.1 shows a sample of fifteen logos obtained from the UMD database.

Back propagation is used to train the network that has been described above. The neural network is trained until its sum squared error falls below $10^{-5}$. We then use ten document images scanned from letters, forms and billing statements to embed the same 105 logos and another 40 new non-logo images on various locations of the scanned documents. All potential logos are correctly detected by the mountain function during the initial phase of detection. The potential logos are then cropped out using an average size of the logos in the database and their first ten semivariogram values are computed, to be used as the input values for the neural network-based classification. Some images of potential logos detected and cropped out by the mountain function are shown in Figure 6.2. If the output value of a potential logo is above a threshold $\kappa$, then it is accepted as a logo, otherwise it is rejected. For this experiment, we set $\kappa = 0.8$ and we obtained a total detection rate $= 96\,\%$ and substitution rate $= 0\,\%$.


**Figure 6.1** Logo numbers 2–16 from the UMD database.


**Figure 6.2** Images of detected potential logos.

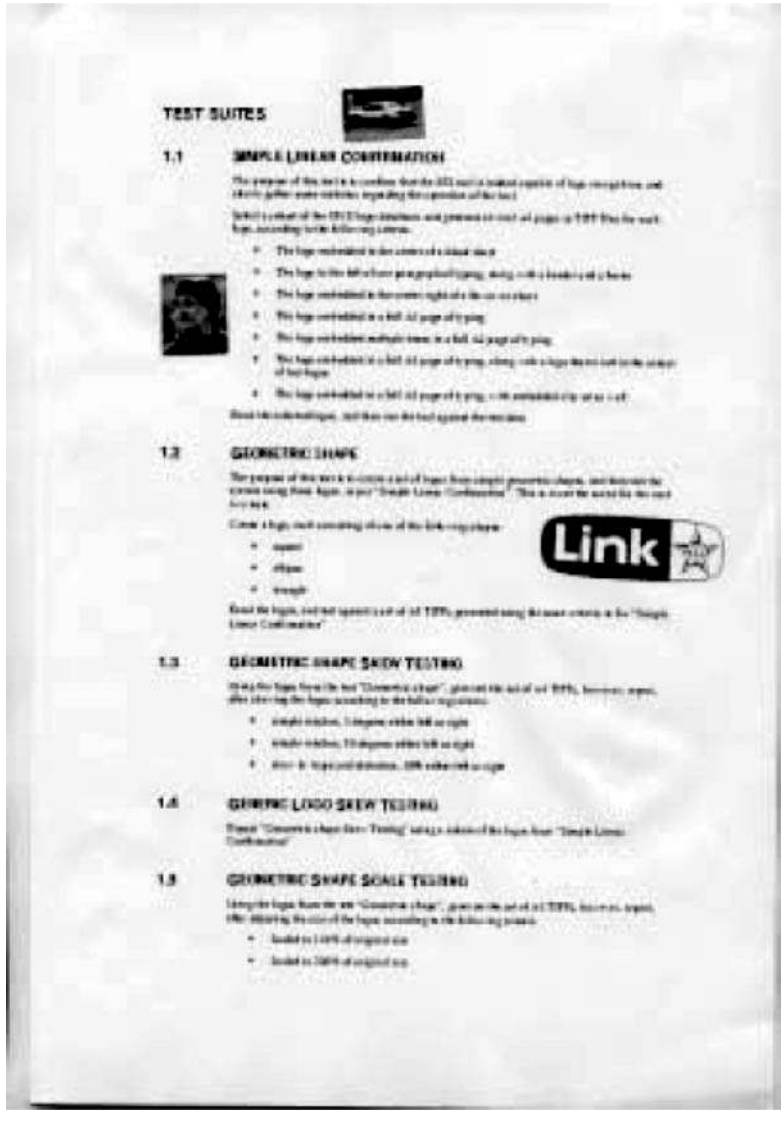



**Figure 6.3** Sample of a document image.

Figure 6.3 shows a sample of the document images, which contains text, photos and logos being placed unconstrainedly within the document space. To test against noise, scaling and rotation, the same testing document images are then degraded with Gaussian noise of zero mean and 0.005 variance, and rotated by five degrees. A sample document is shown in Figure 6.4, that can be considered as practical for real applications. While the mountain function can still detect all the expected potential logos, the verification rate is now reduced by 3 % given the same threshold value.

To study the usefulness of the geostatiscal features extracted from the potential logo images, we train the neural network with other statistical features such as means and variances of the logo and non-logo

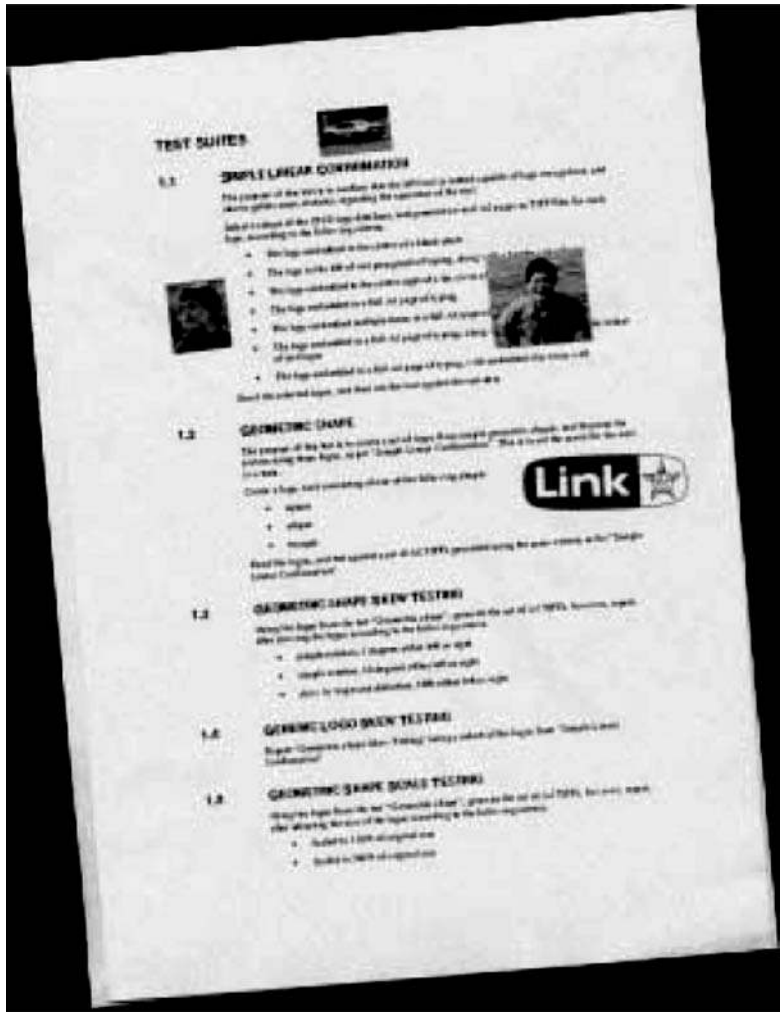



**Figure 6.4** Degraded and rotated document image.

images. The testing results show that the trained neural network cannot classify properly between logo and non-logo images. The reason for this is that there are no obvious differences of means and variances between logos and non-logo images. Furthermore, we use the Higher Order Spectral (HOS) features [25] to extract features from the potential images to train the neural network. These features are obtained by the following steps [26]:

1. Normalizing the image.
2. Applying a smoothing filter to the image.
3. Then, for each angle between 0 and 180 degrees, computing a 1D projection and bispectral features.
4. Concatenating the bispectral features from each angle.

Using the HOS features, the detection rate is 90 %, as compared to the 96 % detection rate using the geostatistical features, but the HOS-based neural network fails to verify all the same rotated logos.

## 5. Conclusions

We have presented an approach for detecting logos that are contained within complex backgrounds of document images. The procedures of this approach start with detecting all potential logos that include logos and images, and then classification is carried out by neural networks to verify the identity of each potential logo against the database. We have also discussed the concept of geostatistics as a useful tool for capturing distinct spatial features of logo images that are used for the learning and classification of neural networks. Many test results have shown the effectiveness of the proposed method, that can also be useful for solving problems in content-based image categorization, particularly for web-based image documents [27]. Applications in this field have become an increasing demand for multimedia industries, such as broadcast news that may contain different categories of image corresponding to news stories, previews, commercial advertisements, icons and logos.

## References

[1] Castelli, V. and Bergman, L. D. (editors), *Image Databases*, John Wiley & Sons, Inc., New York, electronic version, 2002.

[2] Jain, A. K. and Vailaya, A. "Shape-based retrieval: A case study with trademark image databases," *Pattern Recognition*, **31** pp. 1369–1390, 1998.

[3] Rui, Y., Huang, T. S. and Chang, S. "Image retrieval: Current techniques, promising directions, and open issues," *Journal Visual Communication and Image Representation*, **10** pp. 39–62, 1999.

[4] Fuh, C. S., Cho, S. W. and Essig, K. "Hierarchical color image region segmentation for content-based image retrieval system," *IEEE Transactions on Image Processing*, **9**(1), pp. 156–163, 2000.

[5] Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A. and Jain, R. "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(12) pp. 1349–1380, 2000.

[6] Lee, H. K. and Yoo, S. I. "Intelligent image retrieval using neural networks," *IEICE Transactions on Information and Systems*, **E84-D**(12) pp. 1810–1819, 2001.

[7] Chang, M. T. and Chen, S. Y. "Deformed trademark retrieval based on 2D pseudohidden Markov model," *Pattern Recognition*, **34** pp. 953–967, 2001.

[8] Ciocca, G. and Schettini, R. "Content-based similarity retrieval of trademarks using relevance feedback," *Pattern Recognition*, **34** pp. 1639–1655, 2001.

[9] Yoo, H. W., Jung, S. H., Jang, D. S. and Na, Y. K. "Extraction of major object features using VQ clustering for content-based image retrieval," *Pattern Recognition*, **35**, pp. 1115–1126, 2002.

[10] Doermann, D. S., Rivlin, E. and Weiss, I. "Logo recognition using geometric invariants," *International Conference Document Analysis and Recognition*, pp. 894–897, 1993.

[11] Doermann, D. S., Rivlin, E. and Weiss, I. *Logo Recognition*, Technical Report: CSTR-3145, University of Maryland, 1993.

[12] Cortelazzo, G., Mian, G. A., Vezzi, G. and Zamperoni, P. "Trademark shapes description by string-matching techniques," *Pattern Recognition*, **27**(8), pp. 1005–1018, 1994.

[13] Peng, H. L. and Chen, S. Y. "Trademark shape recognition using closed contours," *Pattern Recognition Letters*, **18** pp. 791–803, 1997.

[14] Cesarini, F., Fracesconi, E., Gori, M., Marinai, S., Sheng, J. Q. and Soda, G. "A neural-based architecture for spot-noisy logo recognition," *Proceedings of 4th International Conference on Document Analysis and Recognition*, pp. 175–179, 1997.

[15] Neumann, J., Samet, H. and Soer, A. "Integration of local and global shape analysis for logo classification," *Pattern Recognition Letters*, **23** pp. 1449–1457, 2002.

[16] Seiden, S., Dillencourt, M., Irani, S., Borrey, R. and Murphy, T. "Logo detection in document images," *Proceedings of International Conference on Imaging Science, Systems and Technology*, pp. 446–449, 1997.

[17] Nagy, G. and Seth, S. "Hierarchical representation of optical scanned documents," *Proceedings of the Seventh International Conference on Pattern Recognition*, **1**, pp. 347–349, 1984.

[18] Quinlan, J. R. *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, California, 1992.

[19] Otsu, N. "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, **9**(1) pp. 62–66, 1979.

[20] Yager, R. R. and Filev, D. P. "Approximate clustering via the mountain method," *IEEE Transactions on Systems, Man and Cybernetics*, **24** pp. 1279–1284, 1994.
[21] Matheron, G. *La theorie des variables regionalisees et ses applications*, Cahier du Centre de Morphologie Mathematique de Fontainebleau, Ecole des Mines, Paris, 1970.
[22] Journel, A. G. and Huijbregts, Ch. J. *Mining Geostatistics*. Academic Press, Chicago, 1978.
[23] Isaaks, E. H. and Srivastava, R. M. "Spatial continuity measures for probabilistic and deterministic geostatistics," *Mathematical Geology*, **20**(4) pp. 313–341, 1988.
[24] Isaaks, E. H. and Srivastava, R. M. *An Introduction to Applied Geostatistics*. Oxford University Press, New York, 1989.
[25] Shao, Y. and Celenk, M. "Higher-order spectra (HOS) invariants for shape recognition," *Pattern Recognition*, **34**, pp. 2097–2113, 2001.
[26] Image Recognition Technologies, *Final Report*, Image Research Laboratory, Queensland University of Technology, Brisbane, Australia, February 2002.
[27] Hu, J. and Bagga, A. "Categorizing images in web documents," *Proceedings of SPIE-IS & T Electronic Imaging*, **5010**, pp. 136–143, 2003.

# 7

# An Intelligent Online Signature Verification System

**Bin Li**
Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

**David Zhang**
Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong

*The study of human signatures has a long history, but online signature verification is still an active topic in the field of biometrics. This chapter starts with a detailed survey of recent research progress and commercial products, then proposes a typical online dynamic signature verification system based on time-dependent elastic curve matching. Rather than using special dynamic features such as pen pressure and incline, this system uses the 1D curves of signatures which can be captured using a normal tablet. Static and dynamic features can be well extracted from these two curves about x- and y-coordinates and applied to verification. To improve the performance, we introduce into the system different local weight, personal threshold and auto-update algorithms for reference samples. Finally, we present applications of online signature verification for PDAs and in Internet E-commerce.*

## 1. Introduction

Handwriting is a skill that is personal to individuals. A handwritten signature is commonly used to authenticate the contents of a document or a financial transaction. Along with the development of computer science and technology, automatic signature verification is an active topic in the research and application of biometrics. Technologies and applications of automatic offline (static) and online (dynamic) signature verification are facing many real challenges. For example, there are challenges associated with how to achieve the lowest possible false acceptance and false rejection rate, how to get the best performance as fast and as inexpensively as possible, and how to make applications commercially viable. Nonetheless, it is never long before more new ideas and technologies are employed

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

in this area, or useful applications and ideas are deployed out of this area. Automatic signature verification systems powered by neural networks, parallel processing, distributed computing, network and computer systems and various pattern recognition technologies, are increasingly applicable and acceptable in business areas driven by the growing demands of wired and wireless business. For instance, offline (static) signature verification can be applied in automatic bank processes, document recognition and filing systems, while online (dynamic) signature verification can be applied in automatic personal authentication, computer and network access control, online financial services and in various e-business applications [1,2].

Currently, the identification of a person on the Internet and within an intranet depends on a password, usually stored or communicated as an encrypted combination of ASCII characters. Yet no matter how strong such an encrypted security system is, whether it uses SSL (Secure Socket Layer: 40–100 bit) or the newer SET (Secure Electronic Transaction: 1024 bit) with digital certificates, such conventional password approaches still have fatal shortcomings. For example, passwords are easy to forget, particularly when a user has to remember tens of passwords for different systems, and passwords can be stolen. New verification techniques such as biometrics can be the basis of better authentication systems. Of the many possible biometric schemes, voice is a good candidate, but it depends significantly on an individual's physical condition (e.g. having a cold may degrade verification quality). Use of fingerprints is another good candidate, but fingerprint images can be degraded when an individual perspires or where heavy manual work has damaged the individual's fingers. The eye is yet another strong candidate, but to obtain a good iris image, an individual's eyes must be open and the individual must not be wearing glasses. In addition, to obtain the image, very strong light must be shone onto the retina and the device used to capture an iris or retina image is expensive. Compared with other biometrics, a signature is easily obtained and the devices are relatively cheap. The features of a signature (speed, pen pressure and inclination) are never forgotten and are difficult to steal, meaning that an online human signature verification system has obvious advantages for use in personal identification. Yet, signature verification also has some drawbacks. Instability, emotional and environmental variations may lead to variation in the signature. How to overcome this potential instability and improve the precision of verification is an important topic.

## *1.1 Process and System*

There are two types of signature verification system: online systems and offline systems [3,4]. In offline systems, the signature is written on paper, digitized through an optical scanner or a camera, and verified or recognized by examining the overall or detailed shapes of the signature. In online systems, the signature trace is acquired in real time with a digitizing pen tablet (or an instrumented pen or other touch panel specialized hardware), which captures both the static and dynamic information of the signature during the signing process. Since an online system can utilize not only the shape information of the signature, but also the dynamic time-dependent information, its performance (accuracy) is normally considered to be better than that of an offline system. A typical automatic online signature verification and recognition process was presented by Giuseppe Pirlo in 1993 [5], and is shown in Figure 7.1. Normally, the process of signature verification and recognition consists of a training stage and a testing stage. In the training stage, the system uses the features extracted from one or several training samples to build a reference signature database. These include the stages of data acquisition, preprocessing and feature extraction. During the enrollment stage, signers get their own ID (identification) linked to the signer's reference in the database. In the testing stage, users input their ID and then sign into the input device, either for verification or for recognition. The verification system then uses this ID information to extract the reference from the database, and compares the reference with the features extracted from the input signature. Alternatively, to identify the most similar signatures, the recognition system extracts the features from the input signature and compares them with the features of other signatures in the database, allowing the decision to be made as to whether the test signature is genuine. This chapter will mainly focus on the methods and system design of an online signature verification system.

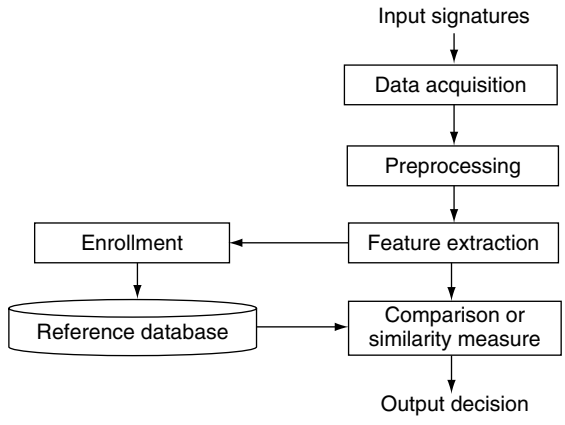



**Figure 7.1** Typical automatic online signature verification process.

## *1.2 The Evaluation of an Online Signature Verification System*

There are many online signature verification algorithms. The performance of such algorithms and systems is largely measured in terms of their error rate.

### 1.2.1 Error Rate

Two types of error rate are commonly used to evaluate a verification and recognition system: the Type I error rate (False Reject Rate or ERR) and the Type II error rate (False Acceptance Rate or FAR). Type II errors imply the recognition of false identifications as positive identifications, e.g. the acceptance of counterfeit signatures as genuine. If we take steps to minimize such false acceptance, however, we will normally increase Type I errors, the rejections of genuine signatures as forgeries [6]. Generally, security will have a higher priority, so, on balance, systems may incline to be more willing to accept Type I errors than Type II, but this will obviously depend on the purpose, design, characteristic and application of the verification system. For example, credit card systems may be willing to tolerate higher Type II error rates rather than risk the possibility of alienating customers whose signatures are frequently rejected. Bank account transactions would not tolerate similar error rates, but would require the lowest level Type II error rate. Figure 7.2 illustrates widely adopted error trade-off curves [7] that

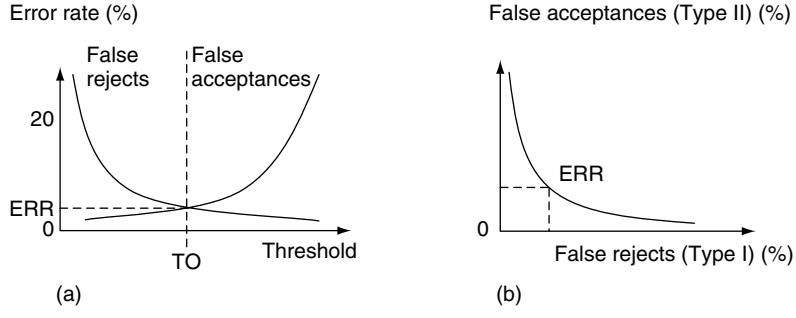



**Figure 7.2** Error rate curves. (a) Error rate vs. threshold; (b) error trade-off curve.

(a) (b) (c)

**Figure 7.3** (a) Genuine signature and (b,c) two forgeries – (b) being a trained forgery and (c) an untrained forgery.

are used to distinguish the relationship between Type I and Type II error rates and their thresholds. The error rate at TO is called the Equal-Error Rate (EER), at which Type I and Type II error rates are the same.

### 1.2.2 Signature Database

To evaluate an online signature verification system, we must build a large signature database of the unique signatures of many individuals. Such a database requires each individual to sign his or her name many times. We call these genuine signatures. Some of these signatures are used as reference samples and others are used as testing samples.

Valid testing also requires that for each signer whose signature is in the database there be several forgeries. The forgeries are of three kinds: random, untrained and trained. Figure 7.3 shows a genuine signature and two forgeries. Random forgeries are very easily obtained, as we simply regard the signatures of other signers in the database as belonging to this class. An untrained forgery is a signature produced by a signer who possesses no information about the genuine signature. Trained forgeries are produced by a professional forger who is in possession of both static and dynamic information about the genuine signature. Obviously, from the point of view of evaluating a system, the trained forgery is the most valuable but it is also very complex to obtain.

So far, there is no public normative signature database, owing to all kinds of reasons.

## 2. Literature Overview

Signatures have been verified online using a wide range of methods. Depending on the signature capture device used, features such as velocity, pen pressure and pen inclination are used, in addition to spatial ($x$, $y$-coordinates) features. Different approaches can be categorized based on the model used for verification. This section introduces several signature verification methods. Since a normative signature database does not exist in the public domain, every research group has collected its own data set. This makes it difficult to compare the different signature verification systems.

### *2.1 Conventional Mathematical Approaches*

Mathematical approaches are still popular in the area of automatic signature verification. The following introduces some of the latest mathematical methods.

Dr Nalwa presented an approach to automatic online signature verification that broke with tradition because it relied primarily on the detailed shape of a signature for its automatic verification, rather than primarily on the pen dynamics during the production of the signature [8]. He challenged the notion that the success of automatic online signature verification hinges on the capture of velocities or

forces during signature production. Nalwa contended that, because of observed inconsistency, it was not possible to depend solely, or even primarily, on pen dynamics and proposed a robust, reliable and elastic local-shape-based model for handwritten online curves. To support his approach, he fleshed out some key concepts, such as the harmonic mean, jitter, aspect normalization, parameterization over normalized length, torque, weighted cross correlation and warping, and subsequently devised the following algorithm components for local and purely shape-based models, and global models based on both shape and time:

- *normalization*, which made the algorithm largely independent of the orientation and aspect of a signature, and made the algorithm inherently independent of the position and size of a signature;
- *description*, which generated the five characteristic functions of the signature;
- *comparison*, which computed a net measure of the errors between the signature characteristics and their prototypes.

Nalwa's model was generated by first parameterizing each online signature curve over its normal arc length. Then, along the length of the curve in a moving coordinate frame, he represented the measures of the curve within a sliding window. The measures of the curve were analogous to the position of the center of mass, the torque exerted by a force, and the moment of inertia of a mass distribution about its center of mass. He also suggested the weighted and biased harmonic mean as a graceful mechanism for combining errors from multiple models, of which at least one, but not necessarily more than one, model is applicable. He recommended that each signature be represented using multiple models, local and global, shape based and dynamics based. Using his shape-based models, it is also possible to apply his approach to offline signature verification. Finally, he outlined a signature verification algorithm that had been implemented and tested successfully, both on databases and in a number of live experiments. Below is a list of the sample size for three different databases that Nalwa used.

- Database 1 (DB1) used a Bell Laboratories in-house developmental LCD writing table with a tethered pen, a total of 904 genuine signatures from 59 signers, and a total of 325 forgeries with an equal-error rate of 3 %.
- Database 2 (DB2) used an NCR 5990 LCD writing table with a tethered pen, a total of 982 genuine signatures from 102 signers, and a total of 401 forgeries with an equal-error rate of 2 %.
- Database 3 (DB3) used an NCR 5990 LCD writing table with a tethered pen, a total of 790 genuine signatures from 43 signers, and a total of 424 forgeries with an equal-error rate of 5 %.

Using the analysis-of-error trade-off curve, the false rejects rate (Type I) versus the false accepts rate (Type II), he obtained an overall equal-error rate that was only about 2.5 %.

One system designed using his approach for automatic on-site signature verification had the principal hardware components of a notebook PC, an electronic writing table, a smart card and a smart card reader. The threshold zero, which distinguished forgeries from genuine signatures, corresponded to 0.50 on the scale in the database experiment, and corresponded roughly to a 0.7 % false rejects rate and a 1 % false accepts rate.

Nelson, Turin and Hastie discussed three methods for online signature verification based on statistical models of features that summarize different aspects of signature shape and the dynamics of signature production [9], and based on the feature statistics of genuine signatures only.

- Using a Euclidean distance error metric and using a procedure for selecting ten out of twenty-two features, their experiments on a database of 919 genuine signatures and 330 forgeries showed a 0.5 % Type I error rate and a 14 % Type II error rate.
- Using statistical properties of forgeries as well as the genuine signatures to develop a quadratic discriminant rule for classifying signatures, the experiments on the same database showed a 0.5 % Type I error rate and 10 % Type II error rate.

In 1997, Ronny Martens and Luc Claesen presented an online signature verification system that identified signatures based on 3D force patterns and pen inclination angles, as recorded during signing [10]. Their feature extraction mechanism used the well-known elastic matching technique but emphasized the importance of the final step in the process: the discrimination based on the extracted features by choosing the right discrimination approach to drastically improve the quality of the entire verification process. To extract a binary decision out of a previously computed feature vector, they used statistical, kernel and Sato's approaches, as well as Mahalanobis distances.

Martens' and Claesen's database consisted of 360 genuine signatures from 18 signers and 615 random forgeries from 41 imitators. Using a kernel function to estimate Gaussian PDFs (Probability Density Functions), they achieved a 0.4 % to 0.3 % equal-error rate. Their techniques, however, were not specific to signature verification, and they should be considered carefully in every process where a classification decision is made using a set of parameters.

## 2.2 Dynamic Programming Approach

Dynamic Time Warping (DTW) is a mathematical optimization technique for solving sequentially structured problems, which has over the years played a major role in providing primary algorithms for automatic signature verification.

This useful method of nonlinear, elastic time alignment still has a high computational complexity due to the repetitive nature of its operations. Bae and Fairhurst proposed a parallel algorithm that used a pipeline paradigm, chosen with the intention of overcoming possible deadlocks in the highly distributed network [11]. The algorithm was implemented on a transputer network on the Meiko Computing Surface using Occam2 and produced a reduction of the time complexity of one order of magnitude.

In 1996, Martens and Claesen discussed an online signature verification system based on Dynamic Time Warping (DTW) [12]. The DTW algorithm originated from the field of speech recognition, and had several times been successfully applied in the area of signature verification, with a few adaptations in order to take the specific characteristics of signature verification into account. One of the most important differences was the availability of a rather large number of reference patterns, making it possible to determine which features of a reference signature were important. This extra amount of information was processed by disconnecting the DTW stage and the feature extraction process. They used a database containing 360 signatures from 18 different persons and used original signatures produced by the other signers as forgeries. The optimal classification was achieved by using Gabor transform-coefficients that described signal contents from 0 Hz to $+/-30$ Hz. As a result, the minimum ERR was 1.4 %.

In their second paper on the use of DTW for signature verification, Martens and Claesen sought to get an alternative DTW approach that was better suited to the signature verification problem [13]. They started by examining the dissimilarities between the characteristics of speech recognition and signature verification and evaluated the algorithm using the same signature database that they used in their 1996 experiment. The optimized EER was about 8 % using the alternative DTW, and about 12 % using the classical DTW. The useful signing information was concentrated in a very small, 20–30 Hz, bandwidth and in their equations, a sample rate faster than 60 Hz was sufficient according to the Nyquist terms.

Paulik, Mohankrishnan and Mikiforuk proposed a time-varying vector autoregressive model for use in signature verification. They treated a signature as a vector random process, the components of which were the $x$ and $y$ Cartesian coordinates and the instantaneous velocity of the recording stylus [14]. This multivariate process was represented by a time-varying $p$th order Vector Autoregressive (VAR) model, which approximates the changes in complex contours typical in signature analysis. The vector structure is used to model the correlation between the signature sequence variables to allow the extraction of superior distinguishing features. The model's matrix coefficients are used to generate the feature vectors that permit the verification of a signer's identity. A database with 100 sample signatures from

16 signers yielded an equal-error rate from 2.87 % to 5.48 % for experiments with different VAR (variance) or 1D (one-dimensional) global or individual thresholds.

Wirtz presented a new technique for use in dynamic signature verification that used a Dynamic Programming (DP) approach for function-based signature verification. Dynamic data, such as pen writing pressure, was treated as a function of positional data, and therefore evaluated locally [15]. Verification was based on strokes as the structural units of the signature. This global knowledge was fed into the verification procedure. The application of a 3D (three-dimensional) nonlinear correlation of the signature signals used the stroke index as the third DP index. In conjunction with the definition of a finite state automaton on the set of reference strokes, the system was correctly able to handle different stroke numbers and missing or additional strokes. The correct alignment of matching strokes and the signature verification process were determined simultaneously. An additional alignment stage before the actual nonlinear correlation was obsolete. Wirtz's experimental database collected 644 genuine signatures and 669 forgeries within two months. The best equal-error rate achieved was 1 % to 1.4 %.

## 2.3 Hidden Markov Model-Based Methods

Due to the importance of the warping problem in signature verification, as well as in handwriting recognition applications, the use of Hidden Markov Models (HMMs) is becoming more and more popular. HMMs are finite stochastic automata and probably represent the most powerful tool for modeling time-varying dynamic patterns. There is a good introduction to the basic principles of HMMs in [16]. There are several papers applying HMMs to handwriting signature verification problems as well.

To represent the signature, L. Yang *et al.* use the absolute angular direction along the trajectory, which is encoded as a sequence of angles [17]. To obtain sequences of the same length, each signature is then quantized into sixteen levels. Another sixteen levels are introduced for pen-up samples. Several hidden Markov model structures were investigated, including left-to-right models and parallel models. The model is trained with the forward–backward algorithm and the probabilities estimated with the Baum–Welch algorithm. In preliminary experiments, the left-to-right model with arbitrary state skips performed the best. Sixteen signatures obtained from thirty-one writers were used for evaluation; eight signatures were used for training and the other eight for testing. No skilled forgeries were available. The experiments showed that increasing the number of states and decreasing the observation length led to a decrease in the false rejects and an increase in false accepts. The best results reported are an FAR of 4.4 % and ERR of 1.75 %.

A method for the automatic verification of online handwritten signatures using both global and local features was described by Kashi, Hu and Nelson in 1997 [18]. These global and local features captured various aspects of signature shape and dynamics of signature production. They demonstrated that with the addition (to the global features) of a local feature based on the signature likelihood obtained from hidden Markov models, the performance of signature verification improved significantly. They also defined a hidden semi-Markov model to represent the handwritten signature more accurately. Their test database consisted of 542 genuine signatures and 325 forgeries. The program had a 2.5 % EER. At the 1 % ERR point, the addition of the local information to the algorithm, which was using only global features, reduced the FAR from 13 % to 5 %.

Dolfing *et al.* addressed the problem of online signature verification based on hidden Markov models in their paper in 1998 [19]. They used a novel type of digitizer tablet and paid special attention to the use of pen pressure and pen tilt. After investigating the verification reliability based on different forgery types, they compared the discriminative value of the different features based on a Linear Discriminant Analysis (LDA) and showed that pen tilt was important. On the basis of 'home- improved', 'over-the-shoulder', and professional forgeries, they showed that the amount of dynamic information available to an imposter was important and that forgeries based on paper copies were easier to detect. In their system, training of the HMM parameters was done using the maximum likelihood criterion

and applying the Viterbi approximation, followed by an LDA. Verification was based on the Viterbi algorithm, which computed the normalized likelihood with respect to the signature writing time. Their database consisted of 1530 genuine signatures, 3000 amateur forgeries written by 51 individuals and 240 professional forgeries. Their results showed an EER between 1 % and 1.9 %.

## 2.4 The Artificial Neural Networks Approach

Along with the vigorous growth of computing science, Artificial Neural Networks (ANNs) have become more and more popular in the area of automatic signature verification. ANNs brought a more computerized and programmable approach to this complex problem.

Lee described three Neural Network (NN) based approaches to online human signature verification: Bayes Multilayer Perceptrons (BMP), Time-Delay Neural Networks (TDNN) and Input-Oriented Neural Networks (IONN). The back perceptron algorithm was used to train the network [20]. In the experiment, a signature was input as a sequence of instantaneous absolute velocities extracted from a pair of spatial coordinate time functions $(x(t), y(t))$. The BMP provides the lowest misclassification error rate among these three types of network. A special database was constructed with 1000 genuine signatures collected from the same subject, and 450 skilled forgeries from 18 trained forgers. The obtained EERs for BMP, TDNN and IONN were 2.67 %, 6.39 % and 3.82 % respectively.

In their paper in 1997, Mohankrishnan, Lee and Paulik examined the incorporation of neural network classification strategies to enhance the performance of an autoregressive model-based signature classification system [21]. They used a multilayer perceptron trained with the back-propagation algorithm for classification. They also presented and compared the results obtained using an extensive database of signatures with those from the use of a conventional maximum likelihood classifier. Using 800 genuine and 800 forged signatures, on the average, the Type I and Type II error rates were about 1.7 % each, while they claimed their identification accuracy was about 97 %.

Matsuura and Sakai presented a stochastic system representation of the handwriting process and its application to online signature verification [22]. Their stochastic system characterizes the motion in writing a signature as a random impulse response. The random impulse response was estimated in terms of the horizontal and vertical components of the handwriting motion, which were considered as the input and output of the system, respectively. They found that, using the random impulse response, it was possible to verify whether a signature was genuine. Their database of 2000 signatures was collected from ten individuals over a six-month period and the EER was claimed to be 5.5 %.

## 2.5 Signature Verification Product Market Survey

As a new market, the automatic signature verification system is not yet popular. However, there are several small to medium companies working on delivering solutions and systems. These products have been adopted mostly in financial, insurance and computer system securities. Among these suppliers, Communication Intelligence Corporation, PenOp Technology and Cyber-SIGN Inc. are known.

Communication Intelligence Corporation (CIC) scientists patented the first mechanism for capturing the biometric qualities of a handwritten signature [23]. CIC's products include 'Signature Capture', 'Verification' and 'Document or Mail Binding'. Signatures are captured along with timing elements (e.g. speed, acceleration) and sequential stroke patterns (whether the 't' was crossed from right to left, and whether the 'i' was dotted at the very end of the signing process). They called these dynamics derived from a person's muscular dexterity 'muscle memory.' Recently, IBM and CIC have announced their plan to add CIC's 'Jot' handwriting recognition and 'WordComplete' shorthand software applications to the IBM 'ThinkPad' and 'WorkPad' hardware [24].

Cyber-SIGN Inc. is a worldwide market and technology leader in the area of biometric signature verification, signature capture and display [25]. Cyber-SIGN analyzes the shape, speed, stroke order, off-tablet motion, pen pressure and timing information captured during the act of signing.

The data-capturing device used is a graphic tablet with a pressure sensitive pen from WACOM [26]. It distributes its system with a software development kit, which allows users to develop their own applications.

The system distributed by DATAVISION uses a signature pad from the same company [27]. The software is integrated with a signature display program. The software is used for account management. Five signatures are used to enroll into the system, from which a template is generated. The template can be updated. The electronic representation of the signature has a size of 108 bytes in addition to an image of the signature that is stored. The software uses both representations for verification. The capabilities of the signature pad used to capture the data are not mentioned.

PenOp Technology was founded in 1990 to be the worldwide leader in electronic signature technology that enables secure e-commerce [28]. PenOp owns a robust and growing portfolio of intellectual property related to electronic signatures and authentication. The PenOp signature software allows signing and authenticating documents online. A digitizing tablet is used to capture a stamp that is based on the captured signature. With a different user verification method (password, etc.), the signature stamp can be affixed to a document with additional information concerning when and where the document was signed. The recipient can extract and verify the signature on the document. Three signatures are used to build a signature template and the template can be updated.

SQN Signature Systems is one of the largest providers of PC-based signature verification systems for banks [28]. SQN customers range from small community banks to large commercial banks. Their signature-related biometric products include 'SQN Safe Deposit Management System', 'SQN VERITAS', 'SQN Signature Sentry' and 'SQN STOP Payment System'.

Gateway File Systems Inc. is a research and development company specializing in application-specific computer Web-based imaging solutions. It provides SignatureTrust™ to, speed up the processing of transactions involving verification of the signing agents authority [29].

The ASV Company provides a banking technology team dedicated to electronic pattern matching solutions. The solutions group focuses on the computerized verification tools that financial institutions require in their signature verification operations, such as eBank™ DISCOVERY for bank check processing [30].

The survey in this section reveals that most of the signature verification and recognition applications are targeted at efficiency improvement in bank processing, or for enhanced security in computer systems. Most of the applications are standalone and even if there is a client–server system, the use of the network is only for the transfer and storage of signatures, and not for real-time signature data acquisition.

## 3. A Typical Online Signature Verification System

In this section, we propose a typical low-cost online signature verification system based on the elastic matching of 1D curves about $x$- and $y$-coordinates, attaching some dynamic features. Just as for a person verifying a signature by eye, different local weights and unfixed thresholds are introduced to improve the performance of the signature verification system.

### *3.1 Data Acquisition*

Differentiating from an offline signature verification system that captures signatures with a scanner or camera, a special pen and tablet is selected as capture device for data acquisition. Four kinds of capture device shown in Figure 7.4 are used in the online system. The former two devices are simpler and cheaper than the others and they can only capture the trajectory of the pen tip with a fixed sampling frequency. The latter is more comfortable to the signer than the former, as the signer can see the trajectory of the pen tip from the LCD when he is signing. The third device is complex and expensive but can collect various kinds of dynamic signature information at high resolution, such as pen-tip

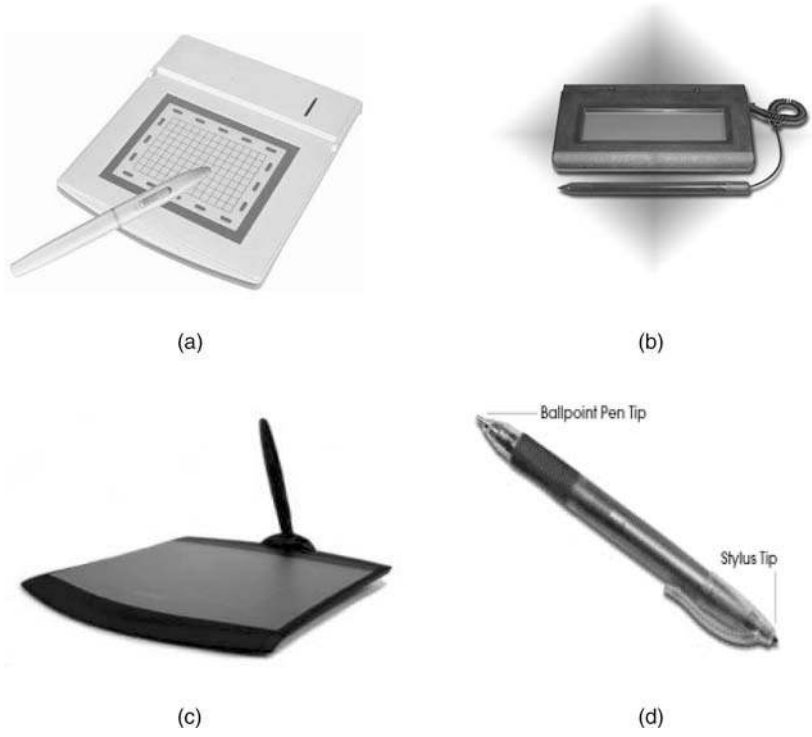



**Figure 7.4** All kinds of capture device. (a) A normal pen and tablet; (b) an LCD Tablet and pen; (c) a tablet that can capture pen pressure and incline; (d) a wireless pen that can write on common paper.

pressure and pen incline. The last device is also expensive and wireless. It makes the signer more comfortable as it can be used to write on paper freely, just like a common pen.

According to the different requirements of applications, we can select different capture devices for data acquisition. For example, in Internet and intranet applications, asking all clients at different locations to use the same special pen and tablet has unavoidable limitations on availability, calibration and cost. Network response is also an important consideration. Therefore, a practical Internet and intranet application requires a simple generic pen and tablet. Furthermore, the velocity and acceleration information can be easily calculated from the pen-tip coordinates and the related time information, see Figure 7.5.

We can divide a signature into two sequences, $(x_i, t_i)$ and $(y_i, t_i)$, corresponding to $x$- and $y$-coordinates (shown in Figure 7.6). These two sequences include both static features and dynamic features.

Different signatures of the same signer may have different size and noise coming from the capture device and hand jitters. So it is necessary to normalize the signature and smooth with a sliding window. In this chapter, the Gauss function is used as this sliding window to smooth the $x$- and $y$-curves of a signature.

$$g(\lambda) = \frac{\exp(-\lambda^2/(2\sigma^2))}{\int_{-L}^{+L} \exp(-\gamma^2/(2\sigma^2))\mathrm{d}\gamma}, -L \le \lambda \le +L \tag{7.1}$$

where $L$ is the width of the sliding window and $\delta \approx L/2$. Curves about the $x$- and $y$-axes after preprocessing are shown in Figure 7.7.

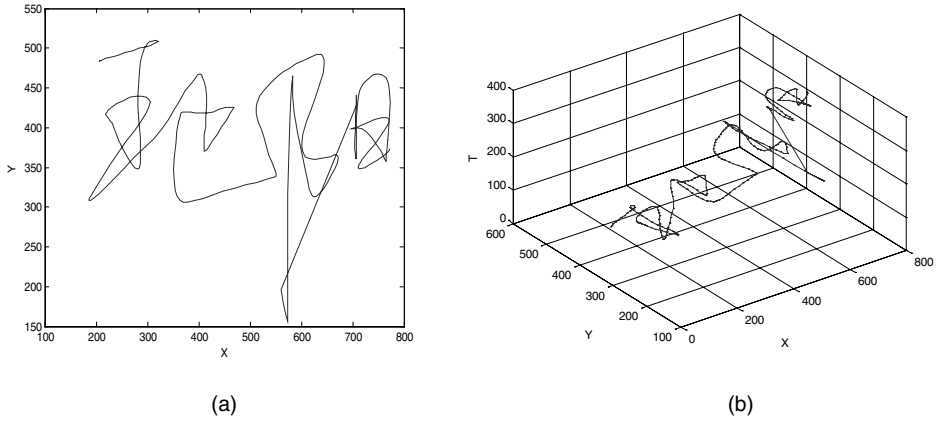



**Figure 7.5** Original signature (a) shown in 2D; (b) shown in 3D and including time information.

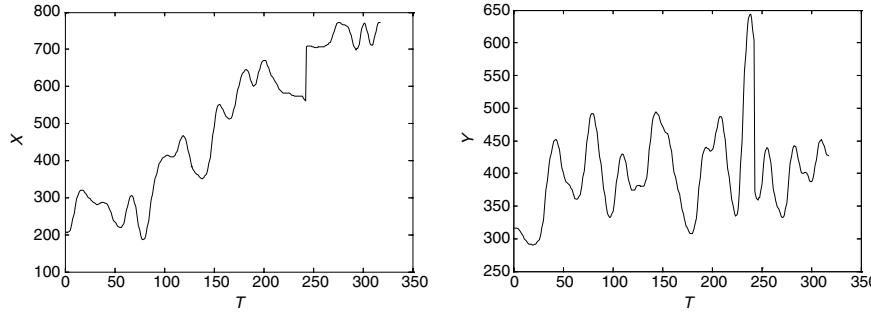



**Figure 7.6** Original curves of a signature about the x-axis and y-axis.

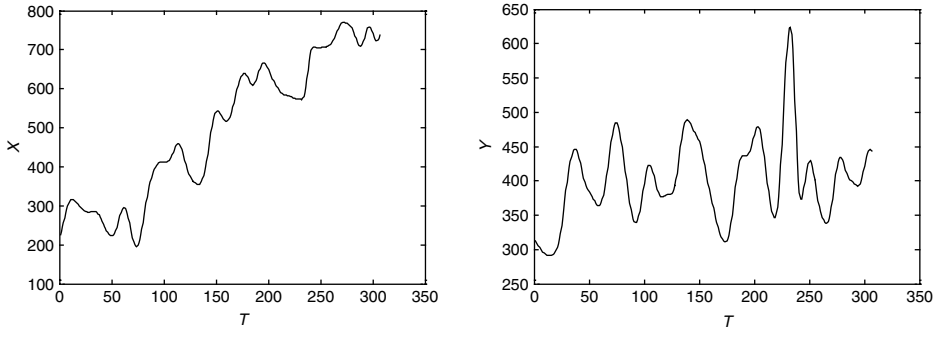



**Figure 7.7** Curves of a signature about the x-axis and y-axis after preprocessing.

## 3.2 Feature Extraction

There is a consensus in the online signature verification research area that feature extraction and classification are the most difficult parts among all the processing steps. The feature extraction process cannot be too complicated and consume too much time because online signature verification in a network is restricted by the response requirement and network bandwidth. Methods of verification can be classified into two sets: those based on parameters and those based on functions. Methods based on parameters always spend much time on the preprocessing and the feature extraction, but methods based on functions have higher performance and less time costs. In this section, a low-cost verification mechanism based on functions is devised.

As we know, if a person wants to imitate others' signatures, usually, he first tries to make the shapes similar, and then imitates the dynamic features. However, the dynamic features are very difficult to imitate. So the verification is classified into two steps: comparing the curves of two signatures about the $x$- and $y$-axis, and then comparing their dynamic features according to the result of the first step.

For the first step, the sequence of inflexions in troughs and crests is detected and marked in each curve. A series of features for a pair of neighboring inflexions to compare two curves is defined as follows:

- The length of these two neighboring inflexions in $t$-coordinates:

$$l^i = t_i - t_{i-1} \tag{7.2}$$

- The obliquity of the line connecting these neighboring inflexions:

$$\theta^i = \arctan\left(\frac{y_i - y_{i-1}}{t_i - t_{i-1}}\right) \tag{7.3}$$

- The position of the inflexion:

$$p^i = \frac{t_i - t_0}{T_s} \tag{7.4}$$

where $(t_i, y_i)$ is the $i$th inflexion, $t_o$ is the $t$-coordinate of the first inflexion and $T_s$ is the length of the whole signature in $t$-coordinates, see Figure 7.8.

Now we can use these respective static features to describe a curve with a sequence of vectors as follows:

$$S = (s_1, s_2, \ldots, s_N), s_i = (l^i, \theta^i, p^i) \tag{7.5}$$

where $N$ is the number of inflexions.

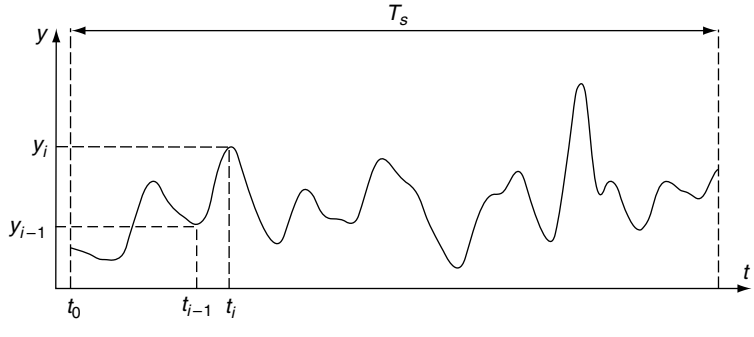



**Figure 7.8** Feature description of a 1D curve.

With this sequence $S$, we can only distinguish genuine signatures from forgeries which are not very similar in shape. But some forgeries produced by a few trained forgers are very similar to genuine signatures in shape. How to distinguish between these forgeries and genuine signatures? Dynamic features can be used to solve this problem. Curves of dynamic features such as velocities and accelerations cannot be used simply as in the above methods, because they are not very similar in terms of the shape of these curves. According to the above inflexions, we can segment a signature into some parts. For each part, the average velocity, the maximum velocity and the deviation of velocity are computed as a vector to describe this signature's dynamic information. For example, these features of a segment of a signature about the $y$-axis are defined as follows:

- The mean velocity between these two neighboring inflexions in $t$-coordinates is:

$$V^i_{\text{mean}} = \frac{\sum_{j=1}^{M} V_i^j}{M}, V_i^j = \frac{y_i^j - y_i^{j-1}}{t_i^j - t_i^{j-1}} \tag{7.6}$$

- Maximum velocity:

$$V^i_{\text{max}} = \max_j(V_i^j) \tag{7.7}$$

- Deviation of velocity:

$$V^i_{\text{d}} = \sqrt{\frac{\sum_{j=1}^{M} (V_i^j - V^j_{\text{mean}})^2}{M}} \tag{7.8}$$

where $i$ is the serial number of a segment and $M$ is the number of sample points of this segment.

We obtained a sequence of dynamic features:

$$W = (w_1, w_2, \ldots, w_N), w_i = (V^i_{\text{mean}}, V^i_{\text{max}}, V^i_{\text{d}}) \tag{7.9}$$

where $N$ is the number of inflexions.

We have a new sequence of vectors to describe a whole signature about the $x$-axis.

$$H = (h_1, h_2 \ldots, h_N), h_i = (l^i_x, \theta^i, p^i, V^i_{\text{mean}}, V^i_{\text{max}}, V^i_{\text{d}}) \tag{7.10}$$

The sequence of vectors about the $x$-axis can be deduced just as for the sequence of vectors about the $y$-axis.

## 3.3 Feature Matching

There are some differences in either the shape or the dynamics of characters between two signatures produced by the same signer. When we compare the curves of two genuine signatures about the $x$- or $y$-axis, the number of their inflexions or the shape of local curves may be different. So the comparison of static features $S = (s_1, \ldots, s_N)$ is implemented with Dynamic Time Warp (DTW), see Figure 7.9. To compare reference sequence $R = (r_1, r_2, \ldots, r_I)$ with test sequence $T = (t_1, t_2, \ldots, t_J)$, a DTW algorithm is introduced as follows:

$$D_{i,j} = \min \begin{pmatrix} D_{i,j} \\ D_{i+1,j} + d_{2,1} \\ D_{i,j+1} + d_{1,2} \end{pmatrix}, 1 \le i \le I, 1 \le j \le J \tag{7.11}$$

where $D_{i,j}$ is the distance between $r_i$ and $t_j$, $d_{1,2}$ and $d_{2,1}$ are punishments.

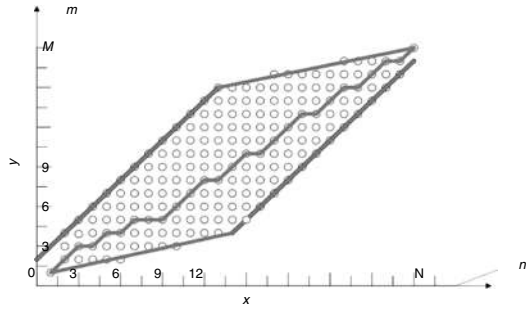



**Figure 7.9** Dynamic time warping.

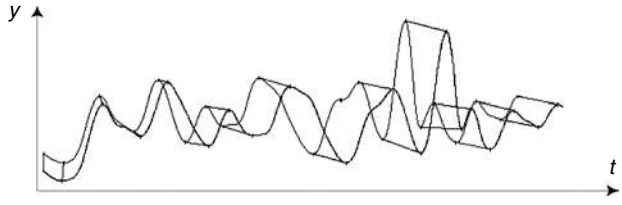



**Figure 7.10** Matching between two genuine signatures.

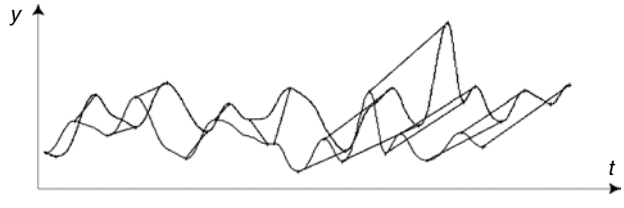



**Figure 7.11** Matching between a genuine signature and a skilled forgery.

For each step in computing the distance, the best matching path is recorded (see Figures 7.10 and 7.11). At the same time, we compute the distance of dynamic features $w_i = (V^i_{\text{mean}}, V^i_{\text{max}}, V^i_{\text{d}})$ between the segment (or the combination of some segments) of the test signature and the corresponding segment (or the combination of some segments) of the reference signature.

## *3.4 Verification*

### 3.4.1 Local Weight and Threshold

Some corresponding segments of two signatures produced by the same signer are very similar and others are not. In other words, some of the distances between each pair of corresponding segments coming from two genuine signatures may be different. Taking all parts of a whole signature as the same weight is unreasonable. So a local weight vector **C** is introduced for each pair of segments when a signature is compared with a reference. This vector has been deduced after a signer enrolled his

signatures. When a signer enrolls his signatures, distance vectors between each signature and others are computed at the same time. The weight vector **C** is deduced and unified by the average reciprocal of these distance vectors.

$$\mathbf{C} = (c_1, c_2, \ldots, c_N), c_i = \frac{\overline{d}_i^{-1}}{\sum_{i=1}^{N} \overline{d}_i^{-1}}, 1 \le i \le N \tag{7.12}$$

where $I$ is the number of segments and $d_i$ is the distance of the $i$th pair of matching segments.

Signatures produced by different signers have different stability. If a fixed threshold is given for all signers in an online signature verification system, its performance will be reduced. So personal thresholds are given for different signers. This personal threshold has been computed according to references after a signer enrolled his signatures. If the signatures of someone are steadier, his threshold is higher, and forging his signature is more difficult. According to the inner class distances and deviation of a signer's enrollments, his personal threshold is given as follows:

$$Th = \alpha_1 \overline{D}_{\text{inner}} + \alpha_2 D_{\text{dev}} \tag{7.13}$$

where $\overline{D}_{\text{inner}}$ is the average of the inner class distances of signer enrollments, and $D_{\text{dev}}$ is the deviation of the signer's enrollments. $\alpha_1$ and $\alpha_2$ are coefficients.

Now, with the personal threshold, we can verify a signature as a genuine signature or a forgery. If the distance of a test signature compared with the reference is lower than the given threshold, the signature is accepted as a genuine signature. Otherwise, it is rejected as a forgery.

### 3.4.2 Auto-update of References

As time passes, the signature of a signer will be changed. The change of signature is gentle and not mutational. So it is very important and necessary that we research the auto-update of references in an online signature verification system.

In this section, a simple strategy of auto-update of references is introduced. At first, we define a coefficient, Reference Value (*RV*), for each reference signature. Then, during each time of signature verification, the coefficient *RV* of each reference will be recomputed and modified if a test signature is accepted as a genuine signature. The algorithm is as follows:

```
if interval > threshold of updating reference interval
  if distance < advance threshold (less than verification threshold)
    (1) Update reference, new signature takes the place of the signature
        for which RV is minimum;
    (2) Recompute verification threshold;
  else
   Continue;
else
  Recompute and update the RV of each reference;
```

## 4. Proposed Online Signature Verification Applications

### *4.1 System Password Authentication*

Online signature verification provides a new and reliable method for system password authentication. Actually, this is not a completely new idea because there have been several proposals and systems in this field. The survey in Section 2.5 presents several applications in this market. However, what is

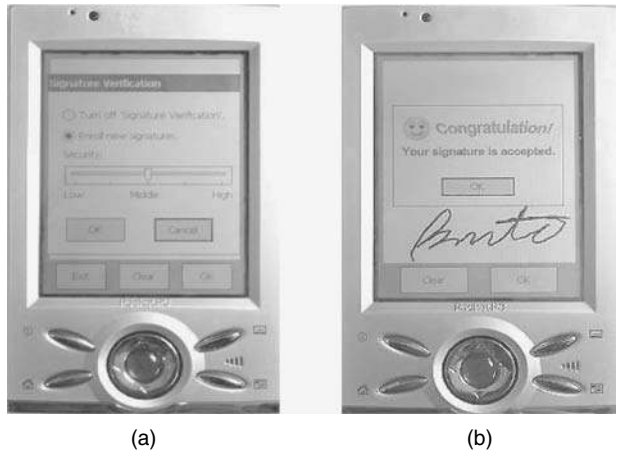
new is that we apply online signature verification to the system authentication of palm devices, such as PDAs. HP Inc. produces a new type of PDA, iPAQ 5450, which has been provided with fingerprint verification by adding a biometric finger scanner [31]. The user can choose to input a password or fingerprint when he wants to enter the system. But its price is also higher because of the new device. To use biometric security on PDAs, most schemes need to add an additional device as with the iPAQ 5450. As we know, a special feature of PDAs is that the user inputs any information with a stylus, and that signature verification schemes only need a stylus to write on the tablet, without any other additional device. So, online signature verification is more suitable to system authentication for PDAs. A demo of an authentication system for PDAs is shown in Figure 7.12.

## *4.2 Internet E-commerce Application*

The fast growing area of e-commence is affected by Internet security issues in personal identification, especially credit card verification, where the number of cases of illegal use of credit card information is rising with the growth of e-commerce. Therefore, a Web-based online signature verification system via the Internet and intranet is proposed in this section to provide a higher level of security in personal identification for e-commerce. As a security enhancement, a combination of traditional character-based password and signature verification can be applied. Public key-based digital signatures are also an important personal information protection method in such network security.

Figures 7.13 and 7.14 show the interpretation of such a system. If we define a trade transaction as a process, the preprocess, such as pretrade registration and agreement, should be performed first. The enrollment and maintenance part includes the following necessary steps:

- The standards and contracts of the signature verification approaches are agreed among all parties – card center, Internet store and cardholder.
- The Internet store enrolls and acquires the access to the bankcard center server.
- The cardholder enrolls in the bankcard center server by training and storing her/his signature information in the server database.
- The cardholder registers herself/himself with the Internet store.
- As an enhancement, the cardholder may require a traditional password as well. After the successful preprocess, the Internet trade transaction can be performed.




(a) (b)

**Figure 7.12** System password authentication for PDAs. (a) Enrollment and set-up; (b) Verification.

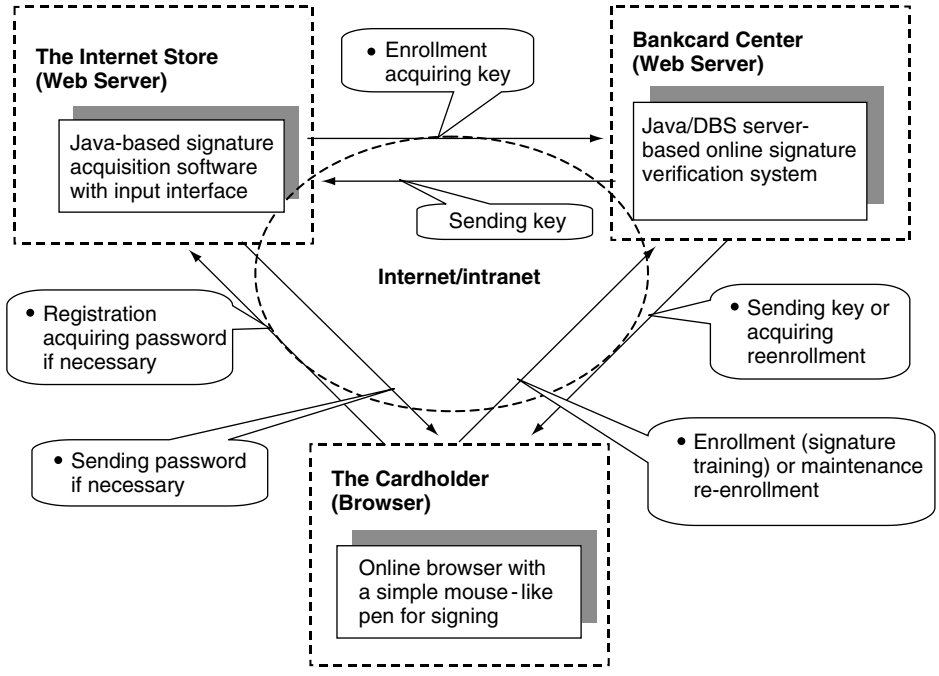



**Figure 7.13** Enrollment and maintenance of an Internet E-commerce application.

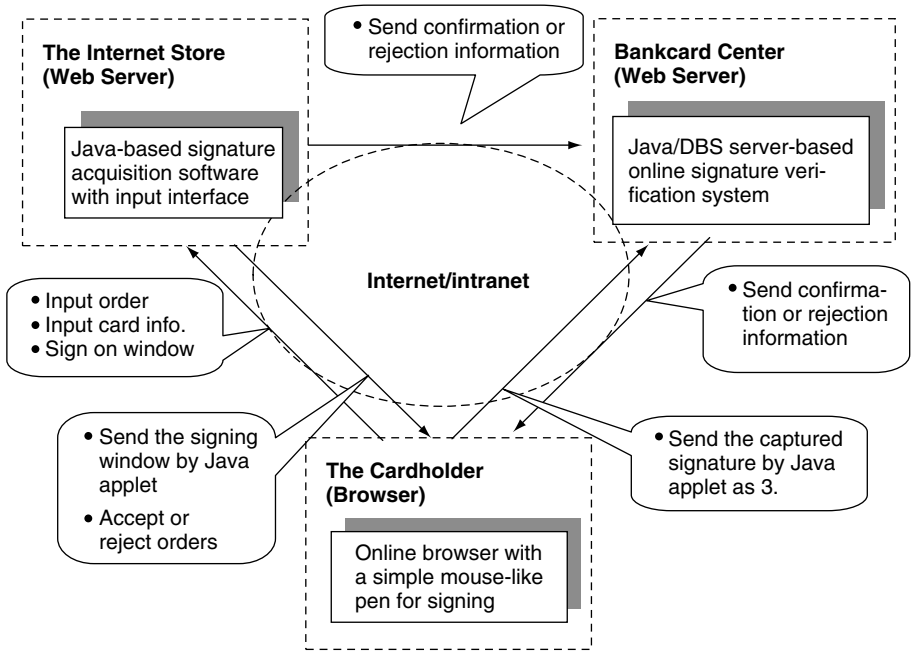



**Figure 7.14** Process of an Internet E-commerce application.

The process part includes the following necessary steps:

- The cardholder inputs an order into the store server.
- The cardholder inputs card information into the store server.
- The store sends the signing window to the cardholder using a Java applet.
- The cardholder signs on the window of his browser.
- The Java applet sends the captured signature to the card center.
- The card center routes confirmation or rejection information to the cardholder.
- The card center routes the confirmation or rejection information to the store as well.
- The store accepts or rejects the order based on the signature verification result.

In the applications market, it is not difficult to find products that are based on automatic signature verification, but most of these are stand-alone systems. In other words, the signer and the verification system execute on the same hardware. With the dramatic growth of the Internet, these applications are not suitable for e-commerce, where a customer and supplier are located in different places.

However, the above system for Internet commerce would be able to deal with geographically separate customers and suppliers. There are a lot of advantages in such a system that uses technology such as Java applets, CGI (Common Gateway Interface), Java Servlets, JDBC (Java Database Connectivity), a DBMS (Database Management System) and the Internet. Different customers can use different input interfaces, such as a pen, mouse or touch screen, for signing, provided they can keep on using this tool after their signature is recorded in the signature server. Internet stores can serve customers worldwide without worrying about the correctness of the credit card information. Such a system can be ported and installed on any Internet server and run on any popular browsers that support Java, such as NS (Netscape) Communicator 4 or MS (Microsoft) Internet Explorer 5.

However, there is still an unresolved issue: security on the Internet. This is a side topic that we believe will be either balanced with Internet business demands or settled in the near future.

## 5. Conclusions

This chapter has introduced recent progress in online signature verification research. A typical low-cost online signature verification system based on time-dependent elastic curve matching has been introduced in detail. Lastly, some application approaches for online signature verification systems have been presented. This chapter foresees the great potential of signature verification in Internet E-commerce and system password authentication of palm devices.

With the vigorous growth of computer science and technology, the application of automatic signature verification will become increasingly accepted in the real world.

## References

[1] Fung, G. S. K., Lau, R. W. H. and Liu, J. N. K. “A Signature Based Password Authentication Method,” *Proceedings of 1997 IEEE International Conference of Systems, Man and Cybernetics*, pp. 631–636, 1997.

[2] Li, B. and Zhang, D. *Signature E-commerce Security System, Biometric Solutions For Authentication in an E-World*, Kluwer Academic Publishers, pp. 187–216, 2002.

[3] Plamondon, R. and Srihari, S. N. “On-line and Off-line Handwriting Recognition: A Comprehensive Survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(1), pp. 63–84, 2000.

[4] Griess, F. D. *On-line Signature Verification*, project report, Michigan State University Department of Computer Science and Engineering, 2000.

[5] Pirlo, G. “Algorithms for Signature Verification, Fundamentals in Handwriting Recognition,” *Proceedings of the NATO Advanced Study Institute on Fundamentals in Handwriting Recognition*, Springer-Verlag, France pp. 435–455, 1993.

[6] Plamondon, R. and Lorette, G. “Automatic Signature Verification and Writer Identification – The State of the Art,” *Pattern Recognition*, **22**(2), pp. 107–131, 1989.

[7] Leclerc, F. and Plamondon, R. "Automatic Signature Verification: The State of the Art – 1989–1993," in *Progress in Automatic Signature Verification*, World Scientific; Singapore pp. 3–20, 1994.
[8] Nalwa, V. S. "Automatic On-line Signature Verification," *Proceedings of the IEEE,* 85(2), pp. 215–239, 1997.
[9] Nelson, W., Turin, W. and Hastie, T. "Statistical Methods for On-line Signature Verification," in *Progress in Automatic Signature Verification*, World Scientific, Singapore, pp. 109–130, 1994.
[10] Martens, R. and Claesen, L. "On-line Signature Verification: Discrimination Emphasised," *Proceedings of ICDAR*, Ulm, IEEE, 1997.
[11] Bae, Y. J. and Fairhurst, M. C. "Parallelism in Dynamic Time Warping for Automatic Signature Verification," *Proceedings of ICDAR*, Ulm, IEEE, 1995.
[12] Martens, R. and Claesen, L. "On-line Signature Verification by Dynamic Time Warping," *Proceedings of the 13th International Conference on Pattern Recognition*, pp. 1015–4651, 1996.
[13] Martens, R. and Claesen, L. "Dynamic Programming Optimization for On-line Signature Verification," *Proceedings of the 4th ICDAR* 1997.
[14] Paulik, M. J., Mohankrishnan, N. and Mikiforuk, M. "A Time Varying Vector Autoregressive Model for Signature Verification," *Proceedings of the IEEE*, 1995.
[15] Wirtz, B. "Stroke-Based Time Warping for Signature Verification," *Proceedings of ICDAR*, Ulm, IEEE, 1995.
[16] Rabiner, L. R. and Juang, B. H. "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, pp. 4–16, 1986.
[17] Yang, L. *et al*. "Application of Hidden Markov Model for Signature Verification," *Pattern Recognition*, **28**(2), pp. 161–170, 1995.
[18] Kashi, R. S. Hu, J. and Nelson, W. L. "On-line Handwritten Signature Verification using Hidden Markov Model Features," *Proceedings of ICDAR*, Ulm, IEEE, 1997.
[19] Dolfing, J. G. A., Aarts, E. H. L., van Oosterhout, J. "On-line Signature Verification with Hidden Markov Models," *Proceedings of the 14th International Conference on Pattern Recognition*, 1998.
[20] Lee, L. L. "Neural Approaches for Human Signature Verification," *Proceedings of ICDAR*, Ulm, IEEE, 1996.
[21] Mohankrishnan, N., Lee, W. S. and Paulik, M. J. "Multi-Layer Neural Network Classification of On-line Signatures," *IEEE 39th Midwest Symposium on Circuits and Systems*, 1996.
[22] Matsuura, T. and Sakai, H. "On Stochastic System Representation of the Handwriting Process and Its Application to Signature Verification," *3rd International Conference on Signal Processing*, 1996.
[23] http://www.cic.com/
[24] http://www.newsfactor.com
[25] http://www.cybersign.com/
[26] http://www.wacom.com/
[27] http://www.datavisionimage.com/
[28] http://www.sqnsigs.com/
[29] http://www.gwfs.bc.ca/
[30] http://www.thediscovery.net/
[31] http://www.hp.com

# 8

# Hybrid Fingerprint Recognition using Minutiae and Shape

**Asker Bazen, Raymond Veldhuis and Sabih Gerez**
University of Twente – EEMCS – SAS,
P.O. Box 217, 7500 AE Enschede, The Netherlands

*This chapter presents a novel minutiae matching method that uses a thin-plate spline model to describe elastic distortions in fingerprints. The thin-plate spline model is estimated using a local and a global matching stage. After registration of the fingerprints according to the estimated model, the number of matching minutiae can be counted using very tight tolerance boxes. For deformed fingerprints, the algorithm gives considerably higher matching scores compared to rigid matching algorithms, while only taking 100 ms on a 1 GHz P-III machine. The additional step of shape matching after elimination of the elastic deformations helps to reduce the error rates further. Furthermore, it is shown that the observed deformations are different from those described by theoretical models proposed in the literature. This chapter is an extension of earlier work that is reported in [1].*

## 1. Introduction

Recognition of people by means of biometric characteristics is an emerging phenomenon in our society. It has received more and more attention during recent years due to the need for security in a large range of applications. Among the many biometric features, the fingerprint is considered one of the most practical ones. Fingerprint recognition requires a minimal effort from the user, does not capture information other than that which is strictly necessary for the recognition process and provides relatively good performance. Another reason for the popularity of fingerprints is the relatively low price of fingerprint sensors, which enables easy integration into PC keyboards, smart cards and wireless hardware.

Eventhough many academic and commercial systems for fingerprint recognition exist, there is a necessity for further research on the topic in order to improve the reliability and performance. Noise in the captured fingerprint image, elastic distortion of the skin when pushing the finger onto the sensor, the partial image of a finger that is obtained, performance problems occurring with recognition in

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

large databases, etc. make it difficult to achieve high performance figures for fingerprint recognition systems. The development of better algorithms can reduce the error rates to levels that are acceptable for a wide range of applications.

The topic of this chapter is fingerprint matching, i.e. the task of comparing a test fingerprint that is actually provided, to a template fingerprint that has been provided earlier, during enrollment. Most fingerprint-matching systems are based on the minutiae, which are the end points and bifurcations of the elevated line structures in the fingerprint, called ridges. A minutiae-based fingerprint-matching system roughly consists of two stages. In the minutiae extraction stage, the minutiae are extracted from the grayscale fingerprint, while in the minutiae matching stage, two sets of minutiae are compared in order to decide whether the fingerprints match. This chapter deals with the compensation of elastic distortions for the sake of improving the performance of minutiae matching.

In minutiae matching, two stages can be distinguished. First, registration aligns both fingerprints as well as possible. Most algorithms use a combination of translation, rotation and scaling for this task. In the rest of this chapter, a transformation that is based on rotation, translation and scaling only will be called rigid. A non-rigid transformation for registration will be called elastic. After registration, the matching score is determined by counting the corresponding minutiae pairs between both fingerprints. Two minutiae correspond if a minutia from the test set is located within a bounding box, or tolerance zone, around a minutia from the template set. The matching score, which is a number in the range from 0 to 1, is calculated as the ratio of the number of matched minutiae and the total number of minutiae.

Unfortunately, there are a lot of complicating factors in minutiae matching. First of all, both sets may suffer from false, missed and displaced minutiae, caused by imperfections in the minutiae extraction stage. Second, the two fingerprints to be compared may originate from a different part of the same finger, which means that both sets overlap only partially. Third, the two prints may be translated, rotated and scaled with respect to each other. The fourth problem is the presence of nonlinear plastic distortions or elastic deformations in the fingerprints, which is generally recognized as the most difficult problem to solve. This problem is addressed in this chapter.

This chapter is organized as follows. First, in Section 2, elastic deformations are discussed and an overview is given of other matching methods that try to deal with them. Next, in Section 3, an elastic minutiae matching algorithm is proposed that estimates a deformation model and uses this model for improved minutiae matching. After the relative distortions have been eliminated, Section 4 presents a method for matching the shape of fingerprints in order to improve the matching performance. Finally, in Section 5, experimental results of the elastic minutiae matching algorithm are given.

## 2. Elastic Deformations

Elastic distortions are caused by the acquisition process itself. During capturing, the 3-dimensional elastic surface of a finger is pressed onto a flat sensor surface. This 3D-to-2D mapping of the finger skin introduces nonlinear distortions, especially when forces are applied that are not orthogonal to the sensor surface. This is a realistic situation, especially when dealing with non-cooperative users that deliberately apply excessive force in order to create intentional elastic deformations. The effect is that the sets of minutiae of two prints of the same finger no longer fit exactly after rigid registration. This is illustrated in Figure 8.1(a), where the ridge skeletons of two prints of the same finger have been registered optimally and displayed in one figure.

In order to tolerate minutiae pairs that are further apart because of plastic distortions, and therefore to decrease the False Rejection Rate (FRR), most algorithms increase the size of the bounding boxes [2]. However, as a side effect, this gives non-matching minutiae pairs a higher probability to get paired, resulting in a higher False Acceptance Rate (FAR). Therefore, changing the size of the bounding box around minutiae only has the effect of exchanging FRR for FAR, while it does not solve the problem of plastic distortions. An alternative approach is to use only local similarity measures, as addressed in

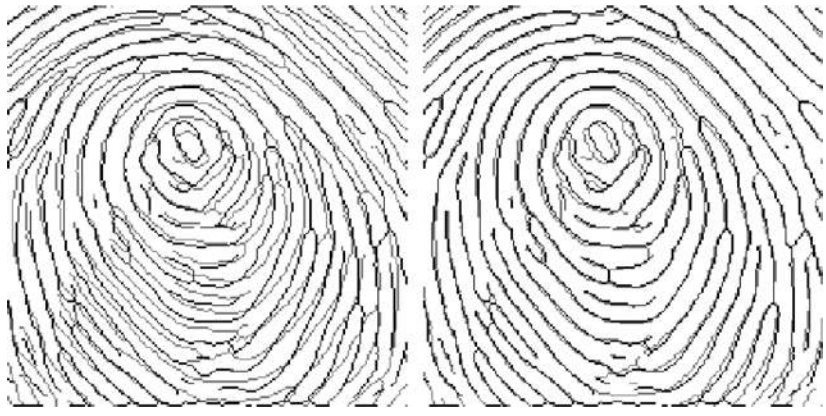



**Figure 8.1** Ridge skeletons of elastically distorted fingerprints that are registered by means of (a) rigid and (b) thin-plate spline algorithms.

Section 3.1, since those are less affected by elastic distortions [3,4]. However, this also decreases the required amount of similarity, and therefore also exchanges FRR for FAR.

Recently, some methods were presented that deal with the problem of matching elastically distorted fingerprints more explicitly, thus avoiding the exchange of error rates. The ideal way to deal with distortions would be to invert the 3D-to-2D mapping and compare the minutiae positions in 3D. Unfortunately, there is no unique way of inverting this mapping. It is therefore reasonable to consider methods that explicitly attempt to model and eliminate the 2D distortion in the fingerprint image. In [5], a method is proposed that first estimates the local ridge frequency in the entire fingerprint and then adapts the extracted minutiae positions in such a way that the ridge distances are normalized all over the image. Although the stricter matching conditions slightly increase the performance of the matching algorithm, this method is not able to solve all possible nonlinear deformations.

Since it is not known in advance whether captured fingerprints contain any distortion, true normalization of the fingerprints to their genuine shape is not possible. The fact that no reference without distortion is available makes normalization in 2D a relative, rather than an absolute, matter. Instead of normalizing each fingerprint on its own, the nonlinear distortions of one fingerprint with respect to the other have to be estimated and eliminated.

In [6], the physical cause of the distortions is modeled by distinguishing three distinct concentric regions in a fingerprint. In the center region, it is assumed that no distortions are present, since this region tightly touches the sensor. The outer, or external, region is not distorted either, since it does not touch the sensor. The outer region may be displaced and rotated with respect to the inner region, due to the application of forces while pressing the finger on the sensor. The region in between is distorted in order to fit both regions to each other, as shown in Figure 8.2. Experiments have shown that this model provides an accurate description of the plastic distortions in some cases. The technique has successfully been applied to the generation of many synthetic fingerprints of the same finger [7]. However, the model has not yet been used in an algorithm for matching fingerprints. Accurate estimation of the distortion parameters is still a topic of research.

Furthermore, a number of nonrigid registration methods have been proposed in other fields than fingerprint matching, see e.g. [8,9]. Most of these methods, however, suffer from two problems. First, they assume that the correspondences between two sets of points are known from the local image structure around these points, which is not a realistic assumption in fingerprint matching, where all

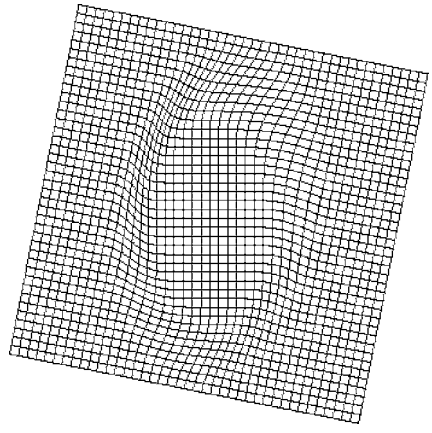


**Figure 8.2** Elastic deformation model of [6].

minutiae points look similar. In addition, they require much processing time, ranging from one to several minutes on a modern PC. In fingerprint matching, the available time is limited to only a few seconds. Therefore, these methods cannot be applied to the fingerprint matching problem.

In this chapter, a minutiae matching algorithm is presented that models elastic distortions by means of a thin-plate spline model, based on the locations and orientations of the extracted minutiae. This model is used to normalize the shape of the test fingerprint with respect to the template. It is, therefore, able to deal with elastically distorted fingerprints. As a consequence, the distances between the corresponding minutiae are much smaller, and tighter bounding boxes can be used in the counting stage. This results in an algorithm that is able to decrease FAR and FRR simultaneously.

## 3. Elastic Minutiae Matching

The elastic minutiae matching algorithm estimates the nonlinear transformation model in two stages. First, the local matching that is presented in Section 3.1 determines which minutiae possibly match, based on local similarity measures. Without this stage, a problem with too many degrees of freedom would have to be solved. Next, the global matching that is presented in Section 3.2 uses the possible correspondences to estimate a global nonrigid transformation. After registration, the corresponding minutiae are counted using bounding boxes that can be chosen rather strictly since the distance between corresponding minutiae after elastic registration is small.

### *3.1 Local Minutiae Matching*

The first step in the proposed matching algorithm is the comparison of local structures. These structures can be compared easily since they contain few minutiae in a small area. In addition, since the structures originate from only a small area in a fingerprint, they are unlikely to be seriously deformed by plastic distortions. The local matching algorithm was inspired by the approach that was described in [10].

Each minutia $m$ in the template and test fingerprints is described by parameters $(x, y, \theta)$, where $(x, y)$ are the pixel coordinates of the minutia and $\theta$ is the orientation of the minutia. The orientation is estimated as one of either opposite orientations of the ridge lines as estimated by the directional field [11]. The correct orientation is selected by tracing all the ridges from the minutia over some distance (one ridge for an end point and three ridges for a bifurcation). For an end point, this directly

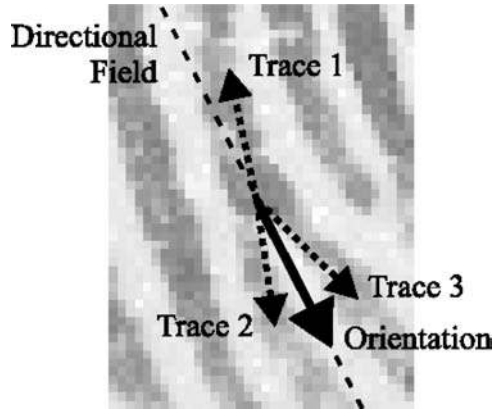



**Figure 8.3** Estimation of the minutiae orientation.

gives the orientation, and for a bifurcation, the orientation that occurs twice is selected. This is illustrated in Figure 8.3. The matching algorithm does not distinguish end points from bifurcations since the type of a minutia can be easily confused due to acquisition noise or pressure differences during acquisition. However, the orientation remains the same when this occurs.

Each minutia defines a number of local structures, which are called minutia neighborhoods. A minutia neighborhood consists of the minutia itself and two neighboring minutiae. When the reference minutia is called $m_0$ and its closest neighbors with increasing distance from $m_0$ are $m_1, m_2, \ldots, m_{n-1}$, with $n$ the number of minutiae, the neighborhoods $\{m_0, m_1, m_2\}$, $\{m_0, m_1, m_3\}$ and $\{m_0, m_2, m_3\}$ are selected for each minutia. Compared with selecting only one neighborhood, this provides more robustness in the local matching stage with respect to false and missing minutiae.

The local matching algorithm compares each minutia neighborhood in the test fingerprint to each minutia neighborhood in the template fingerprint. First, the two neighborhoods are aligned using a least squares algorithm that determines the optimal rotation, translation and scaling. Next, a local matching decision is made by comparing the scaling, the sum of the squared distances between the three corresponding minutiae, and the differences of orientation to a threshold. If the neighborhoods are considered to match, the pair of minutia neighborhoods and the transformation $(t, r, s)$, consisting of translation $t = (t_x, t_y)$, rotation $r$ and scaling $s$, is stored.

After each minutia neighborhood in the test fingerprint has been compared to each minutia neighborhood in the template fingerprint, a list of corresponding minutia neighborhood pairs is obtained. Note that this list does not necessarily contain all true correspondences and that inclusion in this list does not necessarily indicate a true correspondence. However, its size gives a first indication of the degree of similarity of the two fingerprints.

## *3.2 Global Minutiae Matching*

The next step is the determination of the global transformation that optimally registers the two fingerprints. This is called the global matching stage. From the list of local correspondences, the global transformation is determined that is consistent with the largest number of matching minutia neighborhood pairs. It is expected that this transformation explicitly selects the largest number of matching minutiae pairs from the entire minutiae sets.

Several strategies to determine the optimal global transformation from the list of local transformations exist. These methods mainly differ in handling the difficulty of false and contradictory local matches.

In [10], the transformation of the single minutia neighborhood pair that matches best is taken. However, using more information of other local matches certainly improves the accuracy of the registration. Another possibility is to quantize the local registration parameters into bins and construct an accumulator array that counts all occurrences of each quantized registration. Next, the bin that occurs most is selected, and the average of all registrations in that bin is taken. This strategy roughly corresponds to the work in [12], although that method is not based on matching local minutia neighborhoods.

The method that is proposed here achieves further improvement by determining the optimal registration parameters from the positions of the matching minutia neighborhoods, instead of averaging the individual registration parameters. First, the largest cluster of pairs that share approximately the same registration parameters is selected. This is achieved by determining for each matching pair the number of pairs for which the registration parameters differ less than a certain threshold. Next, the transformation $(t, r, s)$ that optimally registers the selected minutiae in the test set to the corresponding minutiae in the template set is calculated in a least squares sense.

However, when applying this registration, elastically deformed fingerprints will not be registered well, as shown in Figure 8.1, simply because an accurate registration $(t, r, s)$ does not exist. Most matching algorithms compensate for this in the counting stage. It has been reported in [2] that for 97.5 % of the minutiae to match, a threshold on the Euclidean distance of two minutiae of $r_0 = 15$ pixels has to be used in 500 dpi fingerprints. As a consequence, minutiae in a rather large part of the image (25 % of the image for 30 minutiae in a $300 \times 300$ image) are considered to match, even when they actually do not match.

In order to allow stricter matching, i.e. a smaller value of $r_0$, elastic registration has to be used to compensate for plastic distortions. A transformation that is able to represent elastic deformations is the Thin-Plate Spline (TPS) model [13]. To our knowledge, it has not been applied earlier to fingerprint recognition. The TPS model describes the transformed coordinates $(x', y')$ both independently as a function of the original coordinates $(x, y)$:

$$x' = f_x(x, y) \tag{8.1}$$

$$y' = f_y(x, y) \tag{8.2}$$

Given the displacements of a number of landmark points, the TPS model interpolates those points, while maintaining maximal smoothness. The smoothness is represented by the bending energy of a thin metal plate. At each landmark point $(x, y)$, the displacement is represented by an additional $z$-coordinate, and for each point, the thin metal plate is fixed at position $(x, y, z)$. The bending energy is given by the integral of the second order partial derivatives over the entire surface and can be minimized by solving a set of linear equations. Therefore, the TPS parameters can be found very efficiently. The TPS model for one of the transformed coordinates is given by parameter vectors $\mathbf{a}$ and $\mathbf{w}$:

$$f(x, y) = a_1 + a_2 x + a_3 y + \sum_{i=1}^{n} w_i U(|P_i - (x, y)|) \tag{8.3}$$

where $U(r) = r^2 \log r$ is the basis function, $\mathbf{a}$ defines the affine part of the transformation, $\mathbf{w}$ gives an additional nonlinear deformation, $P_i$ are the landmarks that the TPS interpolates, and $n$ is the number of landmarks.

The TPS parameters that minimize the bending energy can be found by solving a set of linear equations:

$$\mathbf{K}\mathbf{w} + \mathbf{P}\mathbf{a} = \mathbf{v} \tag{8.4}$$

$$\mathbf{P}^{\mathrm{T}}\mathbf{w} = 0 \tag{8.5}$$

where

$$\mathbf{w} = \left[w_x(1)\ w_x(2) \cdots w_x(n)\right]^{\mathrm{T}} \tag{8.6}$$

$$\mathbf{v} = \left[q_x(1)\ q_x(2) \cdots q_x(n)\right]^{\mathrm{T}} \tag{8.7}$$

$$\mathbf{a} = \left[a_x(1)\ a_x(2)\ a_x(3)\right]^{\mathrm{T}} \tag{8.8}$$

$$\mathbf{P} = \begin{bmatrix} 1 & p_x(1) & p_y(1) \\ 1 & p_x(2) & p_y(2) \\ \vdots & \vdots & \vdots \\ 1 & p_x(n) & p_y(n) \end{bmatrix} \tag{8.9}$$

$$\mathbf{K} = \begin{bmatrix} U(|P_1 - P_1|) & U(|P_1 - P_2|) & \cdots & U(|P_1 - P_n|) \\ U(|P_2 - P_1|) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ U(|P_n - P_1|) & \cdots & \cdots & U(|P_n - P_n|) \end{bmatrix} \tag{8.10}$$

$P_i = (p_x(i), p_y(i))$ is the set of landmark points in the first image, $p_x(i)$ is the $x$-coordinate of point $i$ in set $P_i$, $Q_i = (q_x(i), q_y(i))$ is the set of corresponding points in the second image, and $n$ is the number of landmark points.

In [14], a method is presented to estimate approximating thin-plate splines. These splines do not exactly interpolate all given points, but are allowed to approximate them in favor of a smoother transformation. The smoothness is controlled by a parameter $\lambda$, which weights the optimization of landmark distance and smoothness. For $\lambda = 0$, there is full interpolation, while for very large $\lambda$, there is only an affine transformation left.

For equal isotropic errors at all landmarks, the optimal TPS parameters can be found by solving the following system of equations:

$$(\mathbf{K} + \lambda \mathbf{I})\,\mathbf{w} + \mathbf{P}\mathbf{a} = \mathbf{v} \tag{8.11}$$

$$\mathbf{P}^{\mathrm{T}}\mathbf{w} = 0 \tag{8.12}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix.

In fingerprint matching, it is essential to use approximating thin-plate splines, since this introduces some insensitivity to errors. For instance, minutiae may be displaced a few pixels by the minutiae extraction algorithm, or false local matches may be included in the global matching stage. Interpolating TPS will include these displacement errors in the registration exactly, resulting in strange un-smooth transformations and incorrect extrapolations. Obviously, a smoother transformation that does not take all small details into account is much more robust. In that case, the TPS registration represents the elastic distortions, while in the counting stage, the threshold $r_0$ takes care of local minutiae displacements.

The TPS model is fitted in a number of iterations. First, an initial model is fitted to the minutiae in the minutia neighborhood pairs that were found in the local matching stage. Next, the corresponding minutiae in both sets, differing in location and orientation less than a threshold, are determined and a new model is fitted to those corresponding minutiae. This is repeated with a decreasing threshold $r_0$ until the model has converged to its final state. This iterative process improves the quality of the nonlinear registration considerably and increases the matching score significantly. Finally, the matching score $S$ is calculated by:

$$S = \frac{n_{\mathrm{match}}^2}{n_1 \cdot n_2} \tag{8.13}$$

where $n_{\text{match}}$ is the number of matching minutiae, $n_1$ the number of minutiae in the test fingerprint and $n_2$ the number of minutiae in the template fingerprint. This expression provides the most consistent matching scores when the numbers of minutiae in both fingerprints $n_1$ and $n_2$ are significantly different. The match or non-match decision is then taken by comparing the matching score to a threshold.

Figure 8.1(b) shows the two deformed fingerprints after registration by means of thin-plate splines. The figure clearly shows the much more accurate registration with respect to part (a) of the figure. This means that a much lower threshold $r_0$ can be used in the counting stage, leading to a considerably lower false acceptance rate.

## 4. Shape Matching

Once an elastic registration model that aligns two fingerprints is available, a wide range of additional fingerprint matching algorithms becomes available. In this section, we present a shape matching algorithm that makes use of the directional field. The goal of this algorithm is to verify whether the shapes of the fingerprints after alignment match, and so to discard false matches that accidentally have a large minutiae matching score.

The first step is to warp the test print such that it is aligned with the template print. For this purpose, the estimated TPS model is evaluated on a coarse grid, and the image is warped by a set of bilinear transforms that are defined by the grid points. The result is that both images are aligned elastically, and the ridge lines of both prints are on top of each other in the ideal case (see Figure 8.1(b)).

The similarity score is based on the squared directional field [11], which is encoded in vector $\mathbf{v}$ of two elements with norm $|\mathbf{v}| = 1$ at each point in the image. For each vector, there exists an associated coherence value $W \in [0, 1]$ that indicates the reliability of the directional field estimate. The dissimilarity or distance $d$ between two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, given their coherence values $W_1$ and $W_2$ is defined as:

$$d = \sqrt{W_1 W_2}\,(1 - \mathbf{v}_1 \cdot \mathbf{v}_2) \tag{8.14}$$

where $\mathbf{v}_1 \cdot \mathbf{v}_2$ denotes the inner product of $\mathbf{v}_1$ and $\mathbf{v}_2$. The inner product measures the difference in orientation of both vectors. The coherence weighting as defined in Equation (8.14) has been defined in such a way that dissimilarities in the orientation are penalized, with lower penalties if the orientation estimates are less reliable.

The correspondence of the shape of two fingerprints is found as the mean shape distance over the entire valid fingerprint area after elastic registration. The shape matching score can be combined with the elastic minutiae matching score by taking the average of the two scores.

## 5. Results

The proposed algorithm has been evaluated by applying it to Database 2 of FVC2000 [15] and Database 2 of FVC2002 [16]. The FVC2000 database consists of 880 optical 8-bit grayscale fingerprints, eight prints of each of 110 distinct fingers. The images are captured at 500 dpi, resulting in image sizes of $256 \times 364$ pixels. The FVC2002 database consists of the same number of fingerprints, but captured at 569 dpi, resulting in image sizes of $296 \times 560$ pixels.

Unfortunately, no benchmarks are available in the literature to measure the performance of minutiae-based matching for given fixed sets of minutiae. Any result reported on databases such as those of FVC2000 incorporate the performance of a minutiae extraction stage, which is not a topic of this chapter. However, for the sake of the experiments, a straightforward method was implemented consisting of segmentation, Gabor filtering, binarization, thinning and postprocessing [17].

First, the estimated elastic deformation models have been evaluated visually. In Figure 8.4, the registration results are depicted for typical and heavy distortions in the FVC2000 database. The figure shows the superposition of both minutiae sets (indicated by '×' and 'o') and a grid that visualizes the deformations. The figures clearly show that elastic registration makes the minutiae sets fit much better, i.e. the corresponding '×'s and 'o's are much closer to each other, while the fingerprints are not heavily distorted. For the upper row, the matching scores are 0.42 for the rigid registration and

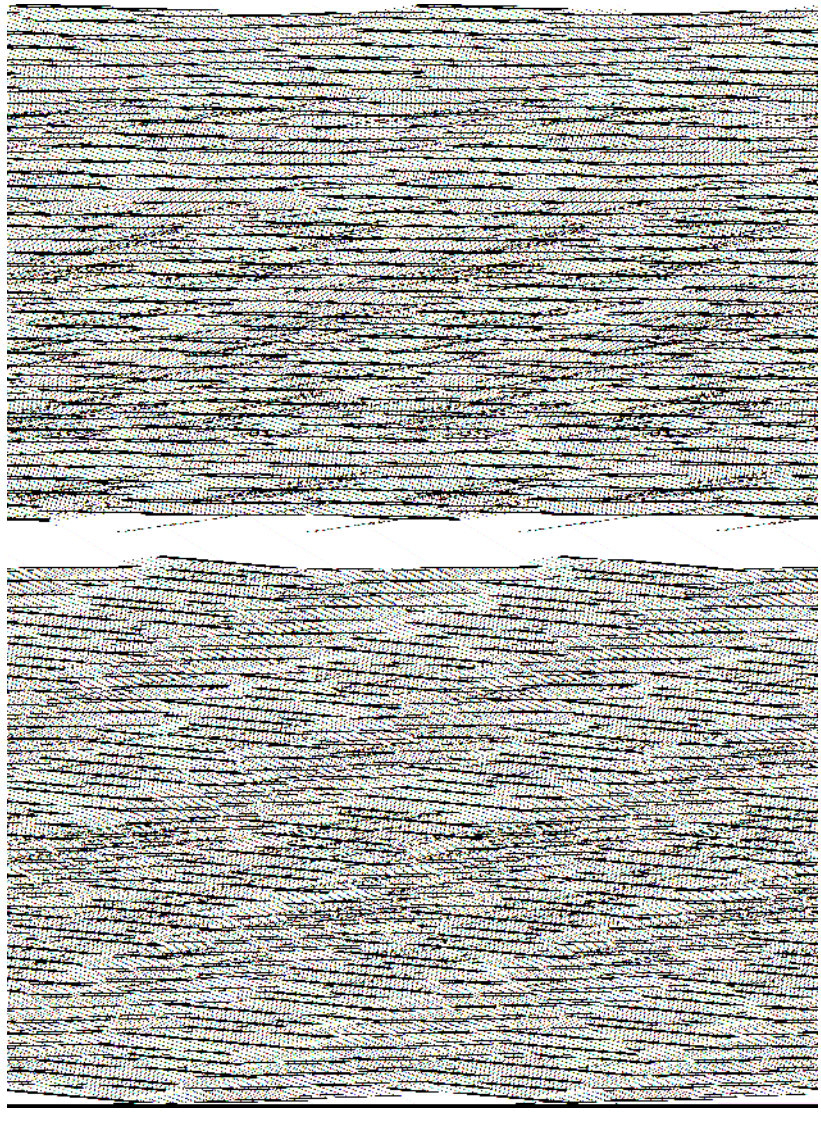


**Figure 8.4** Registration results for the elastic matching algorithm: the superposition of both minutiae sets (indicated by '×' and 'o') and a grid that visualizes the deformations. Top row: typical distortions; bottom row: heavy distortions; left column: rigid registration; right column: TPS registration.

0.68 for the TPS registration, while the scores for the bottom row are 0.04 and 0.25. Furthermore, it is worth noticing that all distortion patterns that we inspected were similar to the patterns that are shown in Figure 8.4, while none of them resembled the distortion model that was proposed in [6].

Next, due to the lack of benchmark results for minutiae matching performance, it was decided to compare TPS-based elastic matching to rigid matching. In both cases, $r_0$ was chosen such that the matching performance was optimized. With $r_0 = 15$ for rigid matching and $r_0 = 5$ for elastic matching, the equal-error rates of the ROC turned out to be 4 % and 2.4 % respectively for Database 2 of FVC2002. In this experiment, 5600 genuine attempts and 9900 impostor attempts have been evaluated. The Receiver Operating Curve (ROC) for the elastic matching experiment is shown in Figure 8.5.

Analysis of false rejects resulted in two possible causes for the low matching scores. The first cause is due to imperfections in the used minutiae extraction stage. Low-quality regions are discarded by the segmentation that precedes the minutiae extraction, leading to very few corresponding minutiae. It is expected that the matching performance increases when combined with better minutiae extraction algorithms. The second cause is a very small overlapping region between the two fingerprints that are matched. In this case too, there are only very few corresponding minutiae. This problem has been solved partially by determining the rectangular overlapping region from the corresponding minutiae pairs, and using only the minutiae in that region for calculation of the matching score, i.e. adjusting $n_1$ and $n_2$ in Equation (8.13).

Furthermore, it is worth noticing that none of the low matching scores is caused by the presence of elastic deformations. The algorithm correctly resolves these distortions, while in some cases, the rigid matching algorithm is not able to do so.

The proposed elastic minutiae matching algorithm is rather fast. In a C++ implementation on a 1 GHz P-III machine, the entire elastic minutiae matching algorithm takes less than 100 ms. Furthermore, it is only marginally more complex than the rigid matching algorithm. The local matching stage takes approximately 50 ms, rigid matching would take 10 ms and elastic matching takes 30 ms.

The combination of elastic minutiae matching and shape matching has also been evaluated by experiments on Database 2 of FVC2002. This results in an equal-error rate, EER = 2.2 %, and a false

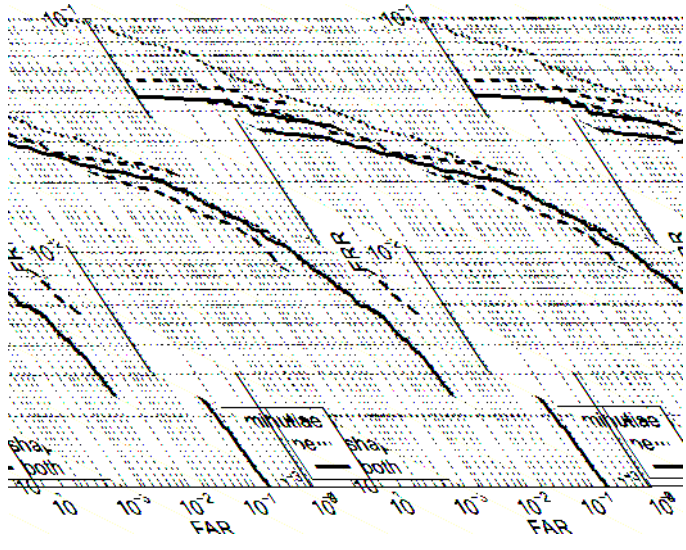



**Figure 8.5** ROCs of minutiae matching, shape matching and a combination of both on Database 2 of FVC2002.

rejection rate, FRR = 5 % at FAR = 0 %. It is worth noticing that 52 out of 100 users are completely error free, and 16 more users have an FRR smaller than 10 % at FAR = 0 %.

## 6. Conclusions

This chapter has proposed a novel minutiae matching algorithm that is able to deal with elastic distortions of fingerprints. Using thin-plate splines, the algorithm handles all possible nonlinear distortions while using very tight bounding boxes. It has been shown that it is able to register distorted fingerprints very well. When applied to elastically deformed fingerprints, the elastic matching algorithm provides considerably better matching scores than rigid matching algorithms. The additional step of shape matching after elimination of the elastic deformations is able to reduce the error rates further. Since a relatively simple minutiae extraction algorithm was used, it is expected that the matching performance can be improved by linking the proposed matching algorithm to better minutiae extraction algorithms.

## Acknowledgment

This research is supported by the Technology Foundation STW, the applied science division of NWO and the technology program of the Ministry of Economic Affairs.

## References

[1] Bazen, A. M. and Gerez, S. H. "Fingerprint matching by thin-plate spline modeling of elastic deformations," *Pattern Recognition*, **36**(8), pp. 1859–1867, 2003.

[2] Pankanti, S., Prabhakar, S. and Jain, A. K. "On the individuality of fingerprints," In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, Hawaii, December 2001.

[3] Kovács-Vajna, Z. M., "A fingerprint verification system based on triangular matching and dynamic time warping," *IEEE Transactions on PAMI*, **22**(11), pp. 1266–1276, 2000.

[4] Ratha, N. K., Bolle, R. M., Pandit, V. D. and Vaish, V. "Robust fingerprint authentication using local structural similarity," In *Proceedings of the 5th IEEE Workshop on Applied Computing Vision*, pp. 29–34, 2000.

[5] Senior, A. W. and Bolle, R. "Improved fingerprint matching by distortion removal," *IEICE Transactions on Information and Systems, Special issue on Biometrics*, **E84-D**(7), pp. 825– 831, July 2001.

[6] Cappelli, R., Maio, D. and Maltoni, D. "Modelling plastic distortion in fingerprint images," In *Proceedings of ICAPR2001, Second International Conference on Advances in Pattern Recognition*, Rio de Janeiro, March 2001.

[7] Cappelli, R., Erol, A. Maio, D. and Maltoni, D. "Synthetic fingerprint-image generation," In *Proceedings of ICPR2000, 15th International Conference on Pattern Recognition*, Barcelona, Spain, September 2000.

[8] Chui, H. and Rangarajan, A. "A new algorithm for non-rigid point matching," In *Proceedings of CVPR*, **2**, pp. 40–51, 2000.

[9] Kumar, S., Sallam, M. and Goldgof, D. "Matching point features under small nonrigid motion," *Pattern Recognition*, **34**(12), pp. 2353–2365, 2001.

[10] Jiang, X. and Yau, W. Y. "Fingerprint minutiae matching based on the local and global structures," In *Proceedings of ICPR2000, 15th International Conference on Pattern Recognition*, **2**, pp. 1042–1045, 2000.

[11] Bazen, A. M. and Gerez, S. H. "Systematic methods for the computation of the directional field and singular points of fingerprints," *IEEE Transactions on PAMI*, **24**(7), pp. 905–919, 2002.

[12] Jain, A. K., Hong, L., Pankanti, S. and Bolle, R. "An identity-authentication system using fingerprints," *Proceedings of the IEEE*, **85**(9), pp. 1365–1388, 1997.

[13] Bookstein, F. L. "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Transactions on PAMI*, **11**(6), pp. 567–585, 1989.

[14] Rohr, K., Fornefett, M. and Stiehl, H. S. "Approximating thin-plate splines for elastic registration: Integration of landmark errors and orientation attributes," In *Proceedings of the 16th International Conference on Information Processing in Medical Imaging*, LNCS 1613, pp. 252–265, June 1999.

[15] Maio, D., Maltoni, D., Cappelli, R., Wayman, J. L. and Jain, A. K. "FVC2000: Fingerprint verification competition," *IEEE Transaction on PAMI*, **24**(3), pp. 402–412, 2002.

[16] Maio, D., Maltoni, D., Cappelli, R., Wayman, J. L. and Jain, A. K. "FVC2002: Second fingerprint verification competition," In *Proceedings of the 16th International Conference on Pattern Recognition (ICPR2002)*, **3**, pp. 811–814, August 2002.

[17] Bazen, A. M. and Gerez, S. H. "Achievements and challenges in fingerprint recognition," In Zhang, D. (ED), *Biometric Solutions for Authentication in an e-World*, Kluwer, pp. 23–57, 2002.

# 9

# Personal Authentication Using the Fusion of Multiple Palm-print Features

**Chin-Chuan Han**
Department of Computer Science and Information Engineering, Chung-Hua University, Hsinchu, Taiwan, R.O.C.

*In this chapter, we propose a scanner-based personal authentication system by using the fusion of multiple palm-print features. For higher accuracy rates in biometrics-based authentication systems, three requirements should be met: personalization, fusion and adaptive thresholding. In order to satisfy these requirements, personal reference templates, an optimal Positive Boolean Function (PBF) and an automatically determined threshold value must be obtained from the training samples of a specified individual. Multiple reference templates of various features are constructed to represent the characteristics of an individual. The multiple verified scores of a query sample are determined by matching with the reference templates in databases. An optimal PBF is found and used to fuse the verified scores to obtain a fused value. This value is compared with the predefined threshold to make the decision. Experimental results verify the validity of our proposed approaches in personal authentication.*

## 1. Introduction

Recently, biometric features have been widely used in many personal authentication applications because they possess the following physiological properties [1]: *universality, uniqueness, permanence, collectability, performance, acceptability* and *circumvention*. According to the above properties, many access control systems adopt biometric features to replace the digit-based password. Biometric features are the features extracted from human biological organs or behavior. The first book addressing various biometric technologies for personal identification in networked society was edited by Jain *et al.* in 1999 [2]. In this book, they make a detailed comparison of 14 different biometric technologies. O'Gorman [3] also surveyed six biometric features in the matching, validation, maximum independent

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

samples per person, sensor cost and sensor size topics. Though fingerprint and eye features provide very high recognition rates, they are unsuitable for identification systems. First, the sensor cost of eye-based features is too high to implement in many low-security demanding applications such as computers, home security systems, restricted entry control, corporate networks, etc. Besides, since fingerprint features are used officially in criminal investigations and commercial transactions, most of the users are unwilling to deliver their fingerprint data to a company or system for privacy reasons. Jain *et al.*[2] mentioned that 'The match between biometrics and an application is determined depending upon the requirements of the given application, the characteristics of the applications and properties of the biometrics.'

Two possible biometric features can be extracted from human hands. First, hand shape geometrical features, such as finger width, length and thickness, are the well-known features adopted in many systems [4,5,6]. For instance, Golfarelli *et al.*[4] extracted 17 hand shape features to verify personal identity. Zunkei [5] introduced a commercial product of hand geometry-based recognition and applied it to many access control systems. Jain *et al.*[6] used deformable matching techniques to verify individuals via hand shapes. The hand shape of a test sample is aligned with that in the database, and a defined distance is calculated for the evaluation of similarity. A 96.5 % accuracy rate and a 2 % False Acceptance Rate (FAR) are achieved by their approaches. These hand shape-based features frequently vary due to the wearing of rings on fingers. Besides, the width or thickness of women's fingers may rapidly vary in a short time due to pregnancy. According to the variation of hand geometry, it can be used in entry control systems with low security requirements and a low rejection rate to record the entry data of employees or users. The reference features in the database should be updated frequently.

The relatively stable features extracted from the hands are the prints of palms. Zhang and Shu [7] applied the datum point invariant property and the line feature matching technique to conduct the verification process via palm-print features. They inked the palm-prints on to paper and then scanned them to obtain $400 \times 400$ images. This method is not suitable for many online security systems because two steps are needed to obtain the palm-print images in their approach. Joshi *et al.* [8] captured an image of the middle finger by using a CCD camera to generate a Wide Line Integrated Profile (WLIP) of length 472. They also used the normalized correlation function to compute the similarity values between the input sample and the reference templates. Furthermore, Zhang [9] proposed a texture-based feature extraction method to obtain the global attributes of a palm. A dynamic selection scheme was also designed to ensure the palm-print samples were correctly and effectively classified in a large database.

In order to improve the verification performance, fusion schemes on multimodal features or verified results have been used widely in biometric-based personal authentication systems. They generate more accurate results than a single constituent classifier. Combinational strategies using averaging, AND, OR, maximum, minimum or voting rules are well-known methods for multiple classifier fusion [10]. In biometric-based authentication applications, Prabhakar and Jain [11] proposed four different fingerprint-matching algorithms and combined them at the decision level. You *et al.* [12] designed a coarse-to-fine searching process using multiple palm-print features to efficiently find the best matching sample in the database. Chibelushi *et al.* [13] made a review of speech-based bimodal recognition. They combined auditory and visual modalities instead of a single modality. Sanderson and Paliwal [14] proposed a feature set, called maximum autocorrelation values, to improve the robustness of a text-independent verification system. They also integrated the audio and visual information in a multimodal verification system. Furthermore, Chatzis *et al.* [15] proposed three fusion schemes to combine still face and speech features. All of their approaches improved the performance of the system with multimodal features.

In this paper, we propose a scanner-based personal authentication system by using multiple palm-print features. Two stages, enrollment and verification, constitute the identification system, as shown in Figure 9.1. In the enrollment stage, $M$ hand images of an individual are collected to be the training samples. These samples should be processed by preprocessing, feature extraction and modeling modules to generate the matching templates. In the verification stage, a query sample is also processed by the preprocessing and feature extraction modules, and then matched with the templates to decide whether

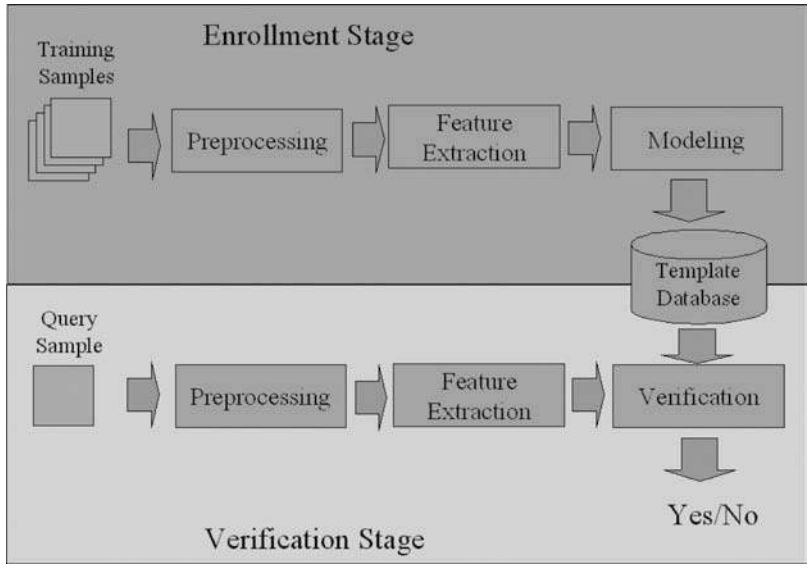



**Figure 9.1** The modules of biometric-based verification systems.

it is a genuine sample or not. To achieve more accuracy, a fusion scheme based on the Positive Boolean Function (PBF) was devised to integrate the multimodal preverified results. The PBF has been successfully applied to the design of a stack filter. Each stack filter corresponding to a PBF possesses the weak superposition property known as threshold decomposition and the ordering property known as the stacking property [16]. This class of stack filters includes many familiar filter types, such as a median filter, a weighted median filter, an order statistic filter, a weighted order median filter, and so on. The main function of a stack filter is to remove the noise, detect the edges, . . . , etc. All their applications focus on image and signal processing. An optimal stack filter $S_f$ was defined as a filter whose Mean Absolute Error (MAE) value between the output of PBFs and the desired signals is minimal. In the supervised enrollment stage of the authentication applications, the identities of the training samples were previously defined and identified. These identities of samples can be defined to be the desired values. The outputs of the PBFs were considered as the integrated results. The positive Boolean function was applied to design a fusion scheme for personal authentication.

The rest of this chapter is organized as follows. In Section 2, four steps for the preprocessing module are executed to find the location of the Region Of Interest (ROI). The feature extraction techniques, including Sobel's and morphological operations are described in Section 3. The modeling procedure for verification purposes is introduced in Section 4. First, the simple and practical multiple template matching method is described in the section to evaluate the similarity between the query and reference samples. Next, the fusion scheme with the PBF is described to increase the system performance. In Section 5, experimental results are demonstrated to verify the validity of our proposed algorithms. Finally, some concluding remarks are given in Section 6.

## 2. Preprocessing

Image preprocessing is usually the first and most essential step in pattern recognition. In our proposed approach, four steps are devised in the preprocessing module. Image thresholding, border tracing, wavelet-based segmentation and Region Of Interest (ROI) location are sequentially executed to obtain

a square region which possesses the palm-print data. In the following contexts, we will briefly present the process of each step. More details can be found in [17].

## 2.1 Step 1: Image Thresholding

The hand images of 256 gray levels are acquired from a platform scanner, as shown in Figure 9.2(a). The image thresholding operation is to binarize the gray images to obtain binary hand shape images. Since the capturing environment is stable and controlled, the threshold value is conveniently fixed to be 70 in our experiments.

## 2.2 Step 2: Border Tracing

After the image thresholding step, the binary images are traced to obtain the contours of hand shape, as shown in Figure 9.2(b), by making use of the border tracing algorithm. The details of the border tracing algorithm can be found in [18].

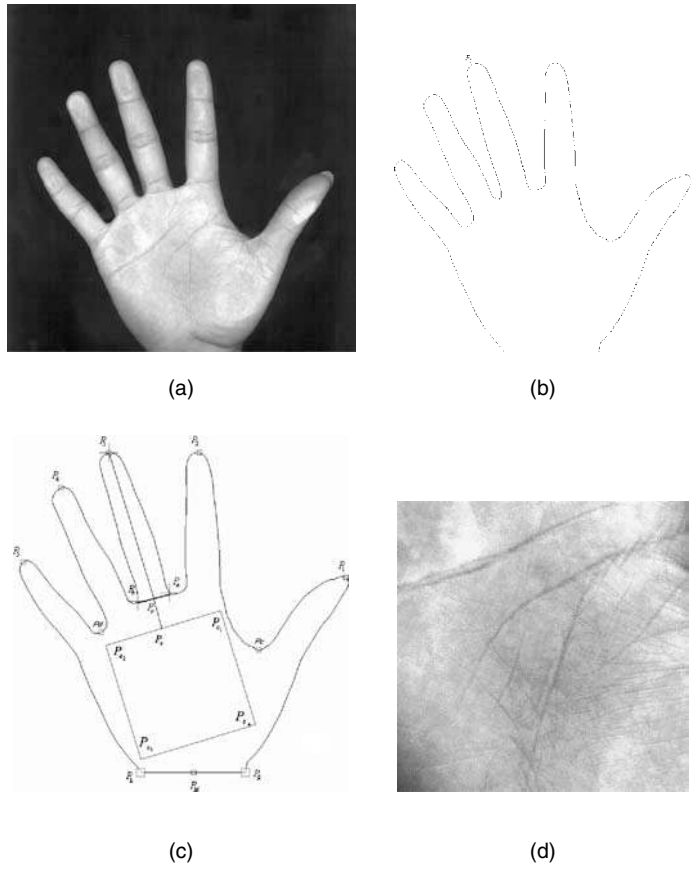



**Figure 9.2** The preprocessing steps. (a) Image thresholding; (b) border tracing; (c) corner detection; and (d) ROI locating.

### 2.3 Step 3: Wavelet-based Segmentation

In the previous step, the border pixels of hand shape are sequentially traced and represented by a set of coordinates $(x_i, y_i), i = 1, 2, \ldots$. In this step, the wavelet-based segmentation technique is adopted to find the locations of five fingertips and four finger roots. As we know, these points are located at the corners of the hand shape. Since the crucial corner points $P_a$, $P_b$ and $P_3$ (see Figure 9.2(c)) determine the ROI location in the palm table, it is very important to explicitly locate the corner points of the hand shape. The wavelet transform can provide stable and effective segmented results in corner detection. The detected corner points in Figure 9.2(b) are illustrated in Figure 9.2(c).

### 2.4 Step 4: Region Of Interest (ROI) Generation

In this step, we will find the Region Of Interest (ROI) in the palm table. Based on the result generated in Step 3, the location of the ROI is determined from points $P_a$, $P_b$, $P_3$ and geometrical formulae. Thus, the square region $P_{e_1}, P_{e_2}, P_{e_3}, P_{e_4}$ of size 256 by 256 is defined to be the region of interest, as shown in Figure 9.2(c). From these four points, the image of the ROI is cut from the hand image, as shown in Figure 9.2(d).

## 3. Feature Extraction

Feature extraction is a step to extract the meaningful features from the segmented ROI for the later modeling or verification process. In extracting the features, we use the operator-based approach to extract the line-like features of palm-prints in the ROI of the palm table.

First, we employ simple Sobel operators to extract the feature points of a palmprint. Four directional Sobel operators $S_0$, $S_{90}$, $S_{45}$ and $S_{135}$ are designed. Consider a pixel of the ROI in the palm table, four directional Sobel operators are performed to select the maximal value as the resultant value of the ROI. This operation is carried out according to the following expression:

$$f^*S = \max(f^*S_0, f^*S_{90}, f^*S_{45}, f^*S_{135}) \tag{9.1}$$

Here, symbol $^*$ is defined as the convolution operation. The Sobel features of the ROI are thus obtained, as shown in Figure 9.3(a).

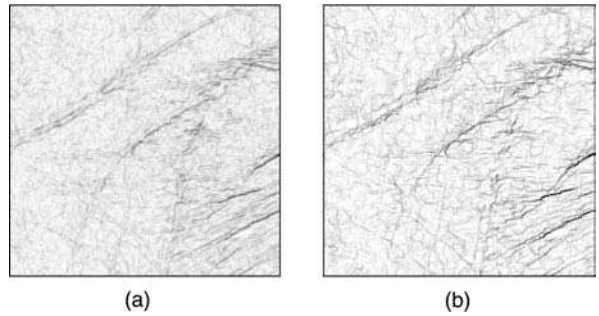



**Figure 9.3** The feature extraction module. (a) The features operated via Sobel operation; and (b) the features operated via morphological operation and linear stretching operation.

Next, we present other complex morphological operators to extract palm-print features. In grayscale morphology theory, two basic operations *dilation* and *erosion* for image $f$ are defined as follows:

$$\text{Dilation}: f \oplus S = \max\{f(s-x)+b(x) | (s-x) \in D_f, \text{and } x \in D_b\} \tag{9.2}$$

$$\text{Erosion}: f \Theta S = \min\{f(s+x)-b(x) | (s+x) \in D_f, \text{and } x \in D_b\} \tag{9.3}$$

Here, $D_f$ and $D_b$ represent the domains of image $f$ and structuring element $b$. In addition, two combination operations called *opening* ($f \circ b$) and *closing* ($f \bullet b$) are extended for further image processing.

In [19], Song, Lee and Tsuji designed an edge detector called an *Alternating Sequential Filter* (ASF), which provides perfect effects in noisy or blurry images. The mechanism of the ASF is constructed as follows. Two filters are defined to be:

$$f_1 = \gamma_l \phi_l \text{ and } f_2 = f_1 \oplus b_{3x3} \tag{9.4}$$

The algebraic opening $\gamma_l$ and closing $\phi_l$ are defined as:

$$\gamma_l = \max(f \circ b_{0,l}, f \circ b_{45,l}, f \circ b_{90,l}, f \circ b_{135,l}) \tag{9.5}$$

$$\phi_l = \min(f \bullet b_{0,l}, f \bullet b_{45,l}, f \bullet b_{90,l}, f \bullet b_{135,l}) \tag{9.6}$$

where symbol $b_{\alpha,l}$ denotes the structuring elements of length $l$ and angle $\alpha$. In our experiments, value $l$ is set to be 5. Next, the morphological operator is defined to be $f_m = f_2 - f_1$. The edge pixels are thus obtained by using the morphological function $f^* f_m$, as shown in Figure 9.3(b).

Now, the feature vectors are created in the following way. For the training samples, the ROI images are uniformly divided into several small grids. The mean values of pixels in the grids are calculated to obtain the feature values. These values are sequentially arranged row by row to form the feature vectors.

## 4. Enrollment and Verification Processes

In an authentication system, two phases, *enrollment* and *verification*, should be executed. In our previous works [17], we have designed a simple and practical technique, *multiple template matching*, to model the verifier of a specific person. In this chapter, a fusion scheme with the PBF will be designed to increase the system performance.

### *4.1 Multitemplate Matching Approach*

Template matching using the correlation function is a common and practical technique utilized in many pattern recognition applications. In this study, we try to use this approach to perform the verification task to decide if the query sample is a genuine pattern or not. Consider a query sample **x** and a template sample **y**, a *correlation* function is utilized to measure the similarity between two feature vectors as follows:

$$R_{xy} = \frac{\sum_{i=1}^{n} (x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y} \tag{9.7}$$

In the above formula, symbols $\mu$ and $\sigma$ represent the mean and standard deviation values, respectively. In addition, value $n$ is the length of the feature vectors. The coefficient value of this linear correlation function is calculated for the similarity evaluation.

In creating the reference templates in the enrollment stage, $M$ samples of individual $X$ are collected to form the matching template database. The main advantage of this approach is that less training time is needed in training the matching model. In the verification stage, the correlation coefficient of query and reference samples is calculated by making use of Equation (9.7). If the reference and test patterns are both derived from the same person, the coefficient value will be approximately one. Otherwise, if the value of similarity approximates to zero, the query sample should be considered as a forged pattern.

From the preceding contexts, the metric we define in determining whether a query sample is genuine or a forgery can be modified to $1-R$. Based on this criterion, it is easy to verify the input pattern by a predefined threshold value $t$. If the value $1-R$ is smaller than threshold $t$, the owner of the query sample is claimed to be individual $X$. Otherwise, the query sample is classified as a forged pattern.

## *4.2 Multimodal Authentication with PBF-based Fusion*

In this section, the fusion scheme with PBF for integrating several preverified results is proposed. In authentication systems, the verification problem can be considered to be a two-category (classes $\varpi_y$ and $\varpi_n$) classification problem in pattern recognition. The similarity value of an input sample generated from the matching algorithm with the templates in the database was compared with a given threshold. This value would determine whether the input sample was a genuine one or not; it was critical in determining the verified results. Since the filtering process using the PBF was performed on the domain of integer values, all similarity values in this study had to be normalized and quantized to be integer values in the range [0,$L$] by using sigmoidal normalization by means of the following equations:

$$\xi_i(\mathbf{x}) = \frac{\varepsilon_i(\mathbf{x}) - \mu_i}{\sigma_i} \tag{9.8}$$

and

$$h(\xi_i(\mathbf{x})) = Q\left(\frac{1}{1+\exp(-t\xi_i(\mathbf{x}))}\right),\ i = 1, 2, \ldots, n \tag{9.9}$$

Here, values $\mu_i$ and $\sigma_i$ denote the mean and standard deviation of feature $R_i$ of the training samples. $Q$ is a quantization function for generating the values in the range [0,$L$].

In considering a sample $\mathbf{x}$ of a specified individual $X$ possessing $n$ modal similarity values $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, which were generated from $n$ modal features or matching algorithms, each element was normalized and quantized to be in the range [0,$L$]. In the verification stage with a single modal rule, if sample $\mathbf{x}$ is a genuine sample, i.e. $\mathbf{x} \in \varpi_y$, then the similarity value should ideally be smaller than the threshold. The similarity value is larger than the threshold value if it is a forged sample, i.e. $\mathbf{x} \in \varpi_n$. Thus, the simple verification rule using one single similarity value $x_i$ for sample $\mathbf{x}$ is designed as:

$$D(\mathbf{x}) = \begin{cases} \mathbf{x} \in \varpi_y(\text{genuine class}) & \text{if } x_i < l \\ \mathbf{x} \in \varpi_n(\text{forgery class}) & \text{otherwise} \end{cases} \tag{9.10}$$

Here, value $l$ is a threshold value in the range $[B_l, B_u]$, and the two boundary values $B_l$ and $B_u$ are manually predefined. The verified result of sample $\mathbf{x}$ is assigned to be value 0 if its similarity value is less than the threshold. Otherwise, the verified result is set to value 1. The verification rule is modified as:

$$x_i^l = D_l(x_i) = \begin{cases} 0 & \text{if } x_i < l, (\text{i.e.}, \mathbf{x} \in \varpi_y) \\ 1 & \text{if } x_i \geq l, (\text{i.e.}, \mathbf{x} \in \varpi_y). \end{cases} \tag{9.11}$$

where, $l = B_{l+1}, B_{l+2}, \ldots, B_u$. This rule is the same as for the threshold function $T_l(.)$ in [16].

In multimodal verification, the threshold value $l$ is compared and set from $B_{l+1}$ to $B_u$ in determining the verified results. The integration function $D_f(.)$ for the individual $X$ is defined as a PBF to integrate the $n$ similarity values $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and its output value $D_f(\mathbf{x})$ is used to determine the verification result. Since the PBF possesses the thresholding decomposition property, the sample $\mathbf{x}$ of $n$ similarity values can be decomposed into $(B_u - B_l)$ binary vectors $x^l = (x_1^l, x_2^l, \ldots, x_n^l) \in \{0, 1\}^n$ by using the threshold function as defined in Equation (9.11). This function can be reconsidered as a penalty function if a verification error occurs.

Now, let us show that the verification problem satisfies the stacking property with the PBF. Consider two samples $\mathbf{x}$ and $\mathbf{y}$ whose corresponding similarity attributes are $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, respectively. If sample $\mathbf{x}$ is nearer to class $\varpi_y$ than sample $\mathbf{y}$, two relations along class $\varpi_y$ and two samples $\mathbf{x}$ and $\mathbf{y}$ should be satisfied as below:

**C1**: If sample $\mathbf{x}$ does not belong to class $\varpi_y$ (e.g. $T_l(D_f(\mathbf{x})) = 1$), then sample $\mathbf{y}$ does not belong to class $\varpi_y$ (e.g. $T_l(D_f(\mathbf{y})) = 1$).

**C2**: If sample $\mathbf{y}$ is an element of class $\varpi_y$ (e.g. $T_l(D_f(\mathbf{y})) = 0$), then sample $\mathbf{x}$ should be an element of class $\varpi_y$ (e.g. $T_l(D_f(\mathbf{x})) = 0$).

The binary vectors $\mathbf{u}^l$ and $\mathbf{v}^l$ are obtained from multimodal attributes $\mathbf{x}$ and $\mathbf{y}$ thresholded at level $l$. Here, $\mathbf{u}^l = (x_1^l, x_2^l, \ldots, x_n^l)$, $\mathbf{v}^l = (y_1^l, y_2^l, \ldots, y_n^l)$, and $l = B_l, B_{l+1}, \ldots, B_u$. If $\mathbf{u}^l \leq \mathbf{v}^l$ for all dimensional elements (e.g. $x_i \leq y_i$, $i = 1, 2, \ldots, n$) and for all levels $l = B_{l+1}, B_{l+2}, \ldots, B_u$, then sample $\mathbf{x}$ is nearer to the class $\varpi_y$ than sample $\mathbf{y}$ according to the voting strategy. The verification function for samples $\mathbf{x}$ and $\mathbf{y}$ is thus defined to be a PBF at all levels. The following are the verification results for samples $\mathbf{x}$ and $\mathbf{y}$ if the binary vectors $\mathbf{u}^l \leq \mathbf{v}^l$ and $l = B_{l+1}, B_{l+2}, \ldots, B_u$.

**C1**: If the verification result for sample $\mathbf{x}$ at level $l$ is output 1, i.e. $f(\mathbf{x}^l) = 1$, then the result from sample $\mathbf{y}$ will be 1, $f(\mathbf{v}^l) = 1$.

**C2**: If the result for sample $\mathbf{y}$ at level $l$ is output 0, i.e. $f(\mathbf{v}^l) = 0$, then the result from sample $\mathbf{x}$ will be 0, $f(\mathbf{u}^l) = 0$.

That means that if $\mathbf{u}^l \leq \mathbf{v}^l$, then $f(\mathbf{u}^l) \leq f(\mathbf{v}^l)$ for all levels. Therefore, the verification function using the PBF satisfies the stacking property.

$M$ genuine samples of a specified individual $\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(M)$, called the positive samples of class $\varpi_y$, and $N$ forged samples $\mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(N)$, called the negative samples, were collected to be the training set $\mathbf{T}$. These samples are normalized and quantized to be in the range [0,$L$]. In the training stage, the samples in class $\varpi_y$ were previously identified to be of zero value and the samples in class $\varpi_n$ were assigned to one desired value, i.e.

$$d_l(\mathbf{x}) = \begin{cases} 0 & \text{for all levels } l = B_{l+1}, B_{l+2}, \ldots, B_u, \text{ when } \mathbf{x} \in \varpi_y \\ 1 & \text{for all levels } l = B_{l+1}, B_{l+2}, \ldots, B_u, \text{ when } \mathbf{x} \in \varpi_n \end{cases} \tag{9.12}$$

Thus, the desired values (ideal values) for the positive samples and the negative samples can be defined in the following equation to satisfy the above requirement.

$$D(\mathbf{x}) = \begin{cases} B_l & \text{when } \mathbf{x} \in \varpi_y \\ B_u & \text{when } \mathbf{x} \in \varpi_n \end{cases} \tag{9.13}$$

In the verification process, the cost of misclassification is defined to be the absolute value between the desired value and the output of the integration function. For any sample in the training set $\mathbf{T}$, the absolute error is thus defined as:

$$C(D_f(\mathbf{x})) = \begin{cases} |D_f(\mathrm{x}) - B_l| & \text{when } \mathbf{x} \in \varpi_y \\ |D_f(\mathrm{x}) - B_u| & \text{when } \mathbf{x} \in \varpi_n \end{cases} \tag{9.14}$$

Since the integration function with the PBF possesses the thresholding decomposition property, the cost of misclassification can be formulated as the summation of misclassification errors occurring at each level. At each level, if we make a correct decision, there is no error in the loss function. Otherwise, the loss of one unit is assigned to any error. Therefore, verification errors will occur at (1) the samples not belonging to class $\varpi_y$, i.e. the forged samples (class $\varpi_n$), $d(\cdot) = 1$, $f(\cdot) = 0$; or (2) the samples belonging to class $\varpi_y$ which are not the genuine samples, i.e. $d(\cdot) = 0, f(\cdot) = 1$. Thus, the total number of Verification Errors (*VE*) of a specified PBF $f$, i.e. the MAE, can be obtained from the following formula:

$$\begin{aligned} VE(f) &= E_{\mathbf{x}(j)\in\mathbf{T}}\left[\left|D(\mathbf{x}(j)) - D_f(\mathbf{x}(j))\right|\right] \\ &= E_{\mathbf{x}(j)\in\mathbf{T}}\left[\left|\sum_{l=B_l+1}^{B_u} d_l(\mathbf{x}(j)) - f(T_l(\mathbf{x}(j)))\right|\right] \\ &= E_{\mathbf{x}(j)\in\mathbf{T}}\left[\sum_{l=B_l+1}^{B_u} \left|d_l(\mathbf{x}(j)) - f(T_l(\mathbf{x}(j)))\right|\right] \\ &= \sum_{l=B_l+1}^{B_u} E_{\mathbf{x}(j)\in\mathbf{T}}\left[\left|d_l(\mathbf{x}(j)) - f(T_l(\mathbf{x}(j)))\right|\right] \end{aligned} \tag{9.15}$$

The best integration function $\hat{f}$ is the optimal PBF with the minimal verification error, i.e. $\hat{f} = \arg\min_f VE(f)$. Furthermore, the VE value can be reformulated as the minimal sum over the $2^n$ possible binary vectors of length $n$ in [20]. The best integration function is obtained from the optimal PBF with the minimal verification errors occurring at all levels. As is known, there are 20 stack filters of window width three, 7581 of window width five, and more than $2^{35}$ of window width seven. It takes a tremendous amount of time to compute the *VE* (i.e. MAE) values over a great number of stack filters and make comparisons among them in order to find the optimal solution. Many researchers [20,21,22] have proposed efficient approaches to find the optimal PBF. In our previous work [22], the graphic search-based algorithm further improved the searching process. First, the problem of finding the optimal solution was reduced to the problem of searching a minimal path in the *Error Code Graph* (ECG). Secondly, two graph searching techniques, *greedy* and A* properties, were applied to avoid the search of the extremely large volume of search space. Thus, only a few nodes need to be traversed and examined in this graph. More details can be found in [22].

## *4.3 Adaptive Thresholding*

In many biometric-based verification models, the selection of threshold value $t$ is the most difficult step in the enrollment stage. It will affect the False Acceptance Rate (FAR) and False Rejection Rate (FRR). Basically, these two values are contradicted by each other. The higher the FAR value is, the lower the FRR value becomes, and vice versa. In general, an identification system with lower FAR requirement will be adopted in a higher security system. On the other hand, the systems with lower FRR requirement are used in many user-friendly control systems. The selection of FAR or FRR value depends on the aim of the applications. In order to evaluate the verification performance, the sum of FAR and FRR values is defined to be the performance index **I** in this study. The main goal of threshold selection is to find the minimal values of FAR and FRR for each individual, which will depend on the characteristics of samples of individual $X$. In other words, an individual $X$ should have his own threshold value $t_X$. The leave-one-out cross validation methodology is applied to evaluate the performance index. More details can be found in [17].

## 5. Experimental Results

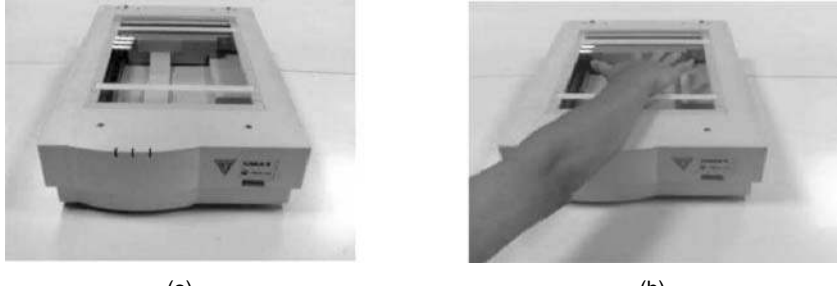
In this section, some experimental results are demonstrated to verify the validity of our approach. First, the experimental environment is set up and described in Section 5.1. Next, the verified results generated by template matching and the PBF-based fusion schemes are demonstrated in Sections 5.2 and 5.3, respectively.

### *5.1 Experimental Environment*

In our experimental environment, a platform scanner, as shown in Figure 9.4(a), is used to capture the hand images. Here, the scanner that we use in our system is a color scanner which is a commercial product of UMAX Co. Users are asked to put their right hands on the platform of the scanner without any pegs, as shown in Figure 9.4(b). The hand images of size 845 by 829 are scanned in grayscale format and in 100 dpi (dot per inch) resolution. Thirty hand images of each individual are grabbed three times within three weeks to construct the database. In the enrollment stage, the first ten images are used to train the verification model. The other 20 images are tested by a trained verifier. Experiments on 1000 positive (genuine) samples and 49 000 negative (forged) samples of 50 people were conducted to evaluate the performance. The verification system is programmed by using the C programming language in a Microsoft Windows environment.

### *5.2 Verification Using a Template Matching Algorithm*

In the first experiment, three kinds of window size: $32 \times 32$, $16 \times 16$ and $8 \times 8$ were adopted to evaluate the performance of the template matching methodology. In each window, the mean value of pixels was computed and considered as an element of vectors. The linear correlation function was used to calculate the similarity between the reference and test samples. Consider a person $X$, ten samples were chosen to be the reference templates of the verifier. These ten positive samples of individual $X$ and 490 negative samples of 49 people were collected to compute the Type I and Type II errors. The results for False Acceptance Rate (FAR) and False Rejection Rate (FRR) by all possible threshold values ranging from 0 to 1 for various grid window sizes were calculated to find the best threshold values, respectively. The threshold value $t_X$ for individual $X$ was chosen by the adaptive selection rule. Thereby, the query samples were verified by the verifier of $X$ and thresholded by the preselected value $t_X$. The multiple template matching algorithm can achieve accuracy rates above 91%, as tabulated in [17].



(a) (b)

**Figure 9.4** The input device for palm-print images.

## 5.3 Verification Using PBF-based Fusion

Three experiments are illustrated in this section to show the effectiveness of our proposed scheme. In this first experiment, the verified results using the single feature and those results using multiple features integrated by the PBF-based fusion scheme are given. The verified results by the PBF-based fusion method are better than those of verifiers using a single feature (see Figure 9.5). In this experiment, the PBF-based scheme selected the best features of size $16 \times 16$. Moreover, four fusion methods, *minimum, maximum, average* and *median*, were utilized to integrate the multiple features in the second experiment. The PBF-based fusion scheme was also adopted to make a comparison with these four methods. The Receiver Operation Characteristic (ROC) curves for these fusion strategies are depicted in Figure 9.6. In the third experiment, the personal threshold value was determined from the training samples. On average, the FAR and FRR values of these five fusion strategies were

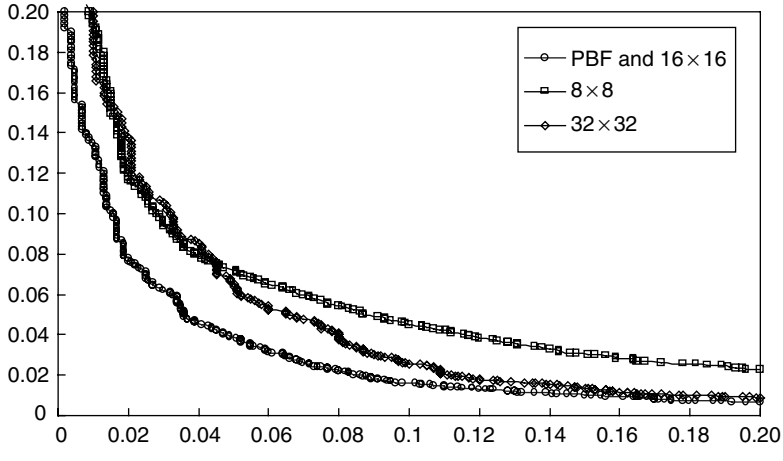



**Figure 9.5** The ROC curves for three kinds of feature and the PBF-based fusion approach.

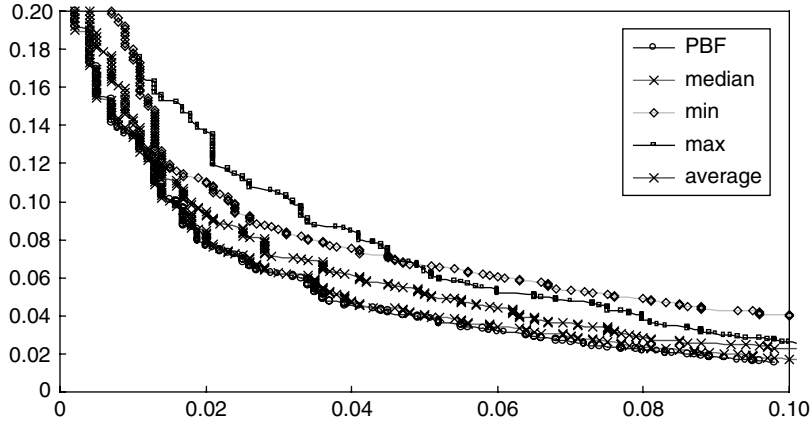



**Figure 9.6** The ROC curves for five fusion schemes.

met at the values $FAR_{\text{PBF}} = 0.043$, $FRR_{\text{PBF}} = 0.017$, $FAR_{\max} = 0.023$, $FRR_{\max} = 0.058$, $FAR_{\min} = 0.019$, $FRR_{\min} = 0.083$, $FAR_{\text{avg}} = 0.021$, $FRR_{\text{avg}} = 0.052$, and $FAR_{\text{med}} = 0.018$, $FRR_{\text{med}} = 0.044$, respectively. Actually, the results generated by the PBF-based scheme are better than those of the others.

## 6. Conclusions

For this study, we have proposed a PBF-based fusion mechanism to integrate the various verified results. This scheme has been applied to palm-print feature-based authentication systems. The hand images were captured from a scanner without any fixed peg. This mechanism is very suitable and comfortable for all users. Besides this, our approach satisfies the personalization, the integration and the adaptive thresholding requirements for biometrics-based authentication systems. The experimental results have confirmed the validity and effectiveness of our approach.

## References

[1] Clarke, R. "Human identification in information systems: Management challenges and public policy issues," *Information Technology & People*, **7**(4), pp. 6–37, 1994.
[2] Jain, A. K., Bolle, R. and Pankanti, S. *Biometrics: Personal Identification in Networked Society*, Kluwer Academic Publishers, 1999.
[3] O'Gorman, L. "Fingerprint Verification," Jain, A. K., Bolle, R. and Pankanti, S. (Eds) in *Biometrics: Personal Identification in Networked Society*, pp. 43–64, Kluwer Academic Publisher, 1999.
[4] Golfarelli, M., Miao, D. and Maltoni, D. "On the error-reject trade-off in biometric verification systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**, pp. 786–796, 1997.
[5] Zunkei, R. L. "Hand geometry based verification" in Jain, A. K., Bolle, R. and Pankanti, S. (Eds) *Biometrics: Personal Identification in Networked Society*, pp. 87–101, Kluwer Academic Publishers, 1999.
[6] Jain, A. K. and Duta, N. "Deformable matching of hand shapes for verification," http://www.cse.msu.edu/~dutanico/, 1999.
[7] Zhang, D. and Shu, W. "Two novel characteristics in palm-print verification: Datum point invariance and line feature matching," *Pattern Recognition*, **32**, pp. 691–702, 1999.
[8] Joshi, D. G., Rao, Y. V., Kar, S., Kumar, V. and Kumar, R. "Computer-vision-based approach to personal identification using finger crease pattern," *Pattern Recognition*, **31**, pp. 15–22, 1998.
[9] Zhang, D. D. *Automated Biometrics: Technologies and Systems*, Kluwer Academic Publishers, 2000.
[10] Kittler, J., Hatef, M., Duin, R. P. W. and Matas, J. "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp. 226–239, 1998.
[11] Prabhakar, S. and Jain, A. K. "Decision-level fusion in fingerprint verification," *Pattern Recognition*, **35**, pp. 861–874, 2002.
[12] You, J., Li, W. and Zhang, D. "Hierarchical palmprint identification via multiple feature extraction," *Pattern Recognition*, **35**, pp. 847–859, 2002.
[13] Chibelushi, C. C., Deravi, F. and Mason, J. S. D. "A review of speech-based bimodal recognition," *IEEE Transactions on Multimedia*, **4**, pp. 23–37, 2002.
[14] Sanderson, C. and Paliwal, K. K. "Noise compensation in a person verification system using face and multiple speech features," *Pattern Recognition*, **36**, pp. 293–302, 2003.
[15] Chatzis, V., Bor's, A. G. and Pitas, I. "Multimodal decision-level fusion for person authentication," *IEEE Transactions on System, Man, and Cybernetics – Part A: Systems and Humans*, **29**, pp. 674–680, 1999.
[16] Wendt, P. D., Coyle, E. J. and Gallagher, N. C. "Stack filter," *IEEE Transactions on Acoustics, Speech, Signal Processing*, **ASSP-34,** pp. 898–911, 1986.
[17] Han, C. C., Cheng, H. L., Lin, C. L., Fan, K. C. and Lin, C.-L. "Personal authentication using palmprint features," *Pattern Recognition*, **36**, pp. 371–382, 2003.
[18] Sonka, M., Hlavac, V. and Boyle, R. *Image Processing, Analysis and Machine Vision*, PWS publishers, 1999.
[19] Song, X., Lee, C. W. and Tsuji, S. "Extraction of facial features with partial feature template," in *Proceedings of Asian Conference on Computer Vision*, pp. 751–754, 1993.

[20] Coyle, E. J. and Lin, J. H. "Stack filters and the mean absolute error criterion," *IEEE Transactions on Acoustics, Speech, Signal Processing*, **ASSP-36**, pp. 1244–1254, 1988.

[21] Lin, J. H. and Kim, Y. T. "Fast algorithm for training stack filters," *IEEE Transactions on Signal Processing*, **42**, pp. 772–781, 1994.

[22] Han, C. C. and Fan, K. C. "Finding of optimal stack filter by graphical searching methods," *IEEE Transactions on Signal Processing*, **45**, pp. 1857–1862, 1997.

# 10

# Intelligent Iris Recognition Using Neural Networks

**Muhammad Sarfraz**
Department of Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

**Mohamed Deriche**
**Muhammad Moinuddin**
**Syed Saad Azhar Ali**
Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

*The objective of this chapter is to present a thorough literature survey on iris recognition work. It also presents a novel approach to iris recognition based on feed-forward neural networks. The features used in this approach are based on the contour of the iris–pupil boundary obtained from radius vector functions and named the 'iris signature'. The proposed technique used is translation, rotation and scale invariant. The classification is performed using two different neural network structures, the Multilayer Feed-forward Neural Network (MFNN) and the Radial Basis Function Neural Network (RBFNN). For feature extraction, the following steps are used:*

1. *The process starts by locating the outer and inner boundaries of the iris.*
2. *The second step is to find the contour of the inner boundary, i.e. the iris–pupil boundary.*
3. *Finally, the iris is represented by 'radius vector functions' and the representation is named the 'iris signature'.*

## 1. Introduction

Nowadays, one of the main threats that IT systems and security environments can face, is the possibility of having intruders in the system. This is normally solved by user identification schemes based on passwords, secret codes or identification cards. Schemes based only on passwords or secret codes

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

can be cracked by intercepting the presentation of such a password, or even by counterfeiting it (via password dictionaries or, in some systems, via brute force attacks). On the other hand, an intruder can attack systems based on identification cards by robbing, copying or simulating them. If the scheme used in a system is based both on a card and a password, the intruder would need to apply more effort to gain entry to the system, and with more advanced technologies, such as smart cards, some vulnerabilities of the system could be avoided.

Technologies that exploit biometrics have the potential for application to the identification and verification of individuals for controlling access to secured areas. Nowadays, a lot of biometric techniques are being developed based on different features and algorithms. This includes recognition of voice, fingerprints, hand shape, retinal scans, handwritten signatures, etc. [1,2]. Unfortunately, from the human factors point of view, these methods are highly invasive. Typically, a person is required to make physical contact with a sensing device (e.g. finger or hand contact) or otherwise take some special action (e.g. recite a specific phonemic sequence). Similarly, there is little potential for covert evaluation. One possible alternative to these methods that has the potential to be less invasive is automated face recognition. As in all pattern recognition problems, the key issue is the relation between inter-class and intra-class variability: objects can be reliably classified only if the variability among different instances of a given class is less than the variability between different classes. Therefore, in the case of face recognition, difficulties arise from the fact that the face is a changeable social organ displaying a variety of expressions, as well as being an active 3D object whose image varies with viewing angle, pose, illumination and age [3,4]. It has been shown that for facial images taken at least one year apart, even the best current algorithms have error rates of 43 %[5] to 50 % [6]. Against this, intra-class (same face) variability is limited because different faces possess the same basic set of features, in the same canonical geometry.

Automated iris recognition is yet another alternative for noninvasive verification and identification of people. Interestingly, the spatial patterns that are apparent in the human iris are highly distinctive to an individual [7,8]. Like the face, the iris is an overt body that is available for remote (i.e. noninvasive) assessment. Unlike the human face, however, the variability in appearance of any one iris might be well enough constrained to make possible an automated recognition system based on currently available machine vision technologies. The possibility that the human iris might be used as a kind of optical fingerprint for personal identification was suggested originally by ophthalmologists. Therefore, the potential of the human iris for such types of problem comes from anatomy of the eye. Some properties of the human iris that enhance its suitability for use in automatic identification include:

1. Its inherent isolation and protection from the external environment, being an internal organ of the eye, behind the cornea and the aqueous humor.
2. The impossibility of surgically modifying it without a high risk of damaging the user's vision.
3. Its physiological response to light, which provides the detection of a dead or plastic iris, avoiding this kind of counterfeit.
4. As a planar object its image is relatively insensitive to angle of illumination, and changes in viewing angle cause only affine transformations; even the nonaffine pattern distortion caused by pupillary dilation is readily reversible.
5. Finally, the ease of localizing eyes in faces, and the distinctive annular shape of the iris, facilitates reliable and precise isolation of this feature and the creation of a size-invariant representation.

Several studies have shown that while the general structure of the iris is genetically determined, the particulars of its minutiae are critically dependent on initial conditions in the embryonic mesoderm from which it develops. Therefore, there are not ever two irises alike, not even for uniovular (identical) twins [9].

The remainder of this chapter is organized as follows: A thorough literature survey is presented in Section 2. Some groundbreaking techniques are discussed in Section 3. Section 4 is the introduction to neural networks. The proposed method is presented in Section 5. In Section 6, simulation results

are presented. A Graphic User Interface (GUI) for the proposed iris recognition algorithm is shown in Section 7. Finally, concluding remarks are given in Section 8.

## 2. Literature Review

Apparently, the first use of iris recognition as a basis for personal identification goes back to efforts to distinguish inmates in the Parisian penal system by visually inspecting their irises, especially the patterning of color [10]. In 1987, the concept of automated iris recognition was proposed by Flom and Safir [11]. It does not appear, however, that this team ever developed and tested a working system. Early work towards actually realizing a system for automated iris recognition was carried out at Los Alamos National Laboratories, CA [12]. Subsequently, two research groups developed and documented prototype iris recognition systems; one by Daugman [9,13,14], and the other by Wildes *et al.* [15,16,17]. The algorithms developed by Daugman [9] are currently being used by many commercial and public entities, including British Telecom, US SandiaLabs, UK National Physical Laboratory, NCR, Oki, IriScan, Iridian, Sensar and Sarnoff. All these reported a false match rate near 0 in all their tests, some of which involved millions of different iris pairings [18]. In this algorithm, the recognition principle is the failure of a test of statistical dependence on iris phase structure encoded by multiscale quadrature wavelets. The combinatorial complexity of this phase information across different people spans about 244 degrees of freedom and generates discrimination entropy of about 3.2 bit/mm$^2$ over the iris, enabling real-time decisions about personal identity with extremely high confidence. The high confidence levels are important because they allow large databases to be searched exhaustively (one-to-many identification mode) without making any false matches, despite so many chances. The results of 2.3 million comparisons among eye images acquired in trials in Britain, the USA and Japan are presented in [18].

Wildes *et al.* [16] in their algorithm exploit a user interface that operators should find easier and less annoying to use. In this approach, the iris is illuminated using a diffuse light source coupled with polarization optics. The algorithm localizes the iris in an image using a histogram-based model fitting approach. The method for representing and matching a given iris image involves registering a captured image to a stored model, filtering with isotropic bandpass filters and subsequent correlation matching.

Some research into automated iris recognition has been carried out in North America [19] and Europe [20]; however, these efforts have not been well documented to date.

Recently, we have also witnessed many new algorithms, including the algorithm by Boles *et al.* [21], which decomposes the iris contour using wavelet transforms, and by Sanchez-Reillo *et al.* [22], where Gabor filters are used. The algorithm developed by de Martin-Roche *et al.* is based on dyadic wavelet transform zero-crossing [23]. A wavelet function that is the first derivative of a cubic spline is used to construct the representation.

Over the past few years there has been considerable interest in the development of neural network-based pattern recognition systems because of their ability to classify nonlinear data. Shah-Hosseini and Safabakhsh proposed a class of neural network named a Time Adaptive Self Organizing MAP (TASOM) [24], that can automatically adjust the learning rate and neighborhood size of each neuron. Each neuron's learning rate is determined by a function of the distance between an input vector and its weight vector. The width of the neighborhood function is updated by a function of the distance between the weight vector of the neuron and the weight vectors of the neighboring neurons. Only one time parameter initialization is sufficient throughout the lifetime of a TASOM to work in stationary and nonstationary environments without retraining. They have tested their algorithm with some standard data sets including the iris plant, breast cancer and BUPA liver disease data. This method, however, has not been applied for iris recognition. In [25], El-Bakry proposed a fast cooperative modular neural network-based iris recognition system. The method was applied earlier to detection of human faces in cluttered scenes [26]. In the iris recognition work, he has used it to

identify automatically human irises in a given image. Since an iris recognition system requires a large database, nonmodular classifiers tend to introduce high internal interference because of the strong coupling among their hidden layer weights [27]. As a result of this, slow learning or overfitting can occur during training. Enlarging the network, increasing the number and quality of training samples, and techniques for avoiding the local minima, will not stretch the learning capabilities of the NN classifier beyond a certain limit, as long as hidden nodes are tightly coupled and hence there is cross talking during learning [28]. So, the modular neural network-based classifier proposed by El-Bakry attempts to reduce the effects of these problems via a divide and conquer approach. Furthermore, faster iris detection is obtained through image decomposition into many subimages, and application of cross correlation in the frequency domain between each subimage and the weights of the hidden layers.

Another neural network for iris recognition was developed by Onsy Abdel Alim and Maha Sharkas [29]. They proposed two feature extraction techniques based on 2D Gabor wavelets and 2D Discrete Cosine Transforms (DCT). They used a Multilayer Perceptron (MLP) NN with back propagation, which consists of one hidden layer and three output neurons to identify three different people. The achieved recognition rate using the DCT coefficients was about 96 %, compared to 92 % obtained using the Gabor coefficients.

In [30], an iris recognition system based on a self-organizing MAP neural network was proposed. A self-organizing MAP neural network has the ability to automatically adjust the learning rate and neighborhood size of each neuron. For iris features, they used only a part of the iris structure. The selected iris structure was then reconstructed into a rectangular format. For classification, a self-organized MAP neural network was trained. The overall accuracy reached was 83 %.

Hybrid pattern recognition systems, especially neuro-fuzzy networks, have gained considerable attention in the past few years. The reason for the popularity of the neuro-fuzzy network is that it combines the advantages of both fuzzy classifiers and neural networks. In [31], the FALCON-ART algorithm is adapted for use in neuro-fuzzy networks and was applied to the iris recognition problem. The average recognition rate obtained was 95.07 %.

Another proposed system in [32] uses a Multilayer Perceptron (MLP) NN as a flexible classifier with modified Co-Occurrence Matrix (COM) derived features. The co-occurrence matrix method is a special orientational second order statistical method introduced in 1974 by R. M. Haralick. In this method, co-occurrences of pixels with particular brightness at a particular distance and orientation are counted. This method is based on the construction of special matrices called co-occurrence matrices, which contain information about the statistical distribution of gray levels in the analyzed image. Then, some statistical features are calculated on the basis of these matrices.

The implemented COM method includes a few changes introduced to the original Haralick method. A Windows-based application called ‘Iris’ was developed that allows presentation of the co-occurrence matrices as images.

Two pieces of preliminary research were performed with co-occurrence matrix-derived features. First, on textures cut directly from eye images to check the chosen method, and second on images obtained using the ‘Iris’ program – to check the ability of collaboration of the whole system. Images of five people were taken into consideration. Then, mean values of the features were calculated on the basis of ten images for each person – for the second piece of research, different smaller numbers of images were used than for the first part. Such obtained features were used to compare irises. In part I of the research, 9 out of 10 iris pairs, and in part II, 4 out of 10 iris pairs, could be distinguished.

## 3. Some Groundbreaking Techniques

Iris recognition is a hot area of research and currently a lot of work is going on. Here, we discuss a few methods which are considered the fundamental work in this field.

## 3.1 Daugman's Method

### 3.1.1 Finding an Iris in an Image

Usually, the pupil center is nasal, and inferior, to the iris center. Its radius can range from 0.1 to 0.8 of the iris radius. So the three parameters defining the pupillary circle must be estimated separately from those of the iris. One effective integrodifferential operator that determines these parameters is,

$$\max_{(r,x_o,y_o)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_o,y_o} \frac{I(x,y)}{2\pi r} \mathrm{d}s \right| \tag{10.1}$$

where $I(x, y)$ is an image containing the eye. The operator searches over the image domain $(x, y)$ for the maximum in the blurred partial derivative with respect to increasing radius $r$, of the normalized contour integral of $I(x, y)$ along a circular arc $\mathrm{d}s$ of radius $r$ and center coordinates $(x_o, y_o)$. The symbol $*$ denotes the convolution and $G_\sigma(r)$ is a smoothing function, such as a Gaussian, of scale $\sigma$. To find the contour of the pupil, the complete operator is used. This operator acts as a circular edge detector, blurred at a scale set by $\sigma$, which searches iteratively for a maximum contour integral derivative with increasing radius at successively finer scales of analysis through the three-parameter space of center coordinates and radius $(r, x_o, y_o)$ defining a path of contour integration. The operator in Equation (10.1) finds both the inner and the outer boundaries of the iris, although the initial search for the outer boundary also incorporates evidence of an interior pupil to improve its robustness, since the outer boundary itself usually has extremely soft contrast when a long-wavelength Near-Infrared (NIR) illumination is used. When the coarse-to-fine iterative searches for both the boundaries have reached single-pixel precision, a similar approach to detect curvilinear edges is used to localize both the upper and lower eyelid boundaries. The path contour integration in Equation (10.1) is changed from circular to accurate, with spline parameters fitted by standard statistical estimation methods to describe optimally the available evidence for each eyelid boundary. The result of all these localization operations is the isolation of iris tissue from other image regions.

### 3.1.2 Iris Feature Encoding by 2D Wavelet Demodulation

Each isolated iris pattern is demodulated to extract its phase information using quadrature 2D Gabor wavelets. It amounts to a patch-wise phase quantization of the iris pattern by identifying in which quadrant of the complex plane each resultant phase lies when a given area of the iris is projected onto complex-valued 2D Gabor wavelets:

$$h\{\mathrm{Re}, \mathrm{Im}\} = \mathrm{sgn}_{\{\mathrm{Re},\mathrm{Im}\}} \int_\rho \int_\phi I(\rho, \phi) e^{-iw(\theta_o - \phi)} e^{-(r_o-\rho)^2/\alpha^2} e^{-(\theta_o-\phi)^2/\beta^2} \rho \mathrm{d}\rho \mathrm{d}\phi \tag{10.2}$$

where $h_{\{\mathrm{Re},\mathrm{Im}\}}$ can be regarded as a complex-valued bit whose real and imaginary parts are either 1 or 0 (sgn) depending on the sign of the 2D integral; $I(\rho, \phi)$ is the raw iris image in a dimensionless polar coordinate system that is size- and translation-invariant, and which also corrects for pupil dilation as explained in a later section. $\alpha$ and $\beta$ are multiscale 2D wavelet size parameters, spanning an 8-fold range from 0.15 mm to 1.2 mm on the iris, $w$ is wavelet frequency, spanning three octaves in inverse propagation to $\beta$; and $(r_o, \theta_o,)$ represent the polar coordinates of each region of iris for which the phasor coordinates $h_{\{\mathrm{Re},\mathrm{Im}\}}$ are computed.

Although 2048 such phase bits are computed for each iris, in a major improvement over the earlier algorithms [9], now an equal number of masking bits are also computed to signify whether any iris region is obscured by eyelids, contains any eyelash occlusions, specular reflections, boundary artifacts

of hard contact lenses, or poor signal-to-noise ratio, and thus should be ignored in the demodulation code as artifacts.

Since the amplitude information is not very discriminating and it depends upon extraneous factors, such as imaging against contrast, illumination and camera gain, phase information is used for recognition.

### 3.1.3 The Test of Statistical Independence: Combinatorics of Phase Sequences

The test of statistical independence can be done using the Boolean XOR applied to the 2048 bit phase vectors that encode the two iris patterns, masked or ANDed by both their corresponding mask bit vectors to prevent non-iris artifacts from influencing iris comparisons. The XOR, denoted by $\otimes$, detects the disagreement between any corresponding pair of bits, while the AND operator, denoted by $\cap$, ensures that the compared bits are both deemed to have been uncorrupted by eyelashes, eyelids, specular reflections or other types of noise. The norms ($||\ ||$) of the resultant bit vector and of the ANDed mask vectors are then measured in order to compute a fractional Hamming Distance (HD) as the measure of the dissimilarity between any two irises having phase code bit vectors denoted by $\{codeA, codeB\}$ and mask bit vectors denoted by $\{maskA, maskB\}$:

$$HD = \frac{\|(codeA \otimes codeB) \cap maskA \cap maskB\|}{\|maskA \cap maskB\|} \tag{10.3}$$

The denominator tallies the total number of phase bits that mattered in iris comparisons after artifacts such as eyelashes and specular reflections were discounted, so the resulting HD is a fractional measure of dissimilarity; 0 would represent a perfect match.

## *3.2 Boles's Method*

In this algorithm, extracting unique features from images of the iris of the human eye and representing these features using the wavelet transform zero-crossings is proposed [21]. This representation is then utilized to recognize individuals from images of the iris of their eyes. The proposed algorithm operates in two stages. The first stage consists of building a one-dimensional representation of the gray-level profiles of the iris, followed by obtaining the wavelet transform zero-crossings of the resulting representation. The second stage is the matching procedure for iris recognition.

### 3.2.1 Extracting Iris Features

The process of information extraction starts by locating the pupil of the eye, which can be done using any edge detection technique. Knowing that it has a circular shape, the edges defining it are connected to form a closed contour. The centroid of the detected pupil is chosen as the reference point for extracting the features of the iris. The gray-level values on the contours of virtual concentric circles, centered at the centroid of the pupil, are recorded and stored in circular buffers. In what follows, for simplicity, one such data set will be used to explain the process and will be referred to as the *iris signature*.

### 3.2.2 Normalization Process

The extracted data from the same iris may be different, even if the diameter of the used virtual circle is kept constant. This is due to the possible variation in the size of the iris in the image as a result of a

change in the camera-to-face distance. For matching purposes, the extracted data must be processed to ensure the accurate location of the used virtual circle and to fix the sample length before constructing the zero-crossing representation. Using the edge-detected image, the maximum diameter of the iris in any image is calculated. In comparing two images, one will be considered as a reference image. The ratio of the maximum diameter of the iris in this image to that of the other image, is also calculated. This ratio is then used to make the virtual circles for extracting the iris features have the same diameter. In other words, the dimensions of the irises in the images will be scaled to have the same constant diameter, regardless of the original size in the images. Furthermore, the extracted information from any of the virtual circles must be normalized to have the same number of data points.

### 3.2.3 Iris Pattern Representation

The next step is to generate a zero-crossing representation from the normalized iris signature. The zero-crossing representation is a stack of inflection points of the iris signature at different resolution levels.

### 3.2.4 Iris Pattern Matching

This process consists of two phases: learning and classification. In the learning phase, the system will construct the model representations based on the irises in noise-free images. A few selected intermediate resolution levels are used in the matching process. According to these representations, a number, $Q$, of intermediate resolution levels will be defined. In practice, the value of $Q$ depends on the total number of resolution levels and is determined based on the magnitude of the zero-crossing representations of the models. A number of intermediate resolution levels, containing most of the energy of the representation, are chosen for use in the matching process. Under uniformly distributed noise, the signal-to-noise ratio at these levels is improved and the effect of noise is reduced significantly. Therefore, choosing a suitable value of $Q$ will make the representation more robust. In the classification phase, the representation of an unknown in an image is constructed with the same normalization value used in constructing the model representations. An unknown signature will be matched with a specific model if the degree of dissimilarity between this signature and that model is the smallest in comparison to other models. The degree of dissimilarity can be calculated based on some dissimilarity functions.

## *3.3 Method of Dyadic Wavelet Transform Zero Crossing*

In 2001, de Martin-Roche *et al.* used dyadic wavelet transform zero crossing for the identification of irises [23]. A wavelet function is the first derivative of a cubic spline. They claimed that this technique is translation, rotation and scale invariant.

### 3.3.1 Localization of Iris Image

First, the image of the eye is converted into grayscale and its histogram is stretched. Then, throughout a gridding process, the center of the iris, as well as the outer boundary, i.e. border between the iris and the sclera, is detected, taking advantage of the circular structure of the latter. Detection is performed by maximizing $D$ which is given by:

$$D = \sum_{m} \sum_{k=1}^{5} \left( I_{n,m} - I_{n-k,m} \right) \tag{10.4}$$

where

$$I_{i,j} = I\left(x_o + i\Delta_\theta \cos\left(j\Delta_\theta\right), y_o + i\Delta_r \sin(j\Delta_\theta)\right) \tag{10.5}$$

where $(x_o, y_o)$ is a point in the grid taken as center, $\Delta_r$ and $\Delta_\theta$ are the increments of ratio and angle, respectively, and $I_{x,y}$ is the image in gray levels.

Once the outer bounds of the iris are detected, everything in the image outside is suppressed, and this same process is performed in order to find the inner boundary, i.e. the frontier between the iris and the pupil. The points inside the last border are also suppressed.

### 3.3.2 Conversion into a 1D Signal

In preprocessing, first of all the diameter sizes of all the irises are made the same, regardless of their original sizes. Then, the centroid of the detected pupil is chosen as the reference point for extracting the features of the irises. In the next step, the gray-level values on the contours of a virtual circle, which is centered at the centroid of the pupil, are recorded. This data set is 256 bits in length and it is named the *iris signature*.

### 3.3.3 Feature Extraction

For iris identification, some features are extracted from the iris signatures which are based on dyadic wavelet transforms.

***Discrete Dyadic Wavelet Transform***

Let $\varphi(x)$ denote the dilation of a function $\varphi$ by a factor $s$:

$$\varphi(x) = \frac{1}{s}\varphi\left(\frac{x}{s}\right) \tag{10.6}$$

The continuous wavelet transform of a continuous function $f(x)$ at the scale $s$ and position $x$ can be written in the following convolutional form:

$$W_s f(x) = f^* \varphi_s(x) \tag{10.7}$$

Let $\hat{\varphi}(w)$ be the Fourier transform of $\hat{\varphi}(x)$, then, according to Grossmann and Morlet [33], if $\hat{\varphi}(w)$ satisfies:

$$\int_{-\infty}^{\infty} \frac{|\hat{\varphi}(w)|^2}{|w|}\,\mathrm{d}w = C_\varphi < \infty \tag{10.8}$$

then $f(x)$ can be reconstructed from its wavelet transform. The above condition is called the admissibility condition and it implies that

$$\hat{\varphi}(0) = 0 \tag{10.9}$$

The term 'wavelet' is used for any function from $L^2(\mathcal{R})$ which satisfies the admissibility condition. If the condition

$$\sum_{-\infty}^{+\infty} \left|\hat{\varphi}\left(2^j w\right)\right|^2 = 1 \tag{10.10}$$

is satisfied then the scale parameter can be sampled along the dyadic sequence $\{2^j\}_{j\in Z}$ while preserving the reconstruction property. Any wavelet satisfying Equation (10.10) is called a dyadic wavelet and the sequence of functions:

$$\left\{W_{2j}f\,(x)\right\}_{j\in Z} \tag{10.11}$$

is called the *dyadic wavelet transform*. For normalization purposes, it is assumed that the finest scale is 1 and the largest scale is $2^J$. Let $\varphi(x)$ be a scaling function whose Fourier transform satisfies:

$$\left|\hat{\phi}\,(w)\right|^2 = \sum_{j=1}^{\infty}\left|\hat{\phi}\left(2^J w\right)\right|^2 \tag{10.12}$$

Now define a smoothing function $S_{2j}$ as:

$$S_{2j}f\,(x) = f^{*}\phi_{2j}\,(x) \tag{10.13}$$

where $\phi_{2j}\,(x) = \dfrac{1}{2j}\phi\left(\dfrac{x}{2j}\right)$. It can be proven that any discrete input signal $f^d$ can be written as $f^d = S_1 f[n]$. Now define a function $S^d_{2J}f$ as:

$$S^d_{2^J}f = \left(S_{2^J}f\,[n+w]\right)_{1\le n\le N} \tag{10.14}$$

and

$$W^d_{2^J}f = \left(W_{2^J}f\,[n+w]\right)_{1\le n\le N} \tag{10.15}$$

where $N$ is the length of $f^d$ and $w$ is a sampling shift that depends only on $\varphi$. Now, the Discrete Dyadic Wavelet Transform (DDWT) of $f^d = S_1 f[n]$ can be defined as the sequence of the discrete signals:

$$\left\{\left(W^d_{2^J}f\right)_{1\le j\le J}, S^d_{2^J}f\right\} \tag{10.16}$$

### 3.3.4 Zero-Crossing Representation of Iris Pattern

It is known that one can obtain the position of multiscale sharp variation points from the zero crossing of the signal convolved with the Laplacian of a Gaussian. If $f \in L^2(\mathcal{R})$ and $\{W_2^J f(x)\}j_{\in Z}$ is its dyadic wavelet transform, then for any pair of consecutive zero crossings of $W_{2j}$ whose abscissae are respectively $(z_{n-1}; z_n)$, the following can be recorded:

$$e_n = \int_{z_{n-1}}^{z_n} W_{2j}f\,\mathrm{d}x \tag{10.17}$$

For any function $W_2^j f$, the position of the zero crossings $(z_n)n \in Z$ can be represented by a piece-wise constant function:

$$Z_{2^j}f = \frac{e_n}{z_n - z_{n-1}} \tag{10.18}$$

Therefore, in order to obtain a complete and stable representation, the zero crossings of the dyadic wavelet transform of the iris signatures are considered, and the value of the wavelet transform between two consecutive zero crossings are also recorded.

### 3.3.5 Classification and Verification

For classification, comparison is made between different users' iris signatures based on the following three methods:

1. The Euclidean distance, which is defined as:

$$d_{\mathrm{E}}(y,p)=\sqrt{\sum_{i=1}^{L}(y_i-p_i)^2} \tag{10.19}$$

   where $L$ is the dimension of the feature vector, $y_i$ is the $i$th component of the sample feature vector and $p_i$ is the $i$th component of the template feature vector.
2. The binary Hamming distance, which is defined as:

$$d_{\mathrm{H}}^{b}(y,p)=\frac{1}{L}\sum_{i=1}^{L}y_i\oplus p_i \tag{10.20}$$

   The Hamming distance measures not the difference between the components of the feature vectors, but only the number of components that differ in value. Assuming that the feature components follow a Gaussian distribution, the number of components of the feature vector falling outside the area defined by the template parameters is obtained by the following Hamming distance:

$$d_{\mathrm{H}}(y,p)=\#\{i,\ldots,L\}:|y_i-p_i^m|>p_i^v \tag{10.21}$$

   where $p_i^m$ is the mean of the $i$th component and $p_i^v$ is the factor of the standard deviation of the $i$th component.
3. The third method uses the distance that directly relates to the zero-crossing representation of a 1D signal. The dissimilarity function, which compares the unknown object $y$ and candidate model $p$ at a particular resolution level $j$, is given by:

$$d_j(y,p)=\frac{\sum_{i=1}^{R_j}\left\{\left[\mu_j(r)\right]_y\left[\rho_j(r)\right]_y-\Gamma\left[\mu_j(r)\right]_p\left[\rho_j(r)\right]_p\right\}}{\Gamma\sum_{i=1}^{R_j}\left\{\left|\left[\mu_j(r)\right]_y\left[\rho_j(r)\right]_y\right|\left|\left[\mu_j(r)\right]_p\left[\rho_j(r)\right]_p\right|\right\}} \tag{10.22}$$

   where $Z_{jp}$ is the zero-crossing representation whose imaginary and real parts are $[\rho_j(r)]_p$ and $[\mu_j(r)]_p$, respectively, and $\Gamma$ is the scale factor and equals the ratio between the average radius of the candidate model and that of the unknown object.

## 4. Neural Networks

In this work, two different structures of neural network are used, multilayer feed-forward neural networks and radial basis function neural networks. A brief overview of each network is given below.

### *4.1 Multilayer Feed-forward Neural Networks (MFNNs)*

Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers', where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer', where the answer is output as shown in Figure 10.1. Most ANNs contain some form of 'learning rule', which modifies the weights of the connections according to the input patterns with which it is presented. In a sense, ANNs learn by example, as do their biological counterparts; a child learns to recognize flowers from examples of flowers.

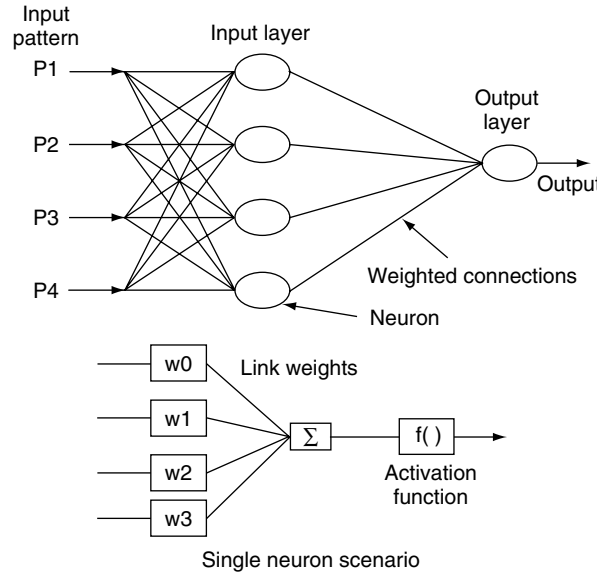



**Figure 10.1** Structure of an MFNN.

### 4.1.1 Learning Algorithm

The Back Propagation (BP) algorithm is the most widely used learning procedure for supervised neural nets. Before beginning training, some small random numbers are usually used to initialize each weight on each connection. BP requires preexisting training patterns, and involves a forward propagation step followed by a backward propagation step. The forward propagation step begins by sending the input signals through the nodes of each layer. A nonlinear activation function, called the sigmoid function, is usually used at each node for the transformation of the incoming signals into an output signal. This process repeats until the signals reach the output layer and an output vector is calculated. The backward propagation step calculates the error vector by comparing the calculated and target outputs. New sets of weights are iteratively updated until an overall minimum error is reached. The Mean-Square Error (MSE) is usually used as a measure of the global error, which can be defined as:

$$MSE = \sum_{j=1}^{N_o} \left(d_j - y_j\right)^2 \tag{10.23}$$

where $N_o$ is the number of output nodes and $y$ and $d$ are the output and target signals respectively. Weights are updated as follows:

$$\underbrace{w_{ji}^{(l)}(n+1)}_{\text{New weights}} = \underbrace{w_{ji}^{(l)}(n)}_{\text{Old weights}} + \mu \underbrace{\delta_j^{(l)}(n)}_{\text{Local gradient}} y_i^{(l-1)}(n) + \alpha \underbrace{\Delta w_{ji}^{(l)}(n-1)}_{\text{Old change in weights}} \tag{10.24}$$

where

$$\delta_j^{(l)}(n) = \begin{cases} \left(d_j(n) - y_j^{(L)}(n)\right) \phi_j'\left(v_j^{(L)}(n)\right), & \text{for neuron } j \text{ in output layer} L \\ \phi_j'\left(v_j^{(L)}(n)\right) \sum_k \delta_k^{(l+1)}(n)\, w_{kj}^{(l+1)}(n), & \text{for neuron } j \text{ in hidden layer } l \end{cases} \tag{10.25}$$

Performance of the trained network can be evaluated by some simple statistical functions such as recognition rate (i.e. the percentage of the total number of correctly classified outcomes over the number of sample points, or simply %Reco) and the mean-square error. If the error value on the test data set begins to increase, training is halted and the results are examined to determine whether they are acceptable. If the results are unacceptable, then it is possible to retrain the network, by either modifying some network parameters (e.g. the seed value for the random number generator and the number of nodes in the middle layer), or increasing or decreasing the variations present in the training patterns. Once an acceptable error value is obtained during the test stage, the network is ready for solving real problems, such as classification of input signals (e.g. well log data) into discrete classes (e.g. lithofacies) and prediction of property values (e.g. porosity and permeability) using some input signals (e.g. well log data).

### 4.1.2 Activation Function

An activation function is used to transform the activation level of a unit (neuron) into an output signal. Typically, activation functions have a 'squashing' effect, i.e. they limit the permissible output range to some finite values.

$$z_k = \sum_{j=1}^{m} w_{kj} x_j \tag{10.26}$$

and

$$y = \phi(z_k) = \phi\left\lfloor w_{kj} x_j \right\rfloor \tag{10.27}$$

where $x_1, x_2, \ldots,$ are the input signals; $w_{k1}, w_{k2} \ldots,$ are synaptic weights of neuron $k$, and $\phi$ is the activation function:

$$\phi(z, \lambda) = \tan sig(z, \lambda) = \frac{1 - e^{-\lambda z}}{1 + e^{-\lambda z}}, \lambda \in R^{+} \tag{10.28}$$

$$\phi(z, \lambda) = \log sig(z, \lambda) = \frac{1}{1 + e^{-\lambda z}}, \lambda \in R^{+} \tag{10.29}$$

$$\phi(z, \alpha, \lambda) = \sin mul(z, \alpha, \lambda) = z + \frac{\alpha}{\lambda \pi} \sin(\lambda \pi z), \lambda \in Z^{+}, 0 < \alpha < 1 \tag{10.30}$$

$$\phi(z) = piecewise(z) = \begin{cases} -1, & \text{if } z < -1 \\ z, & \text{if } -1 \le z \le 1 \\ 1, & \text{if } z > 1 \end{cases} \tag{10.31}$$

Activation functions for the hidden units are needed to introduce nonlinearity into the networks. Nonlinearity makes the multilayer networks more powerful. For back propagation, learning the activation function must be differentiable. The more common activation functions are sigmoidals (log and tangent), piecewise-linear and sinusoidal functions. For hidden units, sigmoidal functions are usually preferable. With sigmoid units, a very small change in the weights will usually produce a change in the outputs, which makes it possible to depict whether that change in weights is good or bad.

## *4.2 Radial Basis Function Neural Networks (RBFNNs)*

An RBFNN is a type of feed-forward network. In these networks, the learning involves only one layer. This results in a reduction in the training time and complexity in comparison with the MFNN. A SISO

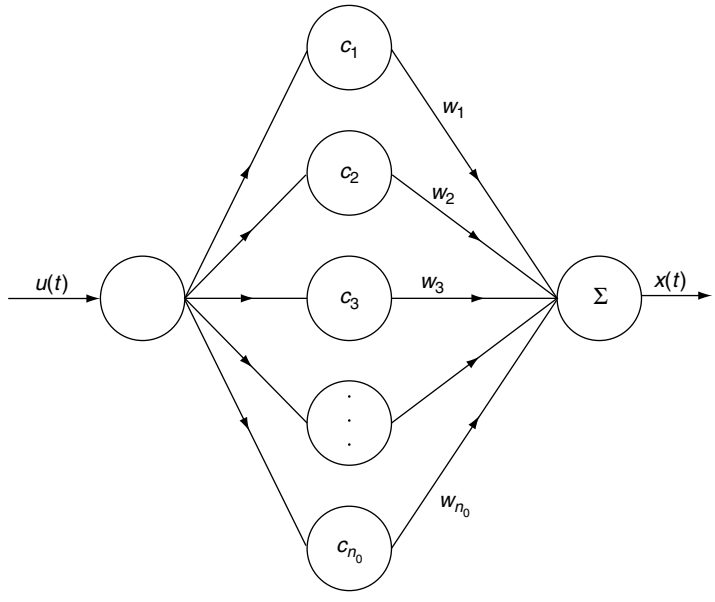



**Figure 10.2** A general RBFNN.

RBFNN is shown in Figure 10.2. It consists of an input node $u(t)$, a hidden layer with $n_0$ neurons and an output node $x(t)$.

Each of the input nodes is connected to all the nodes in the hidden layer through unity weights (direct connections). While each of the hidden layer nodes is connected to the output node through some weights $w_1, \ldots, w_{n_0}$, where $n_0$ is the number of hidden layer nodes (neurons). Each neuron finds the distance, normally by applying the Euclidean norm, between the input and neuron's center and passes the resulting scalar through a nonlinear function. The output of the hidden neuron is given by $\phi(\|u(t) - c_i\|)$, $u(t)$ is the input, $c_i$ is the center of the $i$th hidden layer node, where $i = 1, 2, \ldots, n_0$, and $\phi(\cdot)$ is the nonlinear basis function. Normally, this function is modeled by a Gaussian function of width $\beta$. The output $x(t)$ is a weighted sum of the outputs for different hidden layers:

$$x(t) = W\Phi(t) \tag{10.32}$$

$$x(t) = \sum_{i=1}^{n_0} w_i \phi(\|u(t) - c_i\|) \tag{10.33}$$

where

$$W = [w_1 w_2 \cdots w_{n_0}] \tag{10.34}$$

and

$$\Phi(t) = \left[\|u(t) - c_1\| \, \|u(t) - c_2\| \cdots \|u(t) - c_{n_0}\|\right]^{\mathrm{T}} \tag{10.35}$$

### 4.2.1 Learning Algorithm

The weights of RBFNNs are updated by minimizing the performance index $I$ given by:

$$I = \frac{1}{2} e^2(t) \tag{10.36}$$

where $e(t)$ is the classification error. According to the LMS principle, the weights of the RBFNN at the $k$th iteration should be updated in the negative direction of the gradient as:

$$W_{k+1} = W_k - \alpha \frac{\partial I}{\partial W_k} \tag{10.37}$$

where $\alpha$ is the learning parameter. The partial derivatives for the weights are derived as follows:

$$\begin{aligned} \frac{\partial I}{\partial W} &= \frac{1}{2}\frac{\partial e^2(t)}{\partial W} \\ &= \frac{1}{2}2e(t)\frac{\partial}{\partial W}(y(t) - \hat{y}(t)) \\ &= e(t)\frac{\partial}{\partial W}(y(t) - W\Phi(t)) \\ \frac{\partial I}{\partial W} &= -e(t)\Phi(t) \end{aligned} \tag{10.38}$$

The gradient in Equation (10.38) is used to find the updated weights using Equation (10.37). The final weight update equation will take the form:

$$W_{k+1} = W_k + \alpha e(t)\Phi(t) \tag{10.39}$$

## 5. Proposed Method

In this work, the technique used for extracting the iris features is translation, rotation and scale invariant. The main steps of the algorithm are:

**1. Feature Extraction**

(i) The process starts by locating the outer and inner boundaries of the iris.
(ii) The second step is to find the contour of the inner boundary, i.e. the iris–pupil boundary.
(iii) Finally, the iris is represented by radius vector functions and the representation is named the 'iris signature'.

**2. Classification**. Two neural network techniques have been implemented:

(i) Classification using a Multilayer Feed-forward Neural Network (MFNN).
(ii) Classification using a Radial Basis Function Neural Network (RBFNN).

In the next sections, these steps are discussed in detail.

### *5.1 Localizing the Iris*

#### 5.1.1 Edge Detection

The first step is to locate the iris outer boundary, i.e. the border between the iris and the sclera. This is performed by edge detection on the grayscale iris image. In Matlab, the command 'edge' uses six different methods of edge detection which are as follows:

1. The Sobel method. This finds edges using the Sobel approximation to the derivative. It returns edges at those points where the gradient of the image is maximal.
2. The Prewitt method. This finds edges using the Prewitt approximation to the derivative. It returns edges at those points where the gradient of the image is maximal.
3. The Roberts method. This finds edges using the Roberts approximation to the derivative. It returns edges at those points where the gradient of the image is maximal.

4. The Laplacian of Gaussian method. This finds edges by looking for zero crossings after filtering the image with a Laplacian of Gaussian filter.
5. The Zero-cross method. This finds edges by looking for zero crossings after filtering the image with a filter you specify.
6. The Canny method. This finds edges by finding local maxima of the gradient of the image. The gradient is calculated using the derivative of a Gaussian filter. The method uses two thresholds to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges. This method is robust to additive noise, and able to detect 'true' weak edges.

In this work, the edges of the irises are detected using the 'Canny method' because of its better ability at edge detection. Figures 10.3 and 10.4 show the original and edge images, respectively.

## *5.2 Finding the Contour*

The second step is to find the inner boundary of the iris, i.e. the frontier between the iris and the pupil. For this, the centroid of the detected pupil is chosen as the reference point $(x_c, y_c)$. Given that the edges are represented by binary ones, the top and bottom extreme points (i.e. $(x_c, y_{min})$ and $(x_c, y_{max})$) of the inner boundary are first detected from this reference point by searching for the first one. Once these extreme points are detected, a similar search for the first one is made for all the points on the left $(x < x_c)$, as well as on the right $(x > x_c)$ of the reference point. Thus, all the boundary points are stored in a one-dimensional vector.

## *5.3 Feature Extraction*

One of the most important methods of characterizing a figure is the representation of its contour by a function. This function can be any of the following:

1. The cross-section function (for symmetric figures).
2. The radius vector function (for star-shaped figures).
3. The supports function (mainly for convex figures).

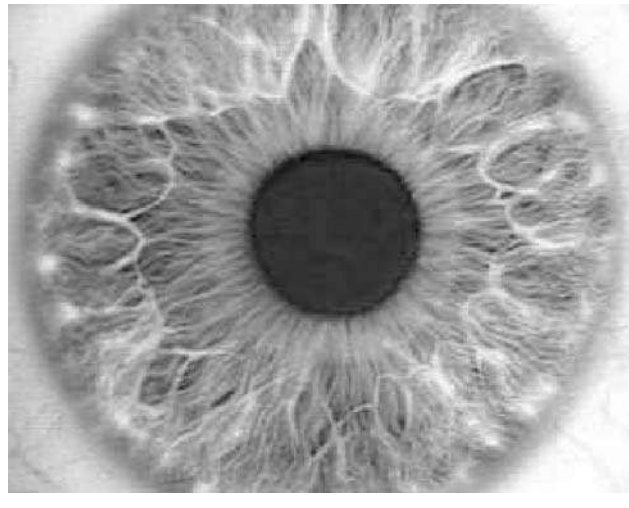


**Figure 10.3** Image of a sample iris.

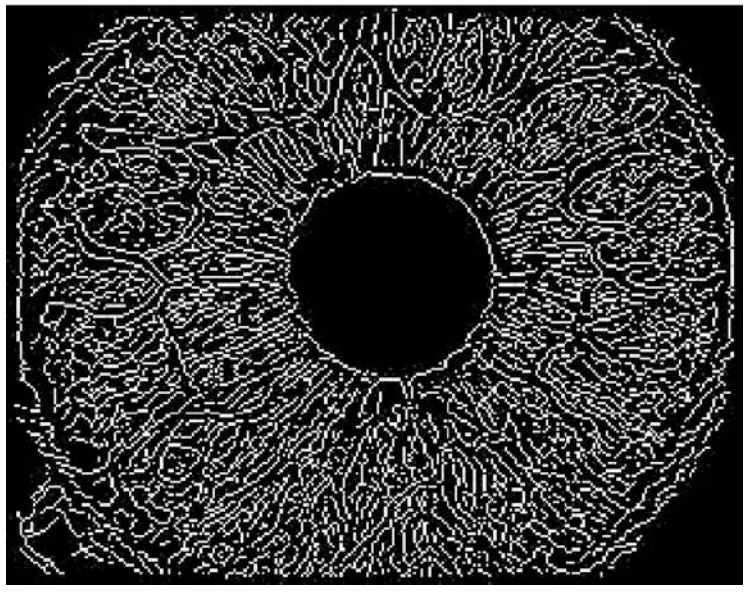


**Figure 10.4** Edges of the sample iris.

Since the inner boundary of the iris is not necessarily symmetric and convex, so the radius vector is an obvious choice for finding the contour. This vector is explained below.

### 5.3.1 Radius Vector Functions

The contour of a figure $F$ is described by the radius vector function. So a reference point must be chosen inside $F$. This may be, for example, the center of gravity, the center of the smallest disc that completely contains the figure, or a biologically important point. The figure is then translated such that this point lies at the origin $o$. Let the translated figure be denoted by $X$. It is necessary that $X$ must be star-shaped with respect to $o$. This means that for any contour point $x$ of $X$, the whole line segment from $o$ to $x$ must be inside $X$. Figure 10.5 shows a star-shaped set. Note that if the star-shaped set is not satisfied because of small irregularities in the contour, the figure may be translated into a star-shaped one by presmoothing it.

The *radius vector function* $r_X(\varphi)$ depends on the angle $\varphi$ made by the line emanating from $o$ with the $x$-axis (see Figure 10.5). The quantity $r_X(\varphi)$ is equal to the length of the line segment from $o$ to the contour point $x$ in which the $\varphi$-ray intersects the boundary. This function characterizes the contour $X$ precisely. That is, $X$ can be uniquely reconstructed if we are given $r_X(\varphi)$. It is obvious that:

$$r_{\lambda X}(\varphi) = \lambda_{rX}(\varphi), \qquad X \subset Y \Rightarrow r_X(\varphi) < r_Y(\varphi) \tag{10.40}$$

The map $X \to r_X$ transforms figures into elements of a function space. If figures have the property that the radius vector function is continuous, then the Banach space $C[0, 2\pi]$ is suitable.

In our scheme, the radius vector method is used to represent the iris contour. Upsampling and downsampling processes are used to normalize the length of the obtained contour. Finally, the 1D feature vector obtained is named the 'iris signature'. A sample of an iris signature is shown in Figure 10.6.

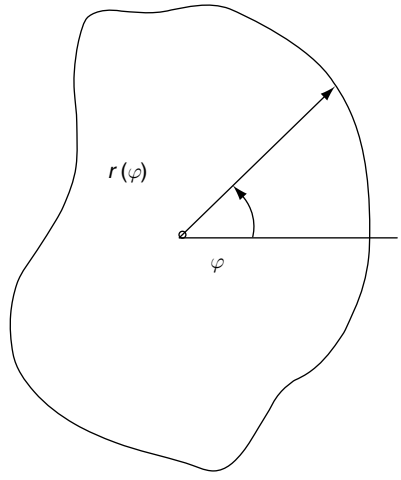



**Figure 10.5** A star-shaped figure. Each ray starting at o forms only one line segment in it. The length of line segment for a ray of direction $\varphi(0 \leq \varphi \leq 2\pi)$ is denoted by $r_X(\varphi)$. This is the radius vector function.

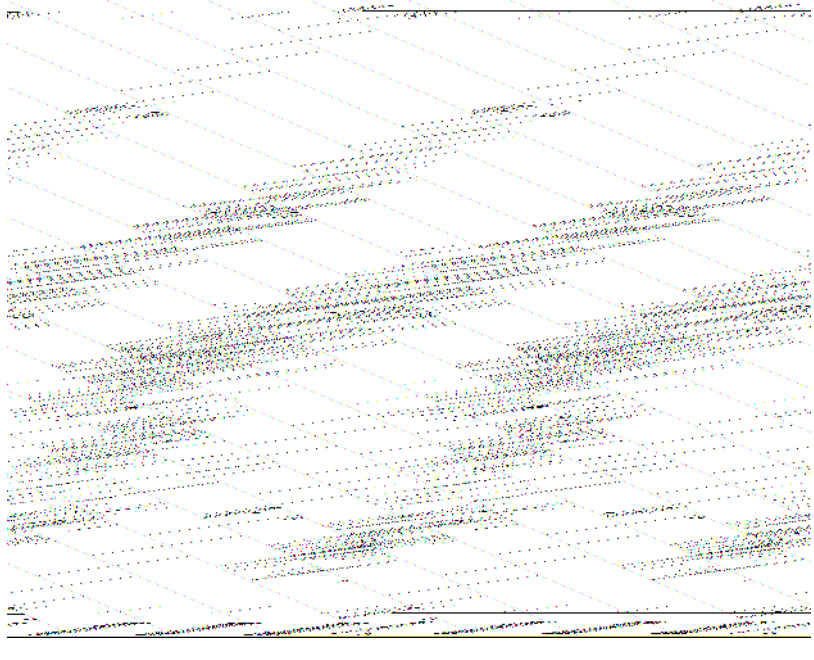


**Figure 10.6** A sample of an iris signature.

### 5.4 Iris Pattern Recognition

The extracted iris signatures obtained above are used for classification. For this purpose, both the MFNN and the RBFNN are used. The process consists of two phases:

1. Learning.
2. Classification.

The results are discussed below.

## 6. Experimental Results

Simulations were performed using ten different iris samples obtained by Daugman. An MFNN and an RBFNN were used for iris recognition.

### 6.1 Results for an MFNN

We used an MFNN consisting of a single hidden layer with 30 neurons and an output layer with ten neurons. The back propagation algorithm was used to update the weights. For training of the NN, noisy iris samples were generated by adding Gaussian noise at 10 dB and 20 dB SNRs. Training was performed on 70 % of the data. Then, testing was carried out on the remaining data at different SNRs. The SNR represents the quality of the acquired image, which may be corrupted with effects due to environment, quantization or digitization of the image. Tables 10.1 and 10.2 represent the classification results for different classes at 10 dB and 20 dB SNRs, respectively. In the tables, 'Tr' and 'Id' stand for true and identified classes, respectively. Notice that an 11th class is displayed in the tables. This class represents the case when a given iris is not assigned to any of the ten classes.

### 6.2 Results for an RBFNN

Using the same set-up as above, the RBFNN was then used for classification. A network with ten centers was designed and found to be optimal for our problem. Having the same nonlinear approximation

**Table 10.1** Percentage recognition for MFNN with 10 dB SNR.

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C1 | 97 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| C2 | 0 | 95.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4.4 |
| C3 | 0 | 0 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| C4 | 0 | 0 | 0 | 95.5 | 0 | 0 | 0 | 0 | 0 | 0 | 4.5 |
| C5 | 0 | 0 | 0 | 0 | 94.8 | 0 | 0 | 0 | 0 | 0 | 5.2 |
| C6 | 0 | 0 | 0 | 0 | 0 | 97.3 | 0 | 0 | 0 | 0 | 2.7 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 0 | 2 |
| C8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 96.8 | 0 | 0 | 3.2 |
| C9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 97 | 0 | 3 |
| C10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 99 | 1 |

**Table 10.2** Percentage recognition for MFNN with 30 dB SNR.

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C1 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| C2 | 0 | 99 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| C3 | 0 | 0 | 99.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.5 |
| C4 | 0 | 0 | 0 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C5 | 0 | 0 | 0 | 0 | 99 | 0 | 0 | 0 | 0 | 0 | 1 |
| C6 | 0 | 0 | 0 | 0 | 0 | 98.6 | 0 | 0 | 0 | 0 | 0.4 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 99 | 0 | 0 | 0 | 1 |
| C8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 0 | 0 | 0 |
| C9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 99.2 | 0 | 0.8 |
| C10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 0 |

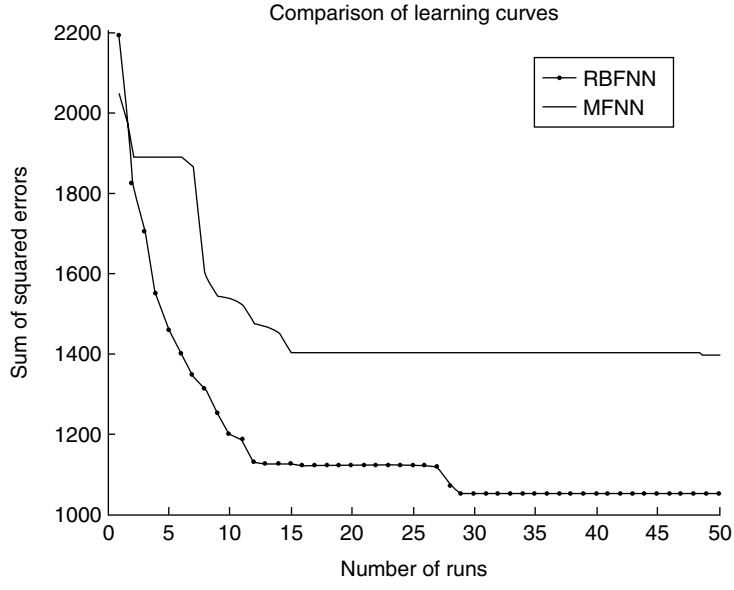



**Figure 10.7** Comparison of learning curves for an RBFNN and an MFNN.

capability as the MFNN, similar results were obtained at different SNRs. The distinguishing feature of the RBFNN is obviously its lower computational complexity, and hence the efficiency in time compared to the MFNN, as can be seen in Figure 10.7. The simulation time was found to be at least half that of the MFNN. The simplicity of the architecture is also notable with just one layer and ten neurons. Classification behavior of the RBFNN is shown in Tables 10.3 and 10.4. The average classification obtained by both the networks was 96.9 % at 10 dB SNR. This accuracy increases to about 100 % at 30 dB SNR or more.

**Table 10.3** Percentage recognition for RBFNN with 10 dB SNR.

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C1 | 95.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4.5 |
| C2 | 0 | 95 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 |
| C3 | 0 | 0 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| C4 | 0 | 0 | 0 | 97 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| C5 | 0 | 0 | 0 | 0 | 95.8 | 0 | 0 | 0 | 0 | 0 | 4.2 |
| C6 | 0 | 0 | 0 | 0 | 0 | 97.3 | 0 | 0 | 0 | 0 | 2.7 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 0 | 2 |
| C8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 96 | 0 | 0 | 4 |
| C9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 97.5 | 0 | 2.5 |
| C10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98.8 | 1.2 |

**Table 10.4** Percentage recognition for RBFNN with 30 dB SNR.

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C1 | 99 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| C2 | 0 | 98.7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1.3 |
| C3 | 0 | 0 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C4 | 0 | 0 | 0 | 99.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0.5 |
| C5 | 0 | 0 | 0 | 0 | 99.2 | 0 | 0 | 0 | 0 | 0 | 0.8 |
| C6 | 0 | 0 | 0 | 0 | 0 | 100 | 0 | 0 | 0 | 0 | 0 |
| C7 | 0 | 0 | 0 | 0 | 0 | 0 | 98.8 | 0 | 0 | 0 | 1.2 |
| C8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 2 |
| C9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 99.2 | 0 | 0.8 |
| C10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 0 |

## 7. Graphic User Interface (GUI)

The GUI for the program is made in Matlab, which shows all the steps of the procedure. In the first step, a window appears showing the database of all the iris patterns, as shown in Figure 10.8.

The user can select any of the iris patterns for testing the recognition performance by clicking on the iris image. As a result of that click, the program starts in the background, first preprocessing the iris data. Preprocessing gives the following in consecutive steps:

1. A grayscale image.
2. An edge-detected image.
3. The iris signature.

All these steps are presented in one window, as shown in Figure 10.9. The iris signature so obtained is the input to the iris classifier. The result of this classification is shown in terms of a recovered iris image. The user can continue the process by clicking the label ‘Press Any Key to Continue’.

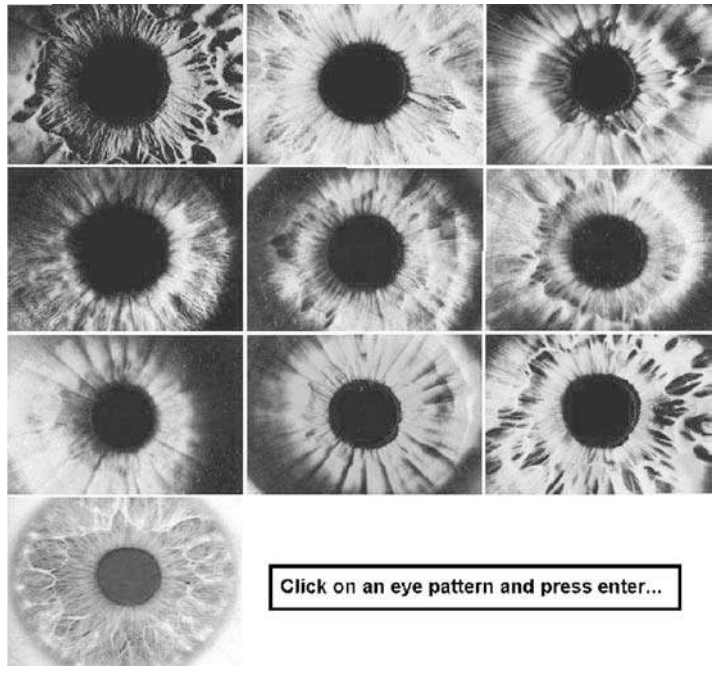



**Figure 10.8** The GUI's main window showing all of the iris patterns.

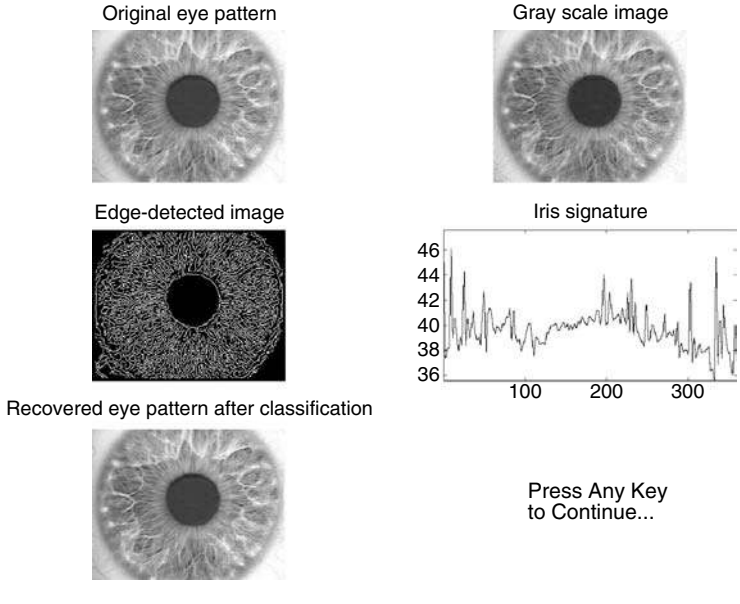



**Figure 10.9** Window showing all the steps of iris recognition.

## 8. Concluding Remarks

In this chapter, the problem of iris recognition has been studied in detail. Several past techniques were discussed. A novel technique is proposed for representing the iris using only its inner boundary. The classification is performed using both an MFNN and an RBFNN. The Neural Pattern Recognition (NeurPR) approach is found to be a promising one for iris recognition. The MFNN and RBFNN gave similar recognition rates, although the RBFNN is preferred in terms of computation load. Our results are currently being expanded to larger databases, such as the Chinese Iris Database.

## References

[1] Miller, B. "Vital signs of identity," *IEEE Spectrum*, **31**, pp. 22–30, 1994.
[2] Jain, A. K., Bolle, R., Pankanti, S. *et al.*, *Biometrics: Personal Identification in Networked Society*, Kluwer Academic Publishers, 1999.
[3] Chellappa, R., Wilson, C. L. and Sirohey, S. "Human and machine recognition of faces: A survey," *Proceedings of the IEEE*, **83**, pp. 705–740, 1995.
[4] Samal, A. and Lyengar, P. A. "Automatic recognition and analysis of human faces and facial expressions: A survey," *Pattern Recognition*, **25**, pp. 65–77, 1992.
[5] Philips, P. J., Moon, H., Rizvi, S. A. and Rauss, P. J. "The FERET evaluation methodology for face recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(10), pp. 1040–1104, 2000.
[6] Pentland, A. and Choudhury, T. "Face recognition for smart environments," *Computer*, **33**(2), pp. 50–55, 2000.
[7] Adler, F. H. *Philosiophy of the Eye*, Mosby, St. Louis, MO, 1965.
[8] Kroeber, A. L. *Anthropology*, Harcourt Brace Jovanovich, New York, 1948.
[9] Dougman, J. "High Confidence visual recognition by test of statistical independence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**, pp. 1148–1161, 1993.
[10] Bertillon, A. "La couleur de 1' iris," *Rev. Sci.*, **36**(3), pp. 65–73, 1885.
[11] Flom, L. and Safir, A. *Iris recognition system*, U.S. Patent 4641349, 1987.
[12] Johnson, R. G. *Can iris patterns be used to identify people*, Los Alamos National Laboratory, CA, Chemical and Laser Sciences Division, Rep. LA-12331-PR, 1991.
[13] Daugman, J. *Biometric signature security system*, Harvard University, Cambridge, MA, Tech. Rep., 1990.
[14] Daugman, J. "High Confidence personal identification by rapid video analysis of iris texture," *Proceedings of IEEE International Carnahan Conference on Security Technology*, pp. 1–11, 1992.
[15] Wildes, R. P., Asmuth, J. C., Green, G. L., Hsu, S. C., Kolzynski, R. J., Matey, J. R. and McBride, S. E. "A machine vision system for iris recognition," *Machines Vision Applications*, **9**, pp. 1–8, 1996.
[16] Wildes, R. P., Asmuth, J. C., Green, G. L., Hsu, S. C., Kolzynski, R. J., Matey, J. R. and McBride, S. E. "A system for automated iris recognition," *Proceedings of IEEE Workshop on Applications of Computer Vision*, Sarasota FL, pp. 121–128, 1994.
[17] Wildes, R. P., Asmuth, J. C., Green, G. L., Hsu, S. C., Kolzynski, R. J., Matey, J. R. and McBride, S. E. *Iris recognition for security access control: Final Report*, National Information Display Laboratory, Princeton, NJ, Tech. Rep., 1992.
[18] Daugman, J. "High Confidence Recognition of Persons by Iris Patterns," *IEEE 35th International Conference on Security Technology*, pp. 254–263, 2001.
[19] Stotz, J. "Personal Communication," 1994.
[20] Malickas, A. "Personal Communication," 1994.
[21] Boles, W. W. and Boashash, B. "A human identification technique using images of the iris and wavelet transform," *IEEE Transactions on Signal Processing*, **46**, pp. 1185–1188, 1998.
[22] Sanchez-Reillo, R. and Sanchez Avila, C. "Processing of the human iris pattern," *Proceedings of IPMU 2000 (8th International Conference on Information Processing and Management of Uncertainity in Knowledge-Based Systems)*, **3**, pp. 653–656, 2000.
[23] de Martin-Roche, D., Sanchez Avila, C. and Sanchez-Reillo, R. "Iris Recognition for Biometric Identification using Dyadic Wavelet Transform Zero-Crossing," *IEEE 35th International Conference on Security Technology*, pp. 272–277, 2001.

[24] Shah-Hosseini, H. and Safabakhsh, R. "Pattern classification by the time adaptive selforganizing MAP," *ICECS 2000, The 7th IEEE International Conference on Electronics, Circuits and Systems*, pp. 495–498, 2000.

[25] El-Bakry, H. M. "Human Iris Detection Using Fast Cooperative Modular Neural Nets," *Proceedings of IJCNN '01, International Joint Conference on Neural Networks*, pp. 577–582, 2001.

[26] El-Bakry, H. M., Abo-Elsoud, M. A. and Kamel, M. S. "Fast Modular Neural Networks for Human Face Detection," *Proceedings of IJCNN, International Joint Conference on Neural Networks*, pp. 320–324, 2000.

[27] Jacobs, R., Jordan, M. and Barto, A. "Task Decomposition through Competition in a Modular Connectionist Architecture: The what and where vision tasks," *Neural computation*, **3**, pp. 79–87, 1991.

[28] Waibel, A. "Modular construction of Time Delay Neural Networks for Speech Recognition," *Neural Computing*, **1**, pp. 39–46, 1989.

[29] Alim, O. A. and Sharkas, M. "Texture Classification of the Human Iris using Artificial Neural Networks," *IEEE MELECON*, May 2002, Cairo, Egypt, pp. 580–583, 2002.

[30] Liam, L. W., Chekima, A., Fan, L. C. and Dargham, J. A. "Iris Recognition using Self-Organizing Neural Network," *Proceedings of the 2002 Student Conference on Research and Development*, Shah Alam, Malaysia, pp. 169–172, 2002.

[31] Van Vuuren, P. A. and Hoffman, A. J. "Improved rule generation for a neuro-fuzzy network," *Neural computing IEEE*, pp. 2845–2850, 2000.

[32] Szewczyk, R., Jablonski, P., Kulesza, Z., Napieralski, A., Cabestany, J. and Moreno, M. "Automatic People Identification on the Basis of Iris-Pattern Extraction Features and Classification," *Proceedings of the 23rd International Conference on Microelectronics (MIEL 2002)*, Yugoslavia, pp. 12–15, 2002.

[33] Grossmann, A., Holschneider, M., Kronland-Martinet, R. and Morlet, J. "Detection of abrupt changes in sound signals with the help of wavelet transforms," *Advances in Electronics and Electron Physics*, Suppl. 19, Inverse problems, Academic Press, 1987.

# 11

# Pose-invariant Face Recognition Using Subspace Techniques

**Mohamed Deriche**
**Mohammed Aleemuddin**
Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Kingdom of Saudi Arabia

*Biometrics refers to the growing technology of automated methods in recognizing individuals based on their physiological or behavioral characteristics (e.g. fingerprints, irises, faces, etc.) to either verify a claimed identity (biometric verification) or establish the identity of an individual (biometric identification). Concerns about security, fraud and illegal access have prompted governments, airlines, banks and others to become more interested in the technology. Face recognition, in particular, has generated much more interest than other biometric systems because of the multitude of applications it enables. Unfortunately, face recognition is susceptible to variations in pose, light intensity, expression, etc. In addition to a review of biometric techniques and current approaches proposed to solve the face recognition problem, we introduce a robust face recognition system, using subspace techniques including Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), to both estimate the pose and identify the individual.*

## 1. Introduction

Physical characteristics such as face, voice, gait, etc., have been used by humans to recognize each other for thousands of years. With the increased need for fast and reliable authentication systems, the use of advanced automated systems based on biometrics technology became a necessity, and this technology is now growing rapidly. The technology makes use of the fact that each person has unique physical traits that are one's characteristics; these cannot be lost, borrowed or stolen. Using biometrics, it became possible to confirm or establish an identity based on 'who the individual is,' rather than by 'what the individual possesses' (e.g. an ID card) or 'what the individual remembers' (e.g. a password).

Passwords determine identity through user knowledge. If anyone knows the password, the person (who could be an impostor) can gainaccess to certain systems or restricted areas and resources. The

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

problem is that a password has nothing to do with the actual owner. Passwords can be stolen, and users can give their passwords to others, resulting in systems that are open to too many unauthorized individuals. There is no foolproof way to make password-protected systems safe from unauthorized users. Also, there is no way for password-based systems to determine user identity beyond doubt!

Biometrics, on the other hand, has been used during recent times to recognize people based on their physical and behavioral characteristics. Examples of different biometric systems include fingerprint recognition, face recognition, iris recognition, retina recognition, hand geometry, voice recognition and signature recognition, among others. Face recognition, in particular, has received considerable attention in recent years from both industry and the research community. The big challenge is, however, that of identifying individuals in everyday settings, such as offices or living rooms. The dynamic, noisy data involved in this type of task is very different to that used in typical computer vision research, where specific constraints are used to limit variations. Historically, such limitations have been essential in order to limit the computational resources required to process, store and analyze the visual data. However, enormous improvements in computers in terms of speed of processing and size of storage media, accompanied by progress in statistical techniques, have made it possible to develop such systems.

The material presented in this work is organized as follows. The first section gives a brief motivation for the problem of authentication. In Section 2, a review of current biometric systems is given. Section 3 discusses in detail the problem of face recognition. Section 4 discusses the different linear subspace decomposition techniques. In Section 5, we present our algorithm called view-based LDA. And in Section 6, concluding remarks are given.

## *1.1 Background*

The applications of biometrics can be divided into three main groups:

1. Commercial applications such as computer network login, electronic data security, e-commerce, Internet access, ATMs, credit cards, physical access control, cellular phones, PDAs, medical records management, distance learning, etc.
2. Government applications such as national ID cards, correctional facilities, drivers' licenses, social security, welfare disbursement, border control, passport control, etc.
3. Forensic applications such as corpse identification, criminal investigation, terrorist identification, parenthood determination, missing children, etc.

The global industry revenues of $729 m in 2002 are expected to reach $4.04 b by 2007, as shown in Figure 11.1, driven by large-scale public sector biometric deployments and the adoption of standardized biometric infrastructures and data formats.

Face recognition has received considerable interest as a widely accepted biometric because of the ease in collecting data, with or without the subject's cooperation. It is being used in a number of applications including crowd surveillance, criminal identification and criminal records, access to entry, etc. Figure 11.2 shows that facial recognition takes 15 % of the total biometric market. Even with such a success, face recognition systems developers have to deal with major issues before such systems become standard.

The requirements for a useful commercial face recognition system in busy, unconstrained environments, such as domestic living rooms or offices, can be split into several categories:

1. General requirements that need to be satisfied by all parts of the system.
2. Acquisitions requirements concerned with collecting and extracting useful information.
3. Face recognition requirements for the recognition stage concerned with how the recognition information is used [2].

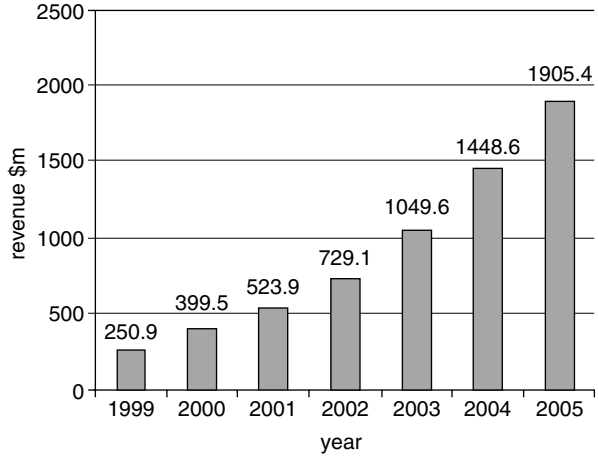



**Figure 11.1** Total biometric technology revenues [1].

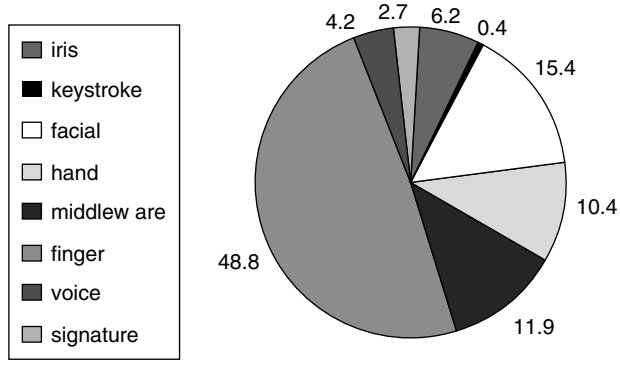



**Figure 11.2** Market shares [1].

## *1.2 The Problem of Pose*

We have seen that the major difficulty in designing robust face recognition systems is the wide variety in the input data. In particular, the problem is to recognize a subject with a varying pose. The problem of pose of the subject is one of the fundamental requirements of robust face recognition systems. Our approach to the problem is to use a combination of linear subspaces.

In previous work, view-based eigenspace systems [3] have been shown to outperform single eigenspace systems. Here, we extend this method and apply it using Linear Discriminant Analysis (LDA). We show that LDA can also be used for pose estimation. We also show that view-based LDA performs better than view-based PCA.

## 2. Review of Biometric Systems

Person authentication is becoming a regular routine in our lives. Whether one wants to access one's bank account or pass through a check point, one needs to identify oneself. Traditionally, there have been three different types of authentication:

1. Something you know – a password, PIN or a piece of personal information (such as your mother's maiden name).
2. Something you have – a card key, smart card or token (like a secure ID card).
3. Something you are – a biometric!

Of the above, a biometric is the most secure and convenient authentication tool. It cannot be borrowed, stolen or forgotten, and forging one is practically impossible (replacement part surgery, by the way, is not considered). Biometrics measures the individual's unique physical or behavioral characteristics to recognize or authenticate their identity. Common physical biometrics include fingerprints, hand or palm geometry, retina, iris, or facial characteristics to mention a few. Behavioral characters include signature, voice (which also has a physical component), keystroke pattern and gait.

A complete biometric recognition system comprises both hardware and software; the hardware collects the data, and the software interprets the data and evaluates acceptability and accessibility. The major steps in building an effective biometric system are as follows (see Figure 11.3):

1. Capture the chosen biometric.
2. Process the biometric and extract and enroll the biometric template.
3. Store the template in a local repository, a central repository or a portable token such as a smart card.
4. Live-scan the chosen biometric.
5. Process the biometric and extract the feature pattern.
6. Match the scanned biometric against stored templates.
7. Provide a matching score.
8. Record a secure audit trail with respect to system use.

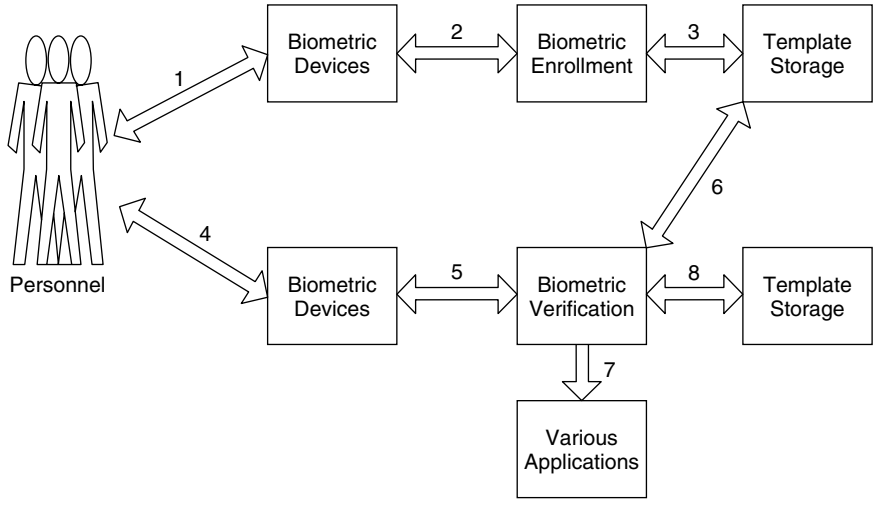



**Figure 11.3** A typical biometric recognition system.

As with any other pattern recognition system, biometric systems can be used for either: verification (or authentication) or identification (or recognition):

- **Biometric verification (Am I who I say I am?)** is a one-to- one match that compares a query biometric image against a template biometric image whose identity is being claimed. To evaluate verification performance, the verification rate (the rate at which legitimate users are granted access) vs. false accepts rate (the rate at which imposters are granted access) is plotted; this is called the ROC curve.
- **Biometric identification (Who am I?)** is a one-to-many matching process that compares a query biometric image against all the template images in a biometric database to determine the identity of the query biometric. The identification of the test image is done by locating the image in the database that has the highest similarity with the test image. The identification method is a closed test; this means that the sensor takes an observation of an individual that is known to be in the database.

## *2.1 Summary of the Performance of Different Biometrics*

Even though numerous biometrics have been proposed, some have been shown to be more successful than others. Each of these has its strengths and weaknesses, with the choice usually depending on the application. No single biometric is expected to meet effectively all the requirements for all the applications. In other words, no biometric is optimal. The match between a specific biometric and an application is determined depending upon the operational mode of the application and the properties of the biometric characteristic. A brief overview of each commonly used biometric is given below.

### 2.1.1 Fingerprint Recognition

Fingerprint recognition systems use the distinctive patterns of the fingerprint to identify or verify identity (see Figure 11.4). The main features used for identification are the ridge endings, bifurcations, dots, crossovers, bridges and their relative positions. There is a variety of approaches used in fingerprint verification. Some emulate the traditional police method of matching minutiae; others use straight

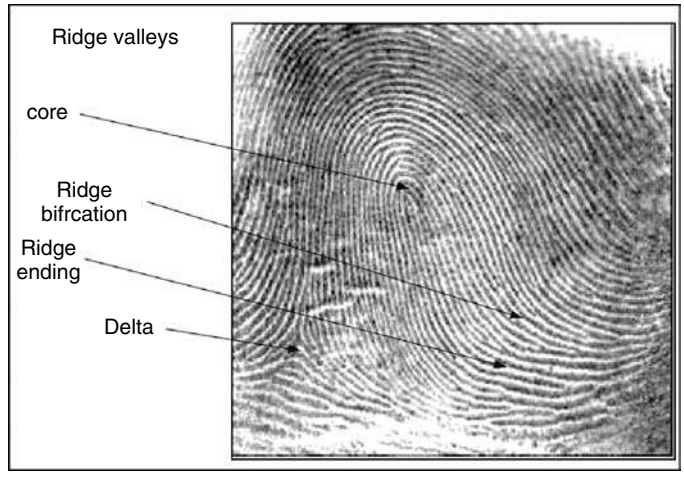



**Figure 11.4** Details of a typical fingerprint image [2].

pattern matching algorithms. The most commonly used matching method is *Minutiae-based Matching*. Two types of matching are possible: *point matching* and *structure matching*. Neural networks have also been proposed for classification. The input to the network is usually the low-pass filtered and averaged central region of the fingerprint. Given a certain fingerprint test pattern, the network assigns a match probability to each of the classes.

With the advances made in computing technology, accompanied by lower prices, fingerprint recognition systems are gaining broader acceptance, despite the common criminal stigma. Besides the decreasing cost of fingerprint recognition systems, the main advantages are: ease of use; high levels of accuracy achieved; and the wide range of deployment environments. Some of the disadvantages, however, are: high false rejection rates; the requirement for high-quality images, which are not always possible; the fact that injuries to fingers may affect successful recognition; the need for complex feature extraction algorithms; inconsistent contact distortions due to mapping of 3D shapes onto 2D surfaces; and non-uniform contact factors like dryness, dirt, sweat, disease, etc., which result in noisy images.

### 2.1.2 Iris Recognition

The iris is the plainly visible, colored ring that surrounds the pupil. It is a muscular structure that controls the amount of light entering the eye, with intricate details that can be measured, such as striations, pits and furrows (see Figure 11.5). No two irises are alike. There is no correlation between the iris patterns of even identical twins, or the right and left eye of the same individual. The amount of information that can be measured from a single iris is much greater than that from a fingerprint. Unlike other biometric technologies that can be used in surveillance mode, iris recognition is an opt-in technology. In order to use the technology, the user must cooperate with the system.

The picture of an eye is first preprocessed to localize the inner and outer boundaries of the iris and the eyelid contours, in order to extract just the iris portion. Eyelashes and reflections that may cover parts of the iris are detected and discounted. Advanced algorithms are then used to encode the iris pattern (such as the demodulation algorithm by Daugman) [5]. This creates a phase code for the texture sequence in the iris, similar to a DNA sequence code. The demodulation process uses a 2D wavelet function to represent the pattern with a 512 byte code. Some of the advantages of iris recognition systems are: accuracy; scalability; and the lack of need for contact. The disadvantage is that recognition accuracy is affected by eye infections and wearing glasses. Accuracy is also affected by how good the collated images are. Note that public acceptance to this technology is still low.

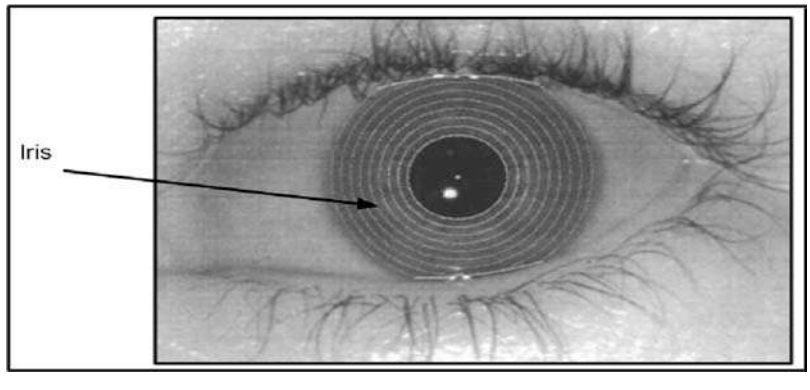



**Figure 11.5** Location of the iris [4].

### 2.1.3 Retina Recognition

A retina-based biometric involves the analysis of the layer of the blood vessels situated at the back of the eye, as shown in Figure 11.6. It involves using a low- intensity light source through an optical coupler to scan the unique patterns of the retina. Retinal scanning can be quite accurate but does require the user to look into a receptacle and focus on a given point. This is not particularly convenient if one is wearing glasses or reluctant about close contact with the reading device. For these reasons, retinal scanning is not warmly accepted by all users, even though the technology itself can work well. Advantages of the retina biometric include: high accuracy (better than fingerprints) and smaller storage requirements (35 bytes of size). Some of the disadvantages are: the system is expensive and intrusive (the eye is illuminated by a bright light); retina recognition is susceptible to eye disease; and it needs several images for registration.

### 2.1.4 Hand Geometry

To use the system, the individual aligns his/her hand according to the guide marks on the hand reader hardware, and the reader captures a three-dimensional image of the fingers and knuckles and stores the data in a template, as shown in Figure 11.7. Hand geometry involves analyzing and measuring the

**Figure 11.6** A sample of a retinal scan [6].

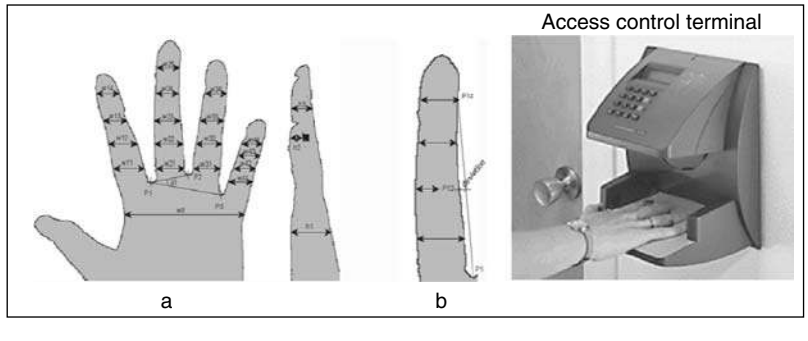



**Figure 11.7** A typical hand geometry reader [7].

shape of the hand. It offers good performance and is relatively easy to use. Hand recognition systems extract about 100 measurements of the length, width, thickness and surface of the hand of four fingers. Hand recognition systems are usually used for verification, where the measurements are compared to a template recorded during enrollment. Ease of integration into other systems and processes, coupled with ease of use, made hand geometry an obvious first step for many biometric projects. Hand geometry has been around for several years, and was used at the 1996 Olympic Games. Hand recognition systems are mainly used for verification and can achieve an equal error rate of 0.1 %.

### 2.1.5 Voice Recognition

Such systems are used to capture the speaker's voice, as well as the linguistic behaviors. They are also used in voice-to-print authentication, where complex technology transforms voice into text. Voice biometrics has the most potential for growth as it requires no new hardware since most PCs already have microphone input. However, poor quality and ambient noise can severely affect verification performance. In addition, the enrollment procedure has often been more complicated than with other biometrics, leading to the perception that voice verification is not user friendly! Its primary use continues to be telephone-based security applications. Its accuracy can be affected by a number of factors including noise and voice variations due to illness and fatigue. One obvious problem with voice recognition is fraud, as the system can always be fooled using a tape of someone else's voice.

### 2.1.6 Signature Recognition

Such systems have a wide public acceptance. Signature recognition systems, also called dynamic signature verification systems, go far beyond simply looking at the shape of a signature (see Figure 11.8). Signature verification analyzes the way a user signs his/her name, signing features such as speed, velocity and pressure are as important as the finished signature's static shape. Signature verification enjoys a synergy with existing processes that other biometrics do not. People are used to signatures as a means for transaction-related identity verification, and most would see nothing unusual in extending this to encompass biometrics. Signature verification devices are reasonably accurate in operation and obviously lend themselves to applications where a signature is an accepted identifier. The problem is that our signatures vary significantly over time and from one instance to another, so

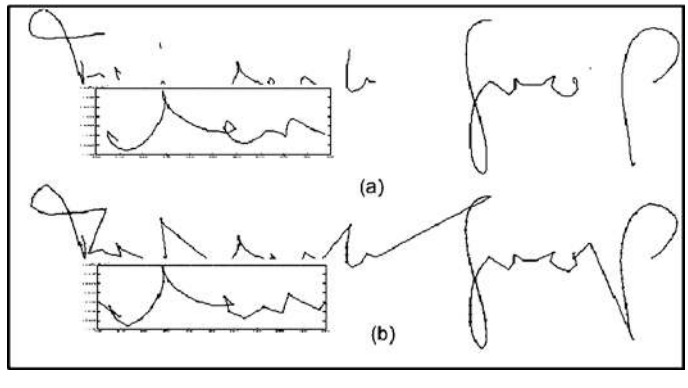



**Figure 11.8** Online processing of signatures. (a) A signature before processing; (b) the signature after preprocessing, it has been smoothed and resampled [8].

strong accuracy requires multiple samples and an extended verification process. Surprisingly, relatively few significant signature applications have emerged compared with other biometric methodologies.

### 2.1.7 Other Biometrics

Apart from the above-mentioned biometrics, there are a few others that are either appropriate for a limited number of applications or are still under development. These include biometrics like DNA, ear, facial thermograms, hand thermograms, hand veins, gait, etc. The pattern of heat radiated by the human body is a characteristic that can be captured by an infrared camera in an unobtrusive way, much like a regular (visible spectrum) photograph. The technology could be used for covert recognition. A thermogram-based system does not require contact and is noninvasive, but image acquisition is challenging in uncontrolled environments, where heat-emanating surfaces (e.g. room heaters and vehicle exhaust pipes) are present in the vicinity of the body. Biometrics which use infrared technology are facial thermograms, hand thermograms, hand veins, etc. Gait is the peculiar way one walks and is a complex spatio-temporal biometric. Gait is not supposed to be very distinctive, but is sufficiently discriminatory to allow verification in some low-security applications. Gait is a behavioral biometric and may not remain invariant, especially over a long period of time, due to fluctuations in body weight, major injuries involving joints or the brain, or due to inebriety. Acquisition of gait is similar to acquiring a facial picture and, hence, may be used as an acceptable biometric. Since gait-based systems use the video-sequence footage of a walking person to measure several different movements of each articulate joint, it is input intensive and computationally expensive.

## *2.2 Selecting the Right Biometric Technology*

What is the right biometric for my business? This question, unfortunately, does not have a single answer, as a number of parameters need to be considered when choosing a certain technology. While retinal scans, for example, are very accurate, the reluctance of people in using such a technology makes it only useful in certain environments, such as highly secure sensitive areas. Different biometric systems are appropriate for different applications, depending on perceived user profiles, the need to interface with other systems or databases, environmental conditions and a host of other application-specific parameters.

However, for any human physiological and/or behavioral characteristic to be acceptable as a biometric, a number of requirements need to be satisfied, including:

1. Universality.
2. Distinctiveness.
3. Permanence.
4. Acceptability.
5. Difficult circumvention [9].

Based on the above parameters, a comparison table for various biometric technologies has been developed (Table 11.1), where high, medium and low are denoted by H, M and L, respectively.

## *2.3 Multimodal Biometric Systems*

Some of the limitations imposed by unimodal biometric systems can be overcome by using multiple biometric modalities (such as face and fingerprint of a person or multiple fingers of a person). Such systems, known as multimodal biometric systems [10], are expected to be more reliable due to the presence of multiple, independent pieces of evidence [11]. These systems are also able to meet the stringent performance requirements imposed by various applications [11]. Multimodal biometric

**Table 11.1** Comparison of various biometric technologies [9].

| Biometric identifier | Universality | Distinctiveness | Permanence | Collectability | Performance | Acceptability | Circumvention |
|---|---|---|---|---|---|---|---|
| DNA | H | H | H | L | H | L | L |
| Ear | M | M | H | M | M | H | M |
| Face | H | L | M | H | L | H | H |
| Facial thermogram | H | H | L | H | M | H | L |
| Fingerprint | M | H | H | M | H | M | M |
| Gait | M | L | L | H | L | H | M |
| Hand geometry | M | M | M | H | M | M | M |
| Hand vein | M | M | M | M | M | M | L |
| Iris | H | H | H | M | H | L | L |
| Keystroke | L | L | L | M | L | M | M |
| Odor | H | H | H | L | L | M | L |
| Palm-print | M | H | H | M | H | M | M |
| Retina | H | H | M | L | H | L | L |
| Signature | L | L | L | H | L | H | H |
| Voice | M | L | L | M | L | H | H |

systems address the problem of nonuniversality, since multiple traits ensure sufficient population coverage. Further, multimodal biometric systems provide anti-spoofing measures by making it difficult for an intruder to simultaneously spoof the multiple biometric traits of a legitimate user. By asking the user to present a random subset of biometric traits (e.g. right index and right middle fingers, in that order), the system ensures that a live user is indeed present at the point of data acquisition. Thus, a challenge–response type of authentication can be facilitated using multimodal biometric systems. Multibiometric systems seek to alleviate some of the unimodel biometric drawbacks by providing multiple evidence of the same identity. These systems help achieve an increase in performance that may not be possible using a single biometric indicator. Combinations of face and fingerprint recognition have proven to give better performance than individual biometrics. Another scheme which used palm geometry and face recognition was also reported to yield good results [12]. However, it is still premature to speak about the best combination of biometrics. A number of research groups are currently working on developing fusion techniques to combine different biometrics. Fusion itself is a growing area of research. Different fusion techniques have been proposed, some carrying the fusion at the data level, others at the feature level, and a third class of techniques carry the fusion at the classifier level.

## 3. Face Recognition Algorithms

Face recognition has received considerable attention in recent years, both from industry and the research community. Among the popular biometric technologies considered above, facial features scored the highest compatibility in a Machine Readable Travel Documents (MRTD) system (see Figure 11.9), based on a number of evaluation factors. A considerable amount of research has been carried out in this field and many techniques have been proposed and implemented. The task of applying face recognition in real life is becoming a reality!

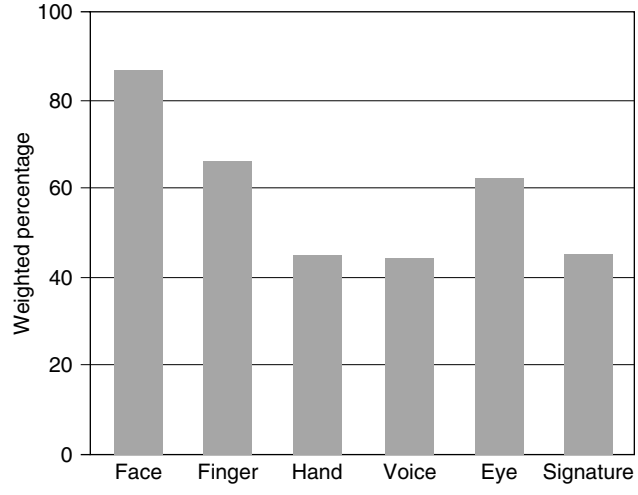



**Figure 11.9** Comparison of various biometrics (based on MRTD compatibility) [13].

The current face recognition techniques can be classified into four main categories based on the way these represent the face:

1. *Appearance based*, which uses holistic texture features.
2. *Model-based*, which employs shape and texture of the face, along with the 3D depth information.
3. *Template-based* face recognition.
4. Techniques using *neural networks*.

A summary of available techniques is displayed in Figure 11.10.

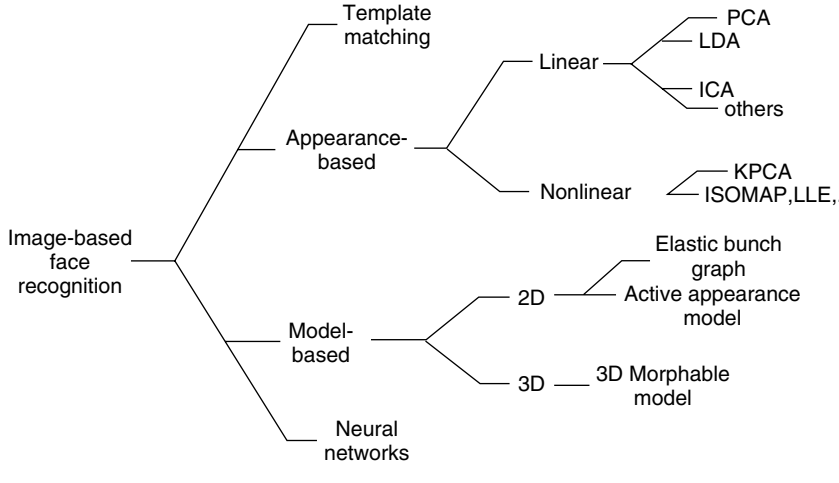



**Figure 11.10** Classification of face recognition techniques.

## *3.1 Template-based Face Recognition*

Template matching involves the use of pixel intensity information, either as original gray-level or processed to highlight specific aspects of the data. The template can either be the entire face or regions corresponding to general feature locations, such as the eyes or the mouth. Cross correlation of test images with all training images is used to identify the best match. Brunelli and Poggio [14] compared feature and template-based methods directly with the same database of frontal face views. Their template matching strategy was based on earlier work, except that they automatically detected and used feature-based templates of mouth, eyes and nose, in addition to the whole face image. They showed that template-based techniques outperform feature-based techniques.

## *3.2 Appearance-based Face Recognition*

Here, the faces are stored as two-dimensional intensity matrices. Each image is represented in a high vector space, i.e. a point in a high-dimensional vector space. In order to identify the different faces, an efficient and effective representation (feature space) is derived, depending on the application of interest. Given a test image, the similarity between the stored prototypes and the test image is then carried in the feature space. This technique is further classified into Linear (subspace) Analysis and Nonlinear (manifold) Analysis.

### 3.2.1 Linear (Subspace) Analysis

PCA, ICA and LDA are classical linear subspace analysis techniques used in face recognition. Each classifier has its own representation of high-dimensional face vector spaces called basis vectors. For each case, the basis vectors are formed according to some statistical considerations. After forming the basis vectors from the face database, faces are then projected onto the basis vectors to get a feature vector. The matching score between the projected test image and the projected training images is calculated. This matching score is inversely proportional to the distance between the projected test image and the projected training images. The larger the matching score, the better the match is.

### 3.2.2 Nonlinear (Manifold) Analysis

The nonlinear manifold is more complicated than linear models. Actually, linear subspace analysis is an approximation of this nonlinear manifold. Direct nonlinear manifold modeling schemes are explored to learn this nonlinear manifold. In what follows, Kernel Principal Component Analysis (KPCA) is introduced as an example of nonlinear analysis.

Kernel PCA is seen as a nonlinear mapping from the input space $R^n$ to the feature space $R^L$, denoted by $\phi(\mathbf{x})$, where $L$ is larger than $n$. This mapping is defined implicitly by specifying the form of the dot product in the feature space:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \bullet \phi(\mathbf{x}_j) \tag{11.1}$$

An example of a kernel commonly used is the Gaussian kernel:

$$\kappa(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|}{2\sigma^2}) \tag{11.2}$$

Let us say we have a training set of faces represented as vectors $\{\mathbf{x}_i \in R^n : i = 1 \ldots N\}$. Then, we have the corresponding set of mapped data points in feature space $\{\phi(\mathbf{x}_i)\}$. The centered points become:

$$\tilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{N}\sum_{i=1}^{N}\phi(\mathbf{x}_i) \quad i = 1 \ldots N \tag{11.3}$$

The kernel PCA algorithm is implemented as follows:

1. Choose an appropriate kernel function $\kappa(.,.)$.
2. Calculate the kernel matrix from the mapped data:

$$\mathbf{K}_{ij} \equiv \phi(\mathbf{x}_i) \bullet \phi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j) \quad i, j = 1, \ldots, N \tag{11.4}$$

3. Using $\mathbf{K}$, we construct the covariance matrix of the centered data:

$$\tilde{\mathbf{K}}_{ij} \equiv \tilde{\phi}(\mathbf{x}_i) \bullet \tilde{\phi}(\mathbf{x}_j) = \mathbf{K}_{ij} - \frac{1}{N}\sum_{p=1}^{N}\mathbf{K}_{ip} - \frac{1}{N}\sum_{q=1}^{N}\mathbf{K}_{qj} + \frac{1}{N^2}\sum_{p,q=1}^{N}\mathbf{K}_{pq} \quad i, j = 1, \ldots, N \tag{11.5}$$

4. Find the set of eigenvectors $\{\mathbf{b}_i^\alpha : i = 1, 2 \ldots N; \alpha = 1, 2 \ldots L\}$ of the matrix $\tilde{\mathbf{K}}$, which are our set of basis vectors $\{\mathbf{b}^\alpha\}$ in the feature space.
5. The KPCA components, which are not normalized, for a test point $\mathbf{x}$ are then given by:

$$p^\alpha(\mathbf{x}) \propto \mathbf{b}^\alpha \bullet \phi(\mathbf{x}) \tag{11.6}$$

The values of the components depend on the normalization taken for the set of vectors $\{\mathbf{b}^\alpha\}$, which are orthogonal but need not be orthonormal. The values of the components can be shifted to make the mean of each component over the data zero.

A comparison of different subspace algorithms was conducted (for $d = 20$) using a five-fold cross-validation method [13], where $d$ is the dimensionality of the subspace. The data was taken from the FERET database, and contained 1829 images from 706 subjects. A summary of the results obtained is given in Table 11.2. It was found that the KPCA method of face recognition performs slightly better than both PCA and ICA algorithms. In terms of the efficiency, PCA is better as it uses less computations.

## *3.3 Model-based Face Recognition*

This approach uses a model of the face to perform recognition. The model is formed from prior knowledge of facial features. A recent technique developed by Wiskott *et al.* is the elastic bunch graph matching technique [15]. Also, Cootes *et al.*, by integrating both shape and texture, developed a new technique called the 2D morphable face model, which measures face variations [16]. A more advanced 3D morphable face model has also been explored to capture the true 3D structure of the human face surface.

The model-based approach usually involves three steps:

1. Developing the face model.
2. Fitting the model to the given facial image.
3. Using the parameters of the fitted model as the feature vector to calculate the similarity between the query face and prototype faces from the database and perform the recognition.

**Table 11.2** Comparison of different subspace techniques.

| | PCA | ICA | KPCA |
|---|---|---|---|
| Accuracy | 77 % | 77 % | 87 % |
| Computation (floating point operation) | $10^8$ | $10^9$ | $10^9$ |
| Uniqueness | Yes | No | Yes |
| Projection | Linear | Linear | Nonlinear |

### 3.3.1 The Feature-based Elastic Bunch Graph Matching Technique

All human faces share a similar topological structure. Wiskott *et al.* presented a general class recognition method for classifying members of a known class of objects. Faces are represented as graphs, with nodes positioned at fiducial points and edges labeled with 2D distance vectors (see Figure 11.11). Each node contains a set of 40 complex Gabor wavelet coefficients, including both phase and magnitude, known as a jet. Face recognition is based on labeled graphs. A labeled graph is a set of nodes connected by edges; nodes are labeled with jets; edges are labeled with distances. Thus, the geometry of an object is encoded by the edges, while the gray-value distribution is patch-wise encoded by the nodes. A face bunch graph has a stack-like structure that combines graphs of individual sample faces. This provides full combinatorial power for this representation, even if it is constituted only from a few graphs.

### 3.3.2 The 3D Morphable Model

This technique utilizes the fact that the human face is a surface lying in 3D space. This implies, in principle, that the 3D model is better at representing faces, especially when handling facial variations such as pose and illumination. Blanz *et al.* [17,18] proposed a method based on a 3D morphable face model that encodes the shape and texture in terms of the model parameters, and presented an algorithm to recover these parameters from a single image of the face (see Figure 11.12).

Performance evaluation of elastic bunch graph matching was conducted on the FERET database. The results are given in Figure 11.13. The recognition results of this system are good on identical poses,

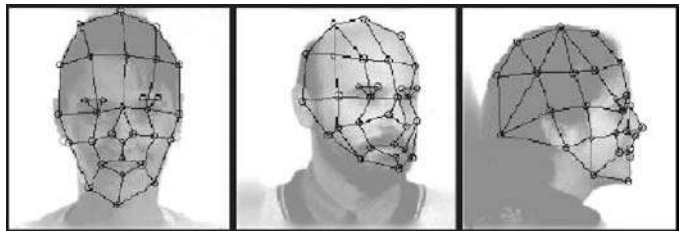


**Figure 11.11** Multiview faces overlaid with labeled graphs [13].

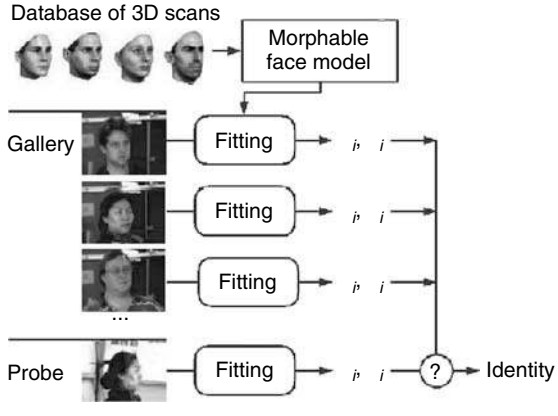



**Figure 11.12** The 3D morphable face model [18].

| | | | Testview | | |
|---|---|---|---|---|---|
| | | | frontal(%) | side(%) | profile(%) |
| Training view | frontal | mean | 94 | 85 | 65 |
| | | std | 6.3 | 20.7 | 18.2 |
| | side | mean | 89 | 90 | 70 |
| | | std | 6.4 | 9.2 | 18.9 |
| | profile | mean | 71 | 71 | 84 |
| | | std | 9.2 | 12.2 | 16.4 |

**Figure 11.13** Recognition results of the 3D morphable model [18].

e.g. frontal views against frontal views. However, across different poses, e.g. frontal views against half profiles, the system performs rather poorly.

## 4. Linear Subspace Techniques

In data mining, we often encounter situations where we have a large number of variables in the database that need to be analyzed. In such situations it is likely that subsets of variables are highly correlated with each other. The accuracy of classifiers was shown to suffer when highly correlated variables are used. One of the key steps in data mining is finding ways to reduce dimensionality without sacrificing accuracy.

Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Independent Component Analysis (ICA) fall under the broad class of linear transformations that transform a number of (possibly) correlated variables into a (smaller) number of variables. The objective is to reduce the dimensionality (number of variables) of the dataset but retain most of the original variability in the data. As such, linear subspace techniques are quite often seen as feature extraction techniques used to reduce or remove redundant or irrelevant information from the data.

The problem of dimensionality in face recognition arises when it is not known which measurement may be important for the application. That is, when there is inadequate knowledge about the structure of the data that may arise in an application. Our main aim is to study and improve classification methods, since the objective in classification is to provide a description that is well matched to the structure of the data.

PCA, ICA and LDA are classical linear subspace techniques available for face recognition. Each classifier has its own representation of high-dimensional face vector spaces called basis vectors. For each case, the basis vectors are formed according to some statistical consideration. After forming basis vectors from the faces database, faces are then projected on basis vectors to get feature vectors. Similarly, the test face is also projected. The similarity measure is used to find a match for the test face in the database. Hence, a matching score between the test face's feature vector and the trained faces' feature vectors is evaluated. The larger the matching score, the better the trained face is matched to the test face.

The above-mentioned techniques can be considered as linear transformations from the original images represented as vectors to the feature space:

$$\mathbf{Y} = \mathbf{W}^{\mathrm{T}}\mathbf{X} \tag{11.7}$$

where $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots . \mathbf{x}_N)$ represents the $n \times N$ data matrix. Each $\mathbf{x}_i$ is a vector of dimension $n$, concatenated from a $p^*p$ face image, where $n = p^*p$. Here, $n$ represents the total number of pixels in the face image and $N$ is the number of different face images in the training set, $\mathbf{Y}$ is the $d^*N$ feature vector matrix, $d$ is the dimension of the feature vector and $\mathbf{W}$ is the transformation matrix. Note that $d \ll n$.

## *4.1 Principal Component Analysis*

PCA is a technique used to uncorrelate input data. It gives an insight into the information content of input data (facial images), emphasizing the significant local and global features. These features might not be related to our notion of facial features such as eyes, nose and mouth. The eigenfaces algorithm is a decomposition algorithm based on principal component analysis that finds the vectors which best account for the distribution of facial images within the entire face database.

Recognition of images using PCA takes three basic steps. The transformation matrix **W** must be created using training images. Next, the training images are projected onto the matrix **W**. Finally, the test images are identified by projecting them into the subspace and comparing them to the trained images in the subspace domain.

### 4.1.1 Creating Matrix W

The following steps are carried out to compute matrix **W**:

1. *Removing the mean*. Each of the training images is first mean adjusted. The mean image is subtracted from each training image. The mean image of the Yale database is shown in Figure 11.14. Doing so results in the following matrix:

$$\mathbf{P} = [\mathbf{x}_1 - \mu, \mathbf{x}_2 - \mu, \mathbf{x}_3 - \mu, \ldots., \mathbf{x}_N - \mu] \tag{11.8}$$

2. *Compute the covariance matrix*. An estimate of the covariance matrix can then be obtained when matrix **P** is multiplied by its transpose (up to a scaling factor):

$$\mathbf{Q} = \mathbf{P} \times \mathbf{P}^{\mathbf{T}} \tag{11.9}$$

3. *Computing the eigenvalues and eigenvectors of* **Q**. The matrix **Q** is of size size $n \times n$ (example $10\,000 \times 10\,000$). It is difficult, if not impossible, to compute the eigenvectors for such a large matrix. This problem is fortunately solved using the method presented by Turk and Pentland [19]. This is done with the following lemma:
**Lemma 1** For any $n \times m$ matrix **L**, mapping $\mathbf{x} \rightarrow \mathbf{Lx}$ is a one to one mapping that maps eigenvectors of $\mathbf{L}^{\mathbf{T}}\mathbf{L}$ $(m \times m)$ onto those of $\mathbf{L}\mathbf{L}^{\mathbf{T}}(n \times n)$.
Therefore, we first write the new covariance matrix:

$$\tilde{\mathbf{Q}} = \mathbf{P}^{\mathbf{T}} \times \mathbf{P} \tag{11.10}$$

**Figure 11.14** Mean face of the training database.

where $\tilde{\mathbf{Q}}$ has dimensions $N \times N_1$, where $N$ is the number of faces in the training set. The $N$ largest eigenvalues and corresponding eigenvectors of $\mathbf{Q}$ can be determined from the $N$ eigenvalues and eigenvectors of $\tilde{\mathbf{Q}}$ as:

$$\lambda_i = \tilde{\lambda}_i, \text{ where } i = 1 \ldots N \tag{11.11}$$

$$\mathbf{e}_i = \tilde{\lambda}_i^{-1/2} \mathbf{P} \tilde{\mathbf{e}}_i, \text{ where } i = 1 \ldots N \tag{11.12}$$

4. *Ordering the eigenvectors*. Order the eigenvectors, $\mathbf{e}_i$s, according to their corresponding eigenvalues, $\lambda_i$s, from high to low. Keep only the eigenvectors associated with the largest $K$ eigenvalues. Form a matrix containing these eigenvectors:

$$\mathbf{W} = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \ldots, \mathbf{e}_K] \tag{11.13}$$

### 4.1.2 Projecting the Training Images

All mean-adjusted images are projected into the eigenspace. To project an image into the eigenspace, calculate the dot product of the image with each of the selected eigenvectors:

$$\mathbf{Y} = \mathbf{W}^{\mathbf{T}} \mathbf{P} \tag{11.14}$$

### 4.1.3 Identifying the Test Images

Each test image is first mean adjusted by subtracting the mean image, and then projected into the same eigenspace defined by $\mathbf{Y}$. The projected test images are compared to every projected training image. A distance measure is then used to match the test image to the training image giving the least distance. A number of similarity measures have been used in the literature, the most common one is the Euclidean distance.

### 4.1.4 Experimental Results

We have performed a leave one out experiment on the Yale faces database [20]. The Yale faces database contains images of 15 subjects in 11 different conditions, like smiling, wearing goggles, different brightness, etc (Figure 11.15). We have trained ten images per person. The plot of eigenvalues shown in Figure 11.16 is called the eigenvalue spectrum. We notice that 20 eigenfaces are enough (Figure 11.17) to reach the maximum recognition accuracy of 93.33 % (Figure 11.18).

## *4.2 Linear Discriminant Analysis*

Linear discriminant analysis has been successfully used as a dimensionality reduction technique for a number of classification problems, including speech recognition, face recognition and multimedia information retrieval. While PCA takes complete training data as one entity, LDA's goal is to find an efficient way to represent the face vector space by exploiting class information.

Using linear discriminant analysis, we desire to find a linear transformation from the original image vectors to the reduced dimension feature vectors as:

$$\mathbf{Y} = \mathbf{W}^{\mathbf{T}} \mathbf{X} \tag{11.15}$$

where $\mathbf{Y}$ is the $d \times N$ feature vector matrix, $d$ is the dimension of the feature vectors, and $\mathbf{W}$ is the transformation matrix. Note that $d \ll n$, for example $d = 25$ and $n = 10\,000$.

**Figure 11.15** Samples of different subjects [20].

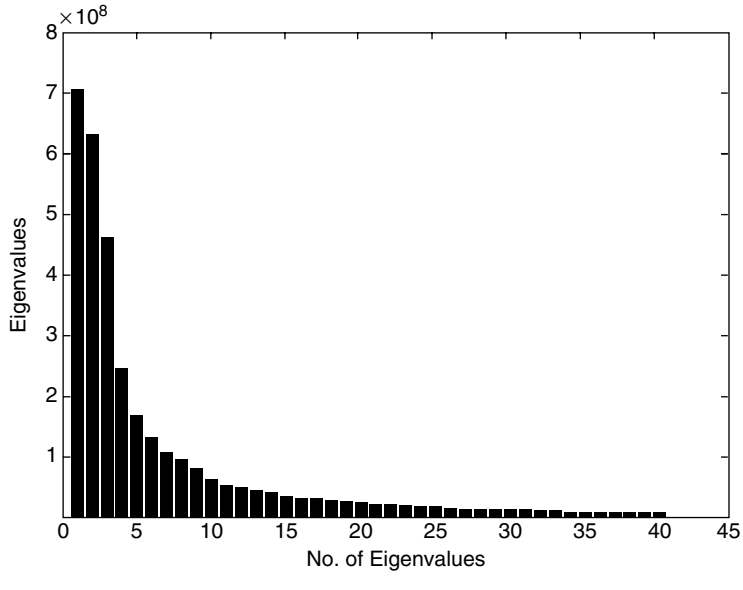



**Figure 11.16** Eigenvalue spectrum.

LDA attempts not only to reduce the dimension of the data, but also to maximize the difference between classes. To find the transformation $\mathbf{W}$, a generalized eigenproblem needs to be solved:

$$\mathbf{S}_b\mathbf{W} = \mathbf{S}_w\mathbf{W}\Lambda \tag{11.16}$$

Solving the above results in the following formulation:

$$\mathbf{W} = \left(\frac{\text{argmax}\mathbf{W}^T\mathbf{S}_b\mathbf{W}}{\mathbf{W}\ \mathbf{W}^T\mathbf{S}_w\mathbf{W}}\right) \tag{11.17}$$

**Figure 11.17** The first six eigenfaces.

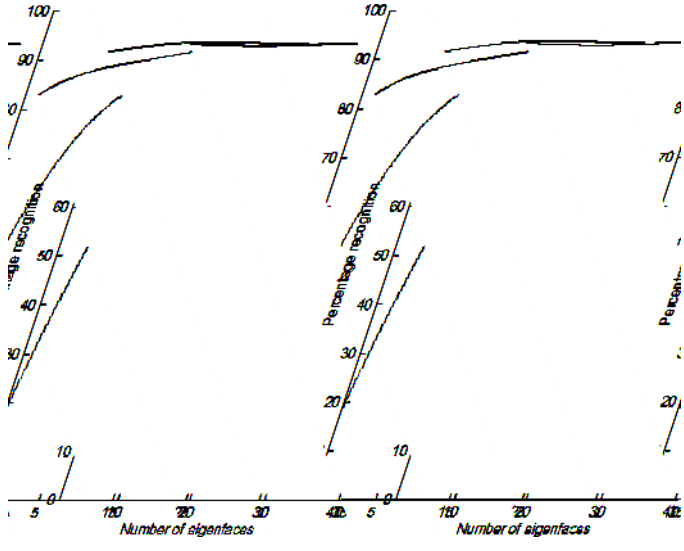



**Figure 11.18** Recognition accuracy with PCA.

where $\mathbf{S}_\mathrm{b}$ is the between-class scatter matrix and $\mathbf{S}_\mathrm{w}$ is the within-class scatter matrix defined as:

$$\mathbf{S}_\mathrm{w} = \sum_{i=1}^{c} P(C_i)\mathbf{S}_i \tag{11.18}$$

$$\mathbf{S}_\mathrm{b} = \sum_{i=1}^{c} P(C_i)(\mu_i - \mu)(\mu_i - \mu)^\mathrm{T} \tag{11.19}$$

where $c$ is the number of classes ($c = 15$) and $P(C_i)$ is the probability of class $i$. Here, $P(C_i) = 1/c$, since all classes are equally probable.

$\mathbf{S}_i$ is the class-dependent scatter matrix and is defined as:

$$\mathbf{S}_i = \frac{1}{\mathbf{N}_i} \sum_{\mathbf{x}_k \in X_i} (\mathbf{x}_k - \mu_i)(\mathbf{x}_k - \mu_i)^{\mathrm{T}} \quad i = 1, \ldots, c \tag{11.20}$$

One method for solving the generalized eigenproblem is to take the inverse of $\mathbf{S}_{\mathrm{w}}$ and solve the following eigenproblem for matrix $\mathbf{S}_{\mathrm{w}}^{-1}\mathbf{S}_{\mathrm{b}}$:

$$\mathbf{S}_{\mathrm{w}}^{-1}\mathbf{S}_{\mathrm{b}}\mathbf{W} = \mathbf{W}\Lambda \tag{11.21}$$

where $\Lambda$ is the diagonal matrix containing the eigenvalues of $\mathbf{S}_{\mathrm{w}}^{-1}\mathbf{S}_{\mathrm{b}}$.

But this problem is numerically unstable as it involves direct inversion of a very large matrix, which is probably close to singular. One method for solving the generalized eigenvalue problem is to simultaneously diagonalize both $\mathbf{S}_{\mathrm{w}}$ and $\mathbf{S}_{\mathrm{b}}$ [21]:

$$\mathbf{W}^{\mathbf{T}}\mathbf{S}_{\mathrm{w}}\mathbf{W} = \mathbf{I}, \quad \mathbf{W}^{\mathbf{T}}\mathbf{S}_{\mathrm{b}}\mathbf{W} = \Lambda \tag{11.22}$$

The algorithm can be outlined as follows:

1. Find the eigenvectors of $\mathbf{P}_{\mathrm{b}}^{\mathbf{T}}\mathbf{P}_{\mathrm{b}}$ corresponding to the largest $K$ nonzero eigenvalues, $\mathbf{V}_{c\times K} = [\mathbf{e}_1, \mathbf{e}_2 \ldots \mathbf{e}_K]$ where $\mathbf{P}_{\mathrm{b}}$ of size $(n \times c)(\mathbf{S}_{\mathrm{b}} = \mathbf{P}_{\mathrm{b}}\mathbf{P}_{\mathrm{b}}^{\mathbf{T}})$.
2. Deduce the first $K$ most significant eigenvectors and eigenvalues of $\mathbf{S}_{\mathrm{b}}$:

$$\mathbf{Y} = \mathbf{P}_{\mathrm{b}}\mathbf{V} \tag{11.23}$$

$$\mathbf{D}_{\mathrm{b}} = \mathbf{Y}^{\mathbf{T}}\mathbf{S}_{\mathrm{b}}\mathbf{Y} = (\mathbf{Y}^{\mathbf{T}}\mathbf{P}_{\mathrm{b}})(\mathbf{P}_{\mathrm{b}}^{\mathbf{T}}\mathbf{Y}) \tag{11.24}$$

3. Let $\mathbf{Z} = \mathbf{Y}\mathbf{D}_{\mathrm{b}}^{-1/2}$, which projects $\mathbf{S}_{\mathrm{b}}$ and $\mathbf{S}_{\mathrm{w}}$ onto a subspace spanned by $\mathbf{Z}$, this results in:

$$\mathbf{Z}^{\mathbf{T}}\mathbf{S}_{\mathrm{b}}\mathbf{Z}(\equiv \mathbf{I}) \text{ and } \mathbf{Z}^{\mathbf{T}}\mathbf{S}_{\mathrm{w}}\mathbf{Z} \tag{11.25}$$

4. We then diagonalize $\mathbf{Z}^{\mathbf{T}}\mathbf{S}_{\mathrm{w}}\mathbf{Z}$, which is a small matrix of size $(K \times K)$:

$$\mathbf{U}^{\mathbf{T}}\mathbf{Z}^{\mathbf{T}}\mathbf{S}_{\mathrm{w}}\mathbf{Z}\mathbf{U} = \Lambda_{\mathrm{w}} \tag{11.26}$$

5. We discard the large eigenvalues and keep the smallest $r$ eigenvalues, including the 0 s. The corresponding eigenvector matrix becomes $\mathbf{R}$ $(k \times r)$.
6. The overall LDA transformation matrix becomes $\mathbf{W} = \mathbf{Z}\mathbf{R}$. Notice that we have diagonalized both the numerator and the denominator in the Fisher criterion.

### 4.2.1 Experimental Results

We have also performed a leave one out experiment on the Yale faces database [20]. The first six Fisher faces are shown in Figure 11.19. The eigenvalue spectrum of between-class and within-class covariance matrices is shown in Figure 11.20. We notice that 14 Fisher faces are enough to reach the maximum recognition accuracy of 93.33 %. The result of recognition accuracy with respect to the number of Fisher faces is shown in Figure 11.21. Using LDA, we have achieved a maximum recognition accuracy of 93.333 %.

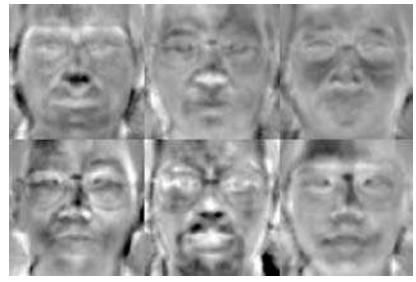


**Figure 11.19** The first six LDA basis vectors.

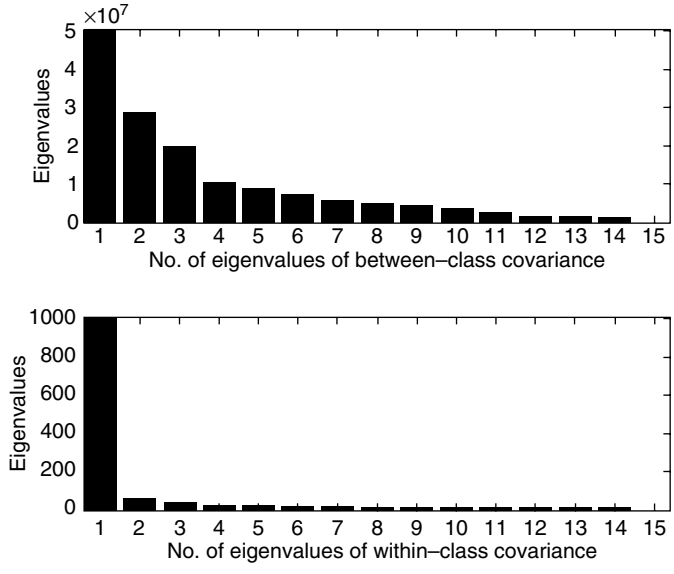



**Figure 11.20** Eigenvalue spectrum of between-class and within-class covariance matrices.

## *4.3 Independent Component Analysis*

Since PCA only considers second order statistics, it lacks information on the complete joint probability density function and higher order statistics. Independent Component Analysis (ICA) accounts for such information and is used to identify independent sources from their linear combination. In face recognition, ICA is used to provide an independent, rather than an uncorrelated, image decomposition.

In deriving the ICA algorithm, two main assumptions are made:

1. The input components are independent.
2. The input components are non-Gaussian.

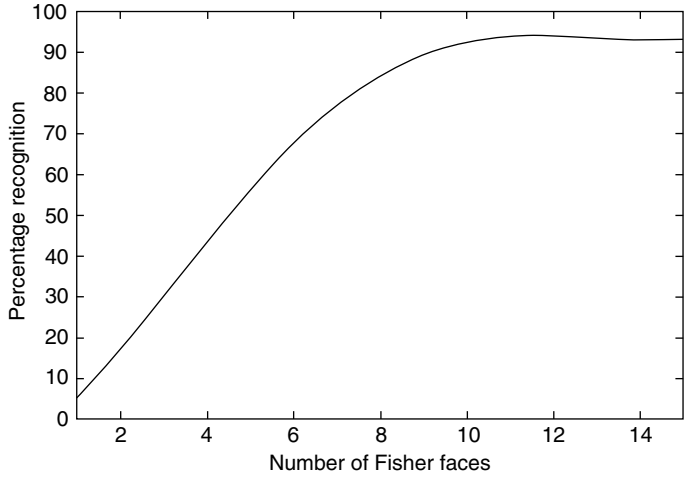



**Figure 11.21** Recognition accuracy with LDA.

Non-Gaussianity, in particular, is measured using the kurtosis function. In addition to the above assumptions, the ICA has three main limitations:

1. Variances of the independent components can only be determined up to a scaling.
2. The order of the independent components cannot be determined (only determined up to a permutation order).
3. The number of separated components cannot be larger than the number of observation signals.

The four main stages of the ICA algorithm are: preprocessing; whitening; rotation; and normalization.

The preprocessing stage consists of centering the data matrix $\mathbf{X}$ by removing the mean vector from each of its column vectors.

The whitening stage consists of linearly transforming the mean removed input vector $\tilde{\mathbf{x}}_i$ so that a new vector is obtained whose components are uncorrelated.

The rotation stage is the heart of ICA. This stage performs source separation to find the independent components (basis face vectors) by minimizing the mutual information.

A popular approach for estimating the ICA model is maximum likelihood estimation, which is connected to the info-max principle and the concept of minimizing the mutual information.

A fast ICA implementation has been proposed in [22]. The FastICA is based on a fixed point iteration scheme for finding a maximum of the non-Gaussianity of $\mathbf{W^T X}$. Starting with a certain activation function $g(.)$ such as:

$$g(u) = \tan(a1u) \text{ or } g(u) = u(-u^2/2) \text{ or } g(u) = u^3 \tag{11.27}$$

the basic iteration in the FastICA is as follows:

1. Choose an initial (random) transformation $\mathbf{W}$.
2. Let $\mathbf{W}^+ = \mathbf{W} + \mu[\mathbf{I} + g(\mathbf{y})\mathbf{y}^{\mathrm{T}}]\mathbf{W}$, where $\mu$ is the learning rate and $\mathbf{y} = \mathbf{W}\mathbf{x}$.
3. Normalize $\mathbf{W}^+$ and repeat until convergence.

The last stage in implementing the ICA is the normalization operation that derives unique independent components in terms of orientation, unit norm and order of projections.

**Figure 11.22** The first six ICA basis vectors.

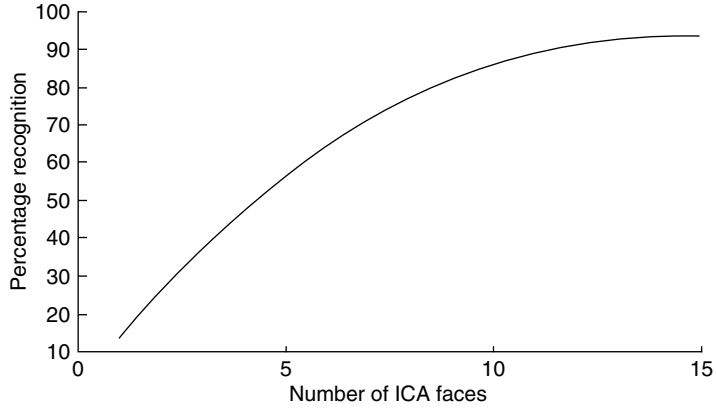



**Figure 11.23** Recognition accuracy with ICA.

### 4.3.1 Experimental Results

We have performed a leave one out experiment on the Yale faces database [20], the same experiment as performed on LDA and PCA. The first six ICA basis vectors are shown in Figure 11.22 and the curve for recognition accuracy is shown Figure 11.23.

## 5. A Pose-invariant System for Face Recognition

The face recognition problem has been studied for more than two decades. In most systems, however, the input image is assumed to be a fixed size, clear background mug shot. However, a robust face recognition system should allow flexibility in pose, lighting and expression. Facial images are high-dimensional data and facial features have similar geometrical configuration. As such, under general conditions where pose, lighting and expression are varying, the face recognition task becomes more difficult. The reduction of that variability through a preliminary classification step enhances the performance of face recognition systems.

Pose variation is a nontrivial problem to solve as it introduces nonlinear transformations (Figure 11.24). There have been a number of techniques proposed to overcome the problem of varying pose for face recognition. One of these was the application of Growing Gaussian Mixtures Models [23], GMMs are applied after reducing the data dimensions using PCA. The problem is that since

**Figure 11.24** A subject in different poses.

GMM is a probabilistic approach, it requires a sufficient amount of training faces, which are usually not available (for example, 50 faces to fit five GMMs). One alternative is to use a three-dimensional model of the face [15]. However, 3D models are expensive and difficult to develop.

The view-based eigenspaces of Moghaddam and Pentland [3] have also shown that separate eigenspaces perform better than using a combined eigenspace of the pose-varying images. This approach essentially consists of several discrete systems (multiple observers). We extend this method and apply it using linear discriminant analysis. In our experiments, we will show that view-based LDA performs better than view-based PCA. We have also demonstrated that LDA can be used to do pose estimation.

## *5.1 The Proposed Algorithm*

We propose, here, a new system which is invariant to pose. The system consists of two stages. During the first stage, the pose is estimated. In stage two, a view-specific subspace analysis is used for recognition. The block diagram is shown in Figure 11.25. To train the system, we first organize the images from the database into three different views and find the subspace transformation for each of these views.

In the block diagram, we show the sizes of the matrices at different stages, so as to get the notion of dimensionality reduction. The matrices **XL**, **XR** and **XF** are of size $60 \times 2128$ (three images/person, 20 people, 2128 pixels per image). **WL**, **WR** and **WF** are the transformation matrices, each containing $K$ basis vectors (where $K = 20$). **YL**, **YR** and **YF** are the transformed matrices, called template matrices, each of size $60 \times K$.

## *5.2 Pose Estimation using LDA*

The pose estimation stage is composed of a learning stage and a pose estimation stage. In this work, we are considering three possible classes for the pose. These are: the left pose at 45°($C_1$), the front pose ($C_2$) and the right pose at 45°($C_3$). Some authors considered five and seven possible rotation angles, but our experiments have shown that the three angles mentioned above are enough to capture the main features of the face.

Each of the faces in the training set is seen as an observation vector $\mathbf{x}_i$ of a certain random vector $\mathbf{x}$. These are denoted as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, Each of these is a face vector of dimension $n$, concatenated from a $p \times p$ facial image ($n$ is the number of pixels in the facial image, for the faces in the UMIST database $n = 2128$). An estimate of the expected value of $\mathbf{x}$ can be obtained using the average:

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i \tag{11.28}$$

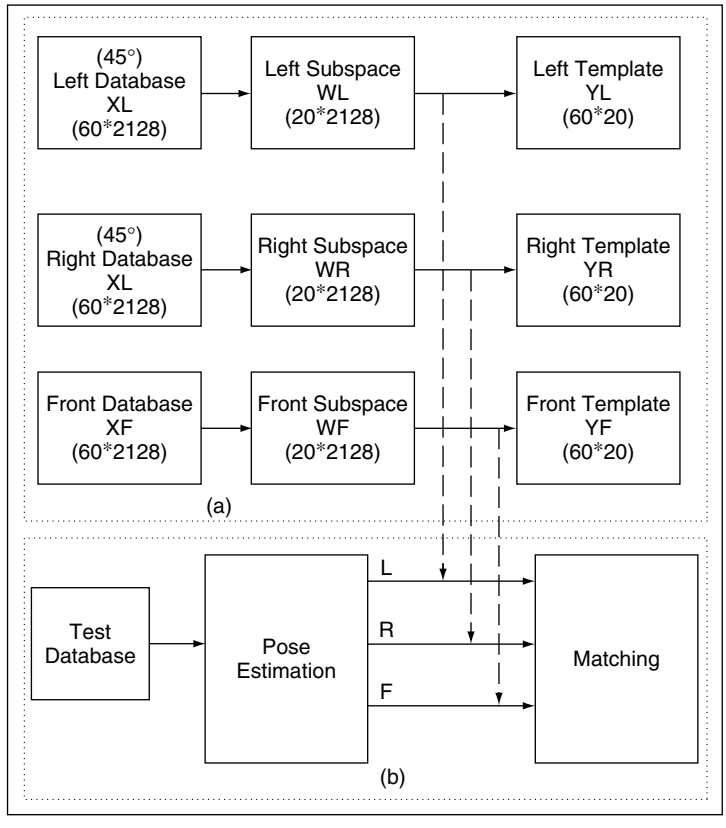



**Figure 11.25** Block diagram of the pose-invariant subspace system. (a) View-specific subspace training; (b) pose estimation and matching.

In this training set, we have $N$ observation vectors $(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N)$, $N_1$ of which belong to class $C_1$, $N_2$ to class $C_2$, and $N_3$ to class $C_3$. These classes represent the left pose at 45°, the front pose and the right pose at 45°, respectively.

After subtracting the mean vector $\boldsymbol{\mu}$ from each of the image vectors, we combine the vectors, side by side, to create a data matrix of size $n \times N$:

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N] \tag{11.29}$$

Using linear discriminant analysis, we desire to find a linear transformation from the original image vectors to the reduced dimension feature vectors as:

$$\mathbf{Y} = \mathbf{W}^{\mathbf{T}}\mathbf{X} \tag{11.30}$$

where $\mathbf{Y}$ is the $d \times N$ feature vector matrix, $d$ is the dimension of the feature vectors and $\mathbf{W}$ is the transformation matrix. Note that $d \ll n$.

As mentioned in Section 4, linear discriminant analysis (LDA) attempts to reduce the dimension of the data and maximize the difference between classes. To find the transformation $\mathbf{W}$, a generalized eigenproblem is solved:

$$\mathbf{S}_b\mathbf{W} = \mathbf{S}_w\mathbf{W}\Lambda \tag{11.31}$$

where $\mathbf{S}_b$ is the between-class scatter matrix and $\mathbf{S}_w$ is the within-class scatter matrix.

Using the transformation **W**, each of the images in the database is transformed into a feature vector of dimension *d*. To estimate the pose of a given image, the image is first projected over the columns of **W** to obtain a feature vector **z**. The Euclidian distance is then used to compare the test feature vector to each of the feature vectors from the database. The class of the image corresponding to the minimum distance is then selected as the pose of the test image.

## *5.3 Experimental Results for Pose Estimation using LDA and PCA*

The experiments were carried out on the UMIST database, which contains 20 people and a total of 564 faces in varying poses. Our aim was to identify whether a subject was in pose left, right or front, so that we could use the appropriate view-based LDA. We performed pose estimation using both techniques, LDA and PCA. The experiments were carried out using three poses for each of the 20 people. We trained the system using ten people and tested it using the remaining ten people. The mean images from the three different poses are shown in Figure 11.26.

Similarly, we trained the 'pose estimation using PCA' algorithm, but here we did not use any class information. Hence, we used the training images in three different poses: left 45 degrees, right 45 degrees and front. The results are show in Table 11.3.

We noticed that LDA outperformed PCA in pose estimation. The reason being the ability of LDA to separate classes, while PCA only classifies features. As mentioned above, LDA maximizes the ratio of variances of between classes to within classes.

## *5.4 View-specific Subspace Decomposition*

Following the LDA procedure discussed above, we can derive an LDA transformation for each of the views. As such, using the images from each of the views and for all individuals, we obtained three transformation matrices: **XL**, **XR** and **XF** for left, right and front views respectively.


**Figure 11.26** Mean images of faces in front, left, and right poses.

**Table 11.3** Experimental results of pose estimation.

| No. of test images | PCA | LDA |
|---|---|---|
| 90 | 90 % | 100 % |
| 180 | 88.333 % | 98.888 % |

## 5.5 Experiments on the Pose-invariant Face Recognition System

We carried out our experiments on view-based LDA and compared the results to other algorithms. In the first experiment, we compared View-based LDA (VLDA) to the Traditional LDA (TLDA) [21]. The Fisher faces for the front, left and right poses are displayed in Figures 11.27, 11.28 and 11.29,

**Figure 11.27** Fisher faces trained for front faces (View-based LDA).

**Figure 11.28** Fisher faces trained for left faces (View-based LDA).

**Figure 11.29** Fisher faces trained for right faces (View-based LDA).

respectively. Figure 11.30 shows the Fisher faces obtained using a unique LDA (TLDA). The performance results are presented in Figure 11.31. We noticed an improvement of 7 % in recognition accuracy. The reason for this improvement is that we managed to reduce within-class correlation by training different view-specific LDAs. This resulted in an improved Fisher criterion. For the same reason, we see that VLDA performs better than TPCA [19] (Figure 11.32). Experiments were also carried out on view-based PCA [3] and the results compared to those of PCA [19] and VPCA [3]. We found that there is not much improvement in the results and the recognition accuracy remains the same as we increase the number of eigenfaces (Figure 11.33). The reason for this could be that PCA just relies on the covariance matrix of the data and training view-specific PCAs does not help much in improving the separation.

For all experiments, we see that the proposed view-based LDA performs better than traditional LDA and traditional PCA. Since the performance of LDA gets better if we have larger databases, we expect

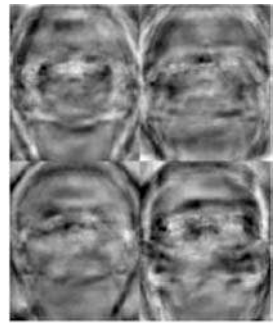


**Figure 11.30** Fisher faces trained for traditional LDA.

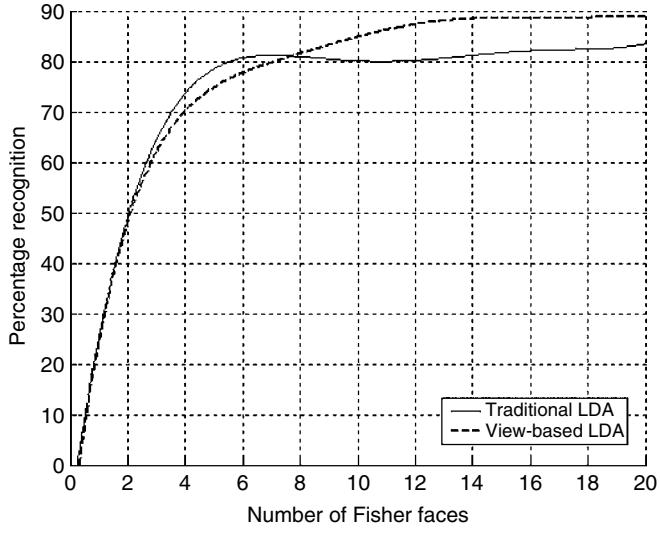



**Figure 11.31** View-based LDA vs. traditional LDA.

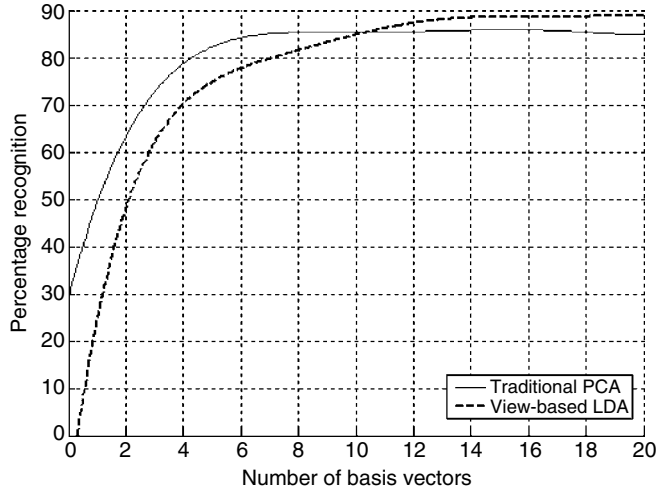



**Figure 11.32** View-based LDA vs. traditional PCA.

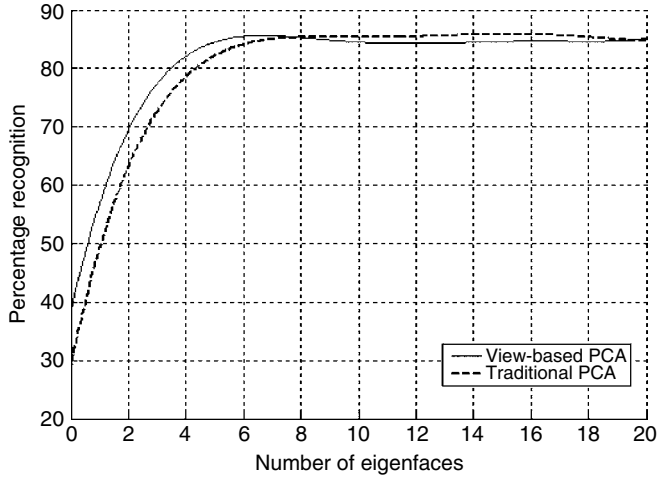



**Figure 11.33** View-based PCA vs. traditional PCA.

our view-based LDA to achieve better recognition accuracy by using more training faces for each of the poses.

Table 11.4 summarizes the results of the experiments carried out using the pose-invariant system. The table summarizes the recognition accuracy, and clearly shows that VLDA outperforms all other algorithms, followed by VPCA, with the maximum number of Fisher/eigenfaces being 20. The computational complexity and memory usage of all algorithms was comparable.

**Table 11.4** Summary of results.

| Algorithm | Time (in s) | Max. recognition accuracy | Memory usage |
|---|---|---|---|
| View-based LDA | 183.650 | 88.333 | 514 320 |
| Traditional LDA | 288.719 | 83.333 | 429 200 |
| View-based PCA | 90.4850 | 84.444 | 514 320 |
| Traditional PCA | 140.9060 | 85 | 429 200 |

## 6. Concluding Remarks

In this chapter, we have presented an overview of biometric techniques and focused in particular on face recognition algorithms. We have discussed in detail the different linear subspace techniques. We introduced a new approach to pose-invariant face recognition using the LDA algorithm. In summary, we would like to conclude the chapter with the following comments:

- Face recognition continues to attract a lot of attention from both the research community and industry.
- We notice that there is a move towards face recognition using 3D models rather than the 2D images used traditionally by authors.
- Numerous techniques have been proposed for face recognition, however, all have advantages and disadvantages. The choice of a certain technique should be based on the specific requirements for the task at hand and the application of interest.
- Although numerous algorithms exist, robust face recognition is still difficult.
- A major step in developing face recognition algorithms is testing and benchmarking. Standard protocols for testing and benchmarking are still being developed.
- Face recognition from video is starting to emerge as a new robust technology in the market.
- The problems of illumination, pose, etc. are still major issues that researchers need to consider in developing robust algorithms.
- Finally, with the introduction of a number of new biometric technologies, there is an urgent need to consider face recognition as part of more comprehensive multimodal biometric recognition systems.

## References

[1] International biometric group, *Market report 2000–2005*, http://www.biometricgroup.com/, September 2001.
[2] Jain, L. C., Halici, U., Hayashi, I., Lee, S. B. and Tsutsui, S. *Intelligent Biometric Techniques in Fingerprint and Face Recognition*. CRC Press, 1990.
[3] Moghaddam, B. and Pentland, A. "Face recognition using view-based and modular eigenspaces," *SPIE*, **2277**, pp. 12–21, 1994.
[4] Iridiantech, http://www.iridiantech.com/how/index.php.
[5] Daugman, J. "Recognizing Persons by Their Iris Patterns," in Jain, A. K. Bolle, R. and Pankanti, S. (Eds), *Biometrics: Personal Identification in Networked Society*, Kluwer Academic Publishers, 1999.
[6] "Eyesearch," www.eyesearch.com/ diabetic.retinopathy.htm.
[7] Ross, A., Jain, A. K. and Pankanti, S. "A prototype hand geometry-based verification system," *Proceedings of Audio-and Video-Based Personal Identification (AVBPA-99)*, pp. 166–171, 1999.
[8] Liu, C., Lu, Z., Zou, M. and Tong, J. "On-line signature verification using local shape analysis." Automation Building, Institute of Automation,.
[9] Ross, A., Jain, A. K. and Prabhakar, S. "An introduction to biometric recognition," to appear in *IEEE Transactions on Circuits and Systems for Video Technology*.
[10] Jain, A. K., Hong, J. L. and Pankanti, S. "Can multibiometrics improve performance?" *Proceedings of AutoID'99*, pp. 59–64, 1999.
[11] Kuncheva, L. I., Whitaker, C. J., Shipp, C. A. and Duin, R. P.W. "Is Independence Good for Combining Classifiers?," *International Conference on Pattern Recognition (ICPR)*, pp. 168–171, 2000.

[12] Jain, A. K. and Ross, A. "Information fusion in biometrics," *Pattern Recognition Letters*, **24**, pp. 2115–2125, 2003.
[13] Lu, X. *Image analysis for face recognition*, Department of Computer Science and Engineering, Michigan State University, USA.
[14] Brunelli, R. and Poggio, T. "Face Recognition: Features versus Templates," *IEEE Transactions on PAMI*, **15**(10), pp. 1042–1052, 1993.
[15] Wiskott, N. K. L., Fellous, J. M. and von der Malsburg, C. "Face recognition by elastic bunch graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7), pp. 775–779, 1997.
[16] Taylor, C. J., Edwards, G. J. and Cootes, T. F. "Face recognition using active appearance models," in *Proceedings ECCV*, **2**, pp. 581–695, 1998.
[17] Huang, J., Heisele, B. and Blanz, V. "Component-based Face Recognition with 3D Morphable Models," *Proceedings of the Fourth International Conference on Audio- and Video-based Biometric Person Authertication*, Surrey, UK, 2003.
[18] Volker Blanz, S. R. and Vetter, T. "Face identification across different poses and illuminations with a 3D morphable model," in *IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 202–207, 2002.
[19] Turk, M. and Pentland, A. "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, **3**(1), pp. 72–86, 1991.
[20] "Yale University face database," http://cvc.yale.edu/projects/yalefaces/yalefaces.html.
[21] Yang, J. and Yu, H. "A direct LDA algorithm for high dimensional data with application to face recognition." Preprint submitted to *Pattern Recognition Letters*, September 2000.
[22] Hyvrinen, A. and Oja, E. "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, **9**(7), pp. 1483–1492, 1997.
[23] Waibel, A., Gross, R. and Yang, J. *Growing Gaussian mixtures models for pose-invariant face recognition*, Interactive Systems Laboratories, Carnegie Mellon University, Pittsburg, PA, USA.
[24] "Speech enhancement and robust speech recognition," http://www.ifp.uiuc.edu/speech/.
[25] "Dna," http://library.thinkquest.org/16985/dnamain.htm.
[26] Hong, L. and Jain, A. K. "Integrating faces and fingerprints for personal identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp. 1295–1307, 1998.
[27] Akamatsu, S., Vonder, C., Isburg, M. and Okada, M. K. "Analysis and synthesis of pose variations of human faces by a linear pcmap model and its application for pose-invariant face recognition on systems," in *Fourth International Conference on Automatic Face and Gesture Recognition*, 2000.
[28] Schrater, P. R. "Bayesian data fusion and credit assignment in vision and fmri data analysis," *Computational Image Proceedings of SPIE*, **5016**, pp. 24–35, 2003.
[29] Moghaddam, B. "Principal manifolds and probabilistic subspaces for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**, pp. 780–788, 2002.
[30] Jamil, N., Iqbal, S. and Iqbal, N. "Face recognition using neural networks," *Proceedings of the 2001 IEEE INMIC Conference*, pp. 277–281, 2001.
[31] Liu, X., Chen, W. Z. T., Hsu, Y. J. "Principal component analysis and its variants for biometrics," *ECJ*, pp. 61–64, 2002.
[32] Comon, P. "Independent component analysis – a new concept?," *Signal Processing*, **36**, pp. 287–314, 1994.
[33] Jutten, C. and Herault, J. "Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, **24**, pp. 1–10, 1991.
[34] Hyvrinen, A. and Oja, E. *Independent component analysis: A tutorial*, Helsinki University of Technology, Laboratory of Computer and Information Science, 1999.
[35] Huber, P. "Projection pursuit," *The Annals of Statistics*, **13**(2), pp. 435–475, 1985.
[36] Jones, M. and Sibson, R. "What is projection pursuit?," *Journal of the Royal Statistical Society*, **ser. A**(150), pp. 1–36, 1987.
[37] Hyvrinen, A. "New approximations of differential entropy for independent component analysis and projection pursuit," *Neural Information Processing Systems*, **10**, pp. 273–279, 1998.
[38] Hyvrinen, A. "Survey on independent component analysis," *Neural Computing Surveys*, **2**, pp. 94–128, 1999.
[39] Sarela, J., Hyvrinen, A. and Vigrio, R. "Spikes and bumps: Artefacts generated by independent component analysis with insufficient sample size," in *International Workshop on Independent Component Analysis and Signal Separation (ICA'99)*, Aussois, France, pp. 425–429, 1999.
[40] Hyvrinen, A. "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks*, **10**(3), pp. 626–634, 1999.

[41] Wang, L., Vigario, R., Karhunen, J., Oja, E. and Joutsensalo, J. "A class of neural networks for independent component analysis," *IEEE Transactions on Neural Networks*, **8**(3), pp. 486– 504, 1997.
[42] Murase, H. and Nayar, S. K. "Parametric eigenspace representation for visual learning and recognition," *Workshop on Geometric Methods in Computer Vision*, SPIE, San Diego, pp. 378–391, 1993.
[43] Wilks, S. S. *Mathematical Statistics*, John Wiley & Sons, Inc., New York, 1962.

# 12

# Developmental Vision: Adaptive Recognition of Human Faces by Humanoid Robots

**Hon-fai Chia**
**Ming Xie**
School of Mechanical and Production Engineering, Nanyang Technological University, Singapore 639798

*The objective of this chapter is to push forward the idea that machine learning should be inclined towards the direction of developmental learning, like a human baby growing up with developmental learning models, developed both physically and mentally. Human face recognition by humanoid robots is chosen as the sample application to illustrate the possibility of developmental vision in humanoids. We start with a Restricted Coulomb Energy (RCE) neural network, which enables the learning of color prototypes and performs human presence detection through skin color segmentation. Then, we choose hidden Markov models for the purpose of both learning and recognition of human facial images, which depend on supervised classification training. As for feature extraction, we propose the method of wavelet packet decomposition.*

## 1. Introduction

Developmental visual capability is an important milestone to be conquered before achieving machine intelligence. An 'intelligent machine' should develop the ability of developmental visual learning, like a newborn who can visually adapt to his/her environment. Numerous developmental models are currently under active investigation and implementation in order to guide young children towards a healthy and fulfilling acquisition of knowledge, attempting to nurture an 'intelligent' human. The development of a human can be broadly classified into two aspects, namely: physical and mental growth. In the aspect of physical growth, a baby will need years to fully develop his/her body, whereas the construction of a humanoid robot may be completed within a year with the current state-of-the-art technology. Humanoid robots, hence, are capable of outperforming humans in the physical growth

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

stage. Then, what about mental growth? Mental growth of humanoid robots is an area with intense challenges and research values for investigation.

Babies can adaptively learn and acquire knowledge by various means, and gain 'intelligence' as they age physically. Robots, however, often operate in a 'single-minded' way, and perform fixed routines with limited adaptive capabilities. Recent advances in artificial intelligence, cognitive science, neuroscience and robotics have stimulated the interest and growth of a new research field, known as computational autonomous mental development. This field prompts scientists to brainstorm on formulating developmental learning models for robots. In order to benefit human–robot interaction, it is necessary to build an interactive dual channel of communication for both robots and humans.

This chapter pushes forward the idea that machine learning should be inclined towards the direction of developmental learning, should automated and intelligent artificial systems be desired. The situation of adaptive recognition of human faces by humanoid robots through developmental vision is to be discussed as a sample application. The developmental vision paradigm for human face recognition by humanoid robots consists of four distinct stages. This chapter is organized as follows. The next section will focus on the discussion of the supervised developmental visual learning through interaction with human masters, which is analogous to a human baby growing up with developmental learning models. In Section 3, we will present how developmental learning of colors is achieved by using a probabilistic RCE neural network to address the problem of detecting the presence of human faces based on the skin color information. Section 4 will explain how to apply methodologies of wavelet packet analysis to perform the estimation of feature maps which are the results of facial locations. We also demonstrate the use of Hidden Markov Models (HMMs) to learn facial image recognition (i.e. training and classification). Finally, experimental results are presented and discussed in Section 5. We also highlight possibilities for future humanoid robots with developmental vision to visually learn and recognize other types of object.

## 2. Adaptive Recognition Based on Developmental Learning

Much research has been attempted into the development of 'machine intelligence', with the goal of achieving seeing and thinking machines, and also machines incorporating the capability to 'grow' incrementally in both psychological and physical aspects. These ambitions are the main motivations behind the recent proliferation of cognitive science studies in trying to discover the complex mental capabilities of humans as a basis for inspiration or imitation.

Lifelong developmental learning by a human allows him or her to accumulate vast domains of knowledge at different stages of life and grow mentally while the biological cycle of physical maturation from infants to fully grown adults takes place.

Therefore, lifelong developmental learning by robots will be a crucial area of research for artificial intelligence ('seeing and thinking machines') in order to allow a quantum leap from the current computer-aided decision making to the future decision making by machines. Machines in the future will hence be treated as agents, instead of as tools to help extend humans' mental and physical capabilities.

### *2.1 Human Psycho-physical Development*

The development of a human can be broadly classified into two aspects, namely: physical and mental growth. The psychological and physical development of a human varies at different stages of life and is dependent on a continuous and complex interplay between heredity and environment. Infants learn about the world through touch, sight, sound, taste and smell. They incrementally learn to make *sense* of the world and communicate with other individuals by representing knowledge in their self-explanatory ways.

Philosophers have tried to conceptualize the autonomous mental development of a growing individual, thus resulting in a controversial debate about representation in the human's mind. Also,

recent research in computational modeling of human neural and cognitive development tries to construct *autonomous intelligent* robots, hence the importance of developmental vision systems.

A brief description of human psychological and physical development is as follows:

1. *Physical development*
   - — The biological cycle of physical maturation from infants to fully grown adults.
   - — Physical development involves changes in body size, proportions, appearance and the functioning of various body systems: brain development; perceptual and motor capacities; and physical health.
   - — Nature has set a general time for muscles to mature, making it possible for a human to accomplish skills as he or she ages.
2. *Psychological development*
   - — There are various stages of growth from infancy to adulthood, formulating an individual's value system and beliefs.
   - — There are numerous psychological models suggesting discovery and assumptions of a human's mental growth, but none can explicitly describe an individual.
   - — A human developmentally learns through accumulation of various schools of knowledge at different stages of growth.

From a human's biological cycle, we can first observe that the motor skills of an individual rely on his or her mental understanding and physical (muscle) maturity. Secondly, time plays a critical role in both physical and mental development of a human. There is always a notable timeframe required for muscle maturity and acquisition of knowledge. Thirdly, a human gains his or her knowledge through an autonomous developmental system with incrementally self-generated representation.

## 2.2 Machine (Robot) Psycho-physical Development

An *intelligent* machine should have developmental visual learning, like a newborn beginning to visually adapt to his or her environment. Comparing the timeframe for physical maturity, a baby will need years to fully develop his or her body, whereas the construction of a humanoid robot can be completed within a year with the current state-of-the-art technology. Humanoid robots, hence, are capable of outperforming humans in the physical development stage. Also, robots can be constructed, based on an application's specifications, to be of varying size, mobility, strength, etc. Hence, robots are able to overcome humans' physical limitations such as build, strength, senses, etc.

Babies can adaptively learn and acquire knowledge by various means, incrementally gaining *intelligence* as they age physically. Robots, however, tend to operate in a 'single-minded' way, performing fixed routines with limited adaptive capabilities. Recent advances in artificial intelligence, cognitive science, neuroscience and robotics have stimulated the interest and growth of a new research field, known as computational autonomous mental development. This field prompts scientists to brainstorm on developing developmental learning models for robots to enable an interactive channel of communication for both robots and humans. In order to achieve beneficial dual communication for both robots and humans, developing a developmental vision system for robots becomes important. Machines in the future will hence be able to extend humans' mental capabilities when they can learn incrementally, supplementing humans in both physical and psychological development.

A brief description of machine psychological and physical development is as follows.

1. *Physical development*
   - — There is a construction cycle of physical maturity within a short span of time (from a few months to a few years). The stages involved are, e.g., design, debugging, simulation, fabrication, testing, etc.
   - — Skills are limited by their mechanisms, kinematics and dynamic constraints, etc.

2. *Psychological development*
   — There is an artificial intelligence limitation; 'rigid program routines'.
   — Machines are restricted by non-interactive knowledge acquisition.
   — This is a challenging area of research! How to shorten the time needed to acquire knowledge through lifelong developmental visual learning by robots. Humanoid robots in the future may have rapid transfer of knowledge between each other, which humans need to learn over a period of time.

## 2.3 Developmental Learning

Sony AIBO is able to learn visually and identify visual objects through interactions with a human mediator [1]. The methodology that enables visual learning by Sony AIBO is supervised learning of words and meanings through interactions between AIBO and its human masters. This methodology is grounded on the assumption of how a child represents and learns meanings through natural language. They have thus demonstrated the research potential in adaptive recognition of visual objects by robots under supervised learning.

Hilary Buxton's [2] recent article about computing conceptual descriptions in dynamic scenes lists various models of learning in a visual system. His article brought forth the philosophy that we are entering an era of more intelligent cognitive vision systems, whereby visual interaction and learning will be one of the motivations for tomorrow's technologies. An *intelligent* humanoid robot should therefore be embedded with visual interaction and learning capabilities. This is thus one of the main motivations for developing developmental vision in humanoid robots.

The initial step to wards developmental vision by robots will be allowing robots to developmentally 'learn' with whom they are conversing, i.e. allowing them to identify their human masters. This will allow humans to view robots as agents instead of simply tools for a specific task, and hence moderately overcome the 'cold metallic' feeling of humanoid robots during human–robot interaction.

Parents will typically be overjoyed if their infant/child is able to recognize them and respond positively when seen. Hence, by facilitating adaptive recognition of human faces by humanoid robots, humanoid robots would then be analogous to infants who can gradually learn to 'detect' and 'recognize' other human counterparts through visual learning. There will be elation and recognition of the humanoid robot as an agent when the interacting human master obtains the acknowledgment from the humanoid robot as a recognized person.

Supervised developmental vision is being proposed in this chapter as humanoid robots are analogous to infants, in that they are both initially untrained in their visual recognition capabilities. Infants need to be guided and supervised during their developmental mental growth, likewise when we try to enable developmental vision in humanoid robots. This will allow us to filter away any undesirable input information, simplifying the nonrequired complexities involved, until humanoid robots 'mature' in their developmental vision systems.

## 2.4 Questions to Ponder

- What is a human's physical limitation? Hence, how can a humanoid robot (machine) assist its human master?
- What is a human's mental limitation? Hence, how can a humanoid robot (machine) assist its human master?
- What are the developmental theories for humans and robots? Can developmental learning theories for humans be applied to humanoid robots and vice versa?

## 3. Developmental Learning of Facial Image Detection

Most of the face detection and recognition algorithms in early research prefer grayscale images, primarily due to their lower computational requirement compared with color images. The identification of color objects and surface boundaries comes naturally to a human observer, yet it has proven to be difficult and complicated to apply to a robot.

But segmentation based on color, instead of only intensity information, can provide an easier distinction between materials, on the condition that robustness against irrelevant parameters is achieved. In this chapter, we focus on the detection of color facial images through developmental learning by an RCE neural network. The RCE neural network is chosen due to its parallel distributed processing capability, nonlinearity, tolerance to error, adaptive nature and self-learning abilities. An adaptive color segmentation algorithm with learning ability can hence facilitate incremental color prototype learning, part of the developmental vision system of humanoid robots.

### *3.1 Current Face Detection Techniques*

Most research work inclines towards face recognition rather than face detection, as it assumes the face locations are predefined. Hence, before discussing current face detection techniques, we should ask ourselves why we need the human face detection phase (why not direct face recognition?)

Face detection is important because it is a preliminary step to following identification applications (e.g. face recognition, video surveillance, etc.). Basically, its goal is to detect the presence of a human in the image and extract salient features that are unique or necessary for subsequent processes. Then, how should we go about it when we understand the importance of why we do it?

The face detection problem can be considered initially as a binary classification problem, i.e. whether the image contains a human face (binary '1') or it contains no human face (binary '0'). This allows the reduction of computation requirements, as subsequent face recognition will be performed if and only if the presence of a human is detected [3–7]. But the binary classification problem, although simple to a human observer, poses a challenging task to robots. Hence, to achieve the objective of incorporating a developmental vision system in humanoid robots, the face detection problem has to be resolved with an algorithm capable of learning incrementally.

The main issues associated with the human face detection problem are as follows:

- variations in lighting conditions;
- different facial expressions by the same individual;
- the pose of an individual face (frontal, nonfrontal and profile views);
- noise in the images and background colors.

Since the beginning of the 1990s, various methodologies have been proposed and implemented for face detection. These methods can be classified roughly into three broad categories [8] as follows:

1. *Local facial features detection*. Low-level computer vision algorithms are applied to detect initially the presence of facial features such as eyes, mouth, nose and chin. Then statistical models of the human face are used for facial feature extraction [9–13].
2. *Template matching*. Several correlation templates are used to detect local subfeatures, which can be considered rigid in appearance [14,15].
3. *Image invariants*. It is assumed that there are certain spatial image relationships common, and possibly unique, to all facial patterns under different imaging conditions [16]. Hence, instead of detecting faces by following a set of human-designed rules, alternative approaches are proposed based on neural networks [17–21] which have the advantage of learning the underlying rules from a given collection of representative examples of facial images, but have the major drawback of being computationally expensive and challenging to train because of the difficulty in characterizing 'nonface' representative images.

Color facial image operations were considered to be computationally expensive in the past, especially in the case of real (video) images. But with rapid enhancements in the capability of computing chips, both in terms of reduction in size (portability) and increase in processing speed, color should be considered as an additional source for crucial information to allow a more reliable and efficient subsequent stage of feature extraction. Color-based approaches are hence preferred and more often investigated in recent research. Using the role of color in the face detection problem will allow us to make use of incremental learning of skin-tone colors to detect the presence of humans in images.

However, the presence of complex backgrounds and different lighting conditions pose difficulties in color-based approaches. These difficulties have resulted in researchers testing the feasibility of using combinations of different methodologies. These combinations of techniques often involve the extraction of multiple salient features to perform an additional level of preprocessing or postprocessing to minimize ambiguities that arise due to the difficulties.[22] Lists some of the major face detection techniques chronologically with some of them adopting color-based approaches.

## *3.2 Criteria of Developmental Learning for Facial Image Detection*

The objective is not to build a novel face detection algorithm and compare with available techniques, but to develop a face detection algorithm with learning capabilities (developmental learning of color prototypes). In this case, two criteria are observed:

- The algorithm should incorporate within it a learning mechanism that is purposeful for developmental learning by humanoid robots.
- The algorithm should estimate and learn representative color features of each color object, since the role of colors should not be ignored in today's context.

## *3.3 Neural Networks*

Neural networks, when introduced in the 1990s, prompted new prospects for AI and showed the potential for real-life usage, as they are thought to be closely related to human mind mapping. The original inspiration for the technique was from examination of bioelectrical networks in the brain formed by neurons and their synapses. In a neural network model, simple nodes (or 'neurons' or 'units') are connected together to form a network of nodes – hence the term 'neural network'[23].

In the past few years, artificial neural networks have been used for image segmentation because of their parallel distributed processing capability, nonlinearity, adaptive nature, tolerance to error and self-learning abilities.

In this section, a color clustering technique for color image segmentation is introduced. The segmentation algorithm is developed on the basis of a Restricted Coulomb Energy (RCE) neural network. Color clustering in $L^*a^*b^*$ uniform color space for a color image is implemented by the RCENN's dynamic category learning procedure. The property of adaptive pattern classification in the RCENN is used to solve the problem of color clustering, in which color classes are represented by both disjoint classes and nonseparable classes whose distributions overlap. To obtain the representative color features of color objects, the RCE training algorithm is extended by using its vector quantization mechanism, representative color features are selected based on 'color density distribution estimation' from the prototype color image, and stored in the prototype layer as the color prototype of a particular object. During the procedure of color image segmentation, the RCE neural network is able to generate an optimal segmentation output in either fast response mode or output probability mode.

The RCE neural network is supposed to fulfill the following objectives:

- explore the suitable color space representation for facial color image segmentation;
- build on the segmentation concept of 'color clustering by prototype learning';

- apply itself to solving the problem of disjoint and overlapping color distributions;
- implement the procedure of representative color feature extraction based on an improved RCE neural network;
- develop an adaptive segmentation algorithm with learning ability for color image segmentation;
- attempt the segmentation algorithm with the application of developmental learning of facial image detection.

## *3.4 Color Space Transformation*

There are numerous ways to represent color. In computer graphics, a common method is to have a triplet of intensity values and, by a unique combination of the three values, a distinct color can be obtained. The color space is hence a three-dimensional space that describes the distribution of physical colors [24].

The color vectors in each of these color spaces differ from one another such that two colors in a particular space are separated by a distance value that is different from the identical two colors in another space. It is by performing some linear or nonlinear transformation that a color representation can be changed from one space to another space.

The selection of color space normally involves consideration of the following factors:

- computional speed;
- how the color representation affects the image processing results;
- interaction between the color distance measures and the respective color spaces.

Many standard color spaces have been proposed and used to facilitate the analysis of color images, RGB, xyz, $L^*a^*b^*$ and HSI are some of the most commonly used color spaces in color vision.

### 3.4.1 RGB Color Space

Red, green and blue are the primary stimuli for human color perception. A color in this space is represented by a triplet of values, typically between zero and one, and is usually scaled by 255 for an 8-bit representation. The secondary colors of RGB, cyan, magenta and yellow, are formed by the mixture of two of the primaries and the exclusion of the third, as shown in Figure 12.1.

In color image processing, RGB color space (Figure 12.2) is the physical color representation, it ensures that there is no distortion of the initial color information in the RGB color space [25–27].

### 3.4.2 RGB Color Space Limitations

Although it relates very closely to the way we perceive color with the light-sensitive receptors found in our retinas, and RGB is the basic color model used in television, computers or any other medium, it cannot be used for print production.

Another limitation is that it falls short of reproducing all the colors that a human can see. The color representation in the RGB space is also sensitive to the viewing direction, object surface orientation, highlights, illumination direction, illumination intensity, illumination color and inter-reflection; hence creating problems in the color production of computer-generated graphics (inconsistent color output).

It also does not exhibit perception uniformity, which implies that the component values are not equally perceptible across the range of that value for a small perturbation to the component.

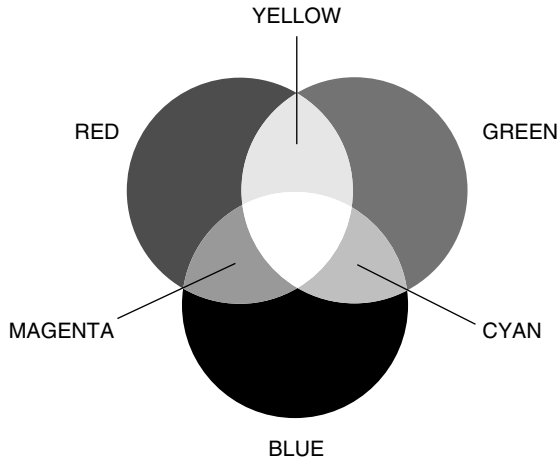



**Figure 12.1** RGB color model.

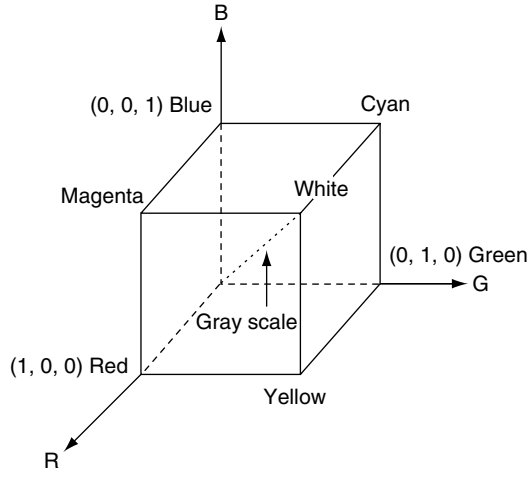



**Figure 12.2** Cartesian representation in RGB color space.

### 3.4.3 XYZ Color Space

The XYZ color space was developed by CIE as an alternative to RGB. A mathematical formula was used to convert the RGB data to a system that uses only positive integers as values. The reformulated tristimulus values were indicated as XYZ. These values do not directly correspond to red, green and blue, but approximately do so. CIE XYZ color space has the following characteristics:

- X, Y and Z are positive for all possible real stimuli.
- The coefficients were chosen such that the Y tristimulus value was directly proportional to the luminance of the additive mixture.
- The coefficients were chosen such that $X = Y = Z$ for a match to a stimulus that has equal luminance at each wavelength.

The conversion from XYZ to RGB space can result in negative coefficients; hence, some XYZ color may be transformed to RGB values that are negative or greater than one. This implies that not all visible colors can be produced or represented using the RGB system.

Since the XYZ space is just a linear translation of the RGB space, the color representation in the XYZ space is sensitive to the viewing direction, object surface orientation, highlights, illumination direction, illumination intensity, illumination color and inter-reflection, just like the RGB space.

Although all human definable colors are present in this color space, it does not exhibit perception uniformity any more than the RGB color space.

### 3.4.4 L*a*b* Uniform Color Space

L*a*b* color space [28] is a uniform color space defined by the CIE in 1976; it maps equally distinct color differences into equal Euclidean distances in space. Presently, it is one of the most popular color spaces for color measurement. In L*a*b* color space, L* is defined as lightness, and a*b* are the chromaticity coordinates. The form of the L*a*b* color space is shown in Figure 12.3.

### 3.4.5 Other Color Spaces

There are other color spaces only available to some specific applications, such as Yxy, L*c*h*, L*u*v*, YUV color spaces, etc. See the color standards of CIE [29].

### 3.4.6 Selection of Color Space For Segmentation

Keeping in mind the selection of color space as mentioned in the early part of Section 3.4, it is required to select a color space with the following properties:

- Color pixels of interest can be clustered into well-defined, less overlapping groups, which are easily bounded by segmentation algorithms in the color space.
- The color space has uniform characteristics in which equal distances on the coordinate correspond to equal perceived color differences.
- The computation of the color space transformation is relatively simple.

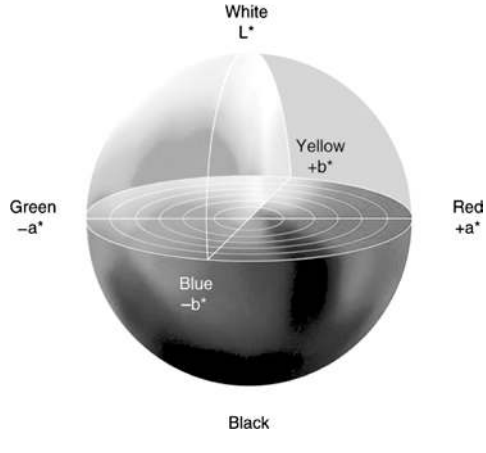



**Figure 12.3** $L^*a^*b^*$ color model.

Among the color spaces, L*a*b* uniform color space representation possesses the uniform property and demonstrates better segmentation results than others [30]. Hence, in this chapter, L*a*b* color space is selected as color coordinate for clustering-based segmentation.

### 3.4.7 RGB to L*a*b* Transformation

The transformation of RGB to L*a*b* color space is represented as follows:

1. R, G and B values in RGB primary color space are converted into X, Y and Z tristimulus values defined by CIE in 1931.

$$\begin{aligned} X &= 2.7690R + 1.7518G + 1.1300B \\ Y &= 1.0000R + 4.5907G + 0.0601B \\ Z &= 0.0000R + 0.0565G + 5.5943B \end{aligned} \tag{12.1}$$

2. L*a*b* values are obtained by a cube-root transformation of the X, Y and Z values:

$$\begin{aligned} L^* &= 116[Y/Y_0]^{1/3} - 16 \\ a^* &= 500\left[(X/X_0)^{1/3} - (Y/Y_0)^{1/3}\right] \\ b^* &= 200\left[(Y/Y_0)^{1/3} - (Z/Z_0)^{1/3}\right] \end{aligned} \tag{12.2}$$

where $X/X_0 > 0.01, Y/Y_0 > 0.01$ and $Z/Z_0 > 0.01$. $X_0, Y_0, Z_0$ are XYZ tristimulus values for reference white, which are selected as 255 for an 8-bit image data representation.

In this chapter, color image segmentation is implemented in L*a*b* color space. The RGB to L*a*b* transformation is used to convert original RGB image data into L*a*b* image data.

## *3.5 RCE Adaptive Segmentation*

The proposed framework for developmental learning of facial image detection is shown in Figure 12.4.

### 3.5.1 Color Clustering by Learning

The segmentation algorithm should not be built by presetting the segmentation threshold on the basis of color distributions of some specific color objects, as it would suffer from the problem of selecting different thresholds for different color images. An adaptive segmentation algorithm is hence proposed, whereby it segments the color image by various prototype modes derived from experience learning.

***Prototype Mode***

The prototype view of concept formation posits that a concept is represented by the summary description in which the features need to be characterized by the concept instances. In the view of the prototype mode, color image segmentation can be regarded as a procedure of color classification on the basis of

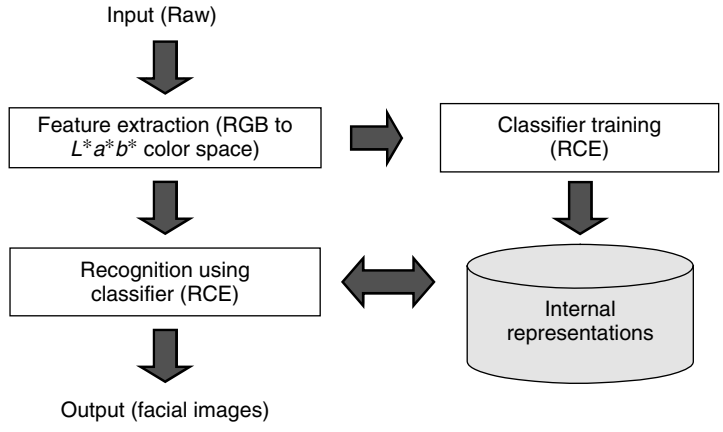



**Figure 12.4** Flowchart for developmental learning of facial image detection.

various color prototypes. Each color prototype is an abstract representation of color features for one specific color object, it represents a region of color distribution in L*a*b* color space.

Suppose one color image consists of $\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots \boldsymbol{C}_{\mathrm{n}}$ color classes (e.g. skin, hair, clothes, etc.), each color class $\boldsymbol{C}_i$ possesses a set of color prototypes $\boldsymbol{P}^i(\boldsymbol{p}^{\mathrm{i}}_1, \boldsymbol{p}^{\mathrm{i}}_2, \ldots, \boldsymbol{p}^{\mathrm{i}}_{\mathrm{m}})$ in L*a*b* color space. Define $\boldsymbol{X}$ as a point in L*a*b* color space, it refers to a pixel $\boldsymbol{S}_x$ in the color image. Then, pixel $\boldsymbol{S}_x$ is segmented into color class $\boldsymbol{C}_i$ only when the point $\boldsymbol{X}$ belongs to $\boldsymbol{P}^i$,

$$\text{If} \quad \boldsymbol{X} \in \boldsymbol{P}^i \quad \text{then} \quad \boldsymbol{S}_{\mathrm{X}} \in \boldsymbol{C}_i \tag{12.3}$$

The color prototype is defined as a spherical influence field with variable radius in L*a*b* color space and is required by learning from the training set.

### *Spherical Influence Field*

In L*a*b* color space, suppose a color class $\boldsymbol{C}_i$ possesses a set of color prototypes $\boldsymbol{P}^i(\boldsymbol{p}^{\mathrm{i}}_1, \boldsymbol{p}^{\mathrm{i}}_2, \ldots, \boldsymbol{p}^{\mathrm{i}}_{\mathrm{m}}), \boldsymbol{p}^i_j$ being one color prototype with in color class $\boldsymbol{C}_i$, then it forms a spherical influence field $\boldsymbol{F}^i_j$ with the following properties:

- $\boldsymbol{F}^i_j$ is a spherical region in L*a*b* color space.
- The center of spherical region $\boldsymbol{X}^i_j$ is called the center of the prototype.
- The radius of the spherical region, $\lambda^i_j$, is defined as the threshold of color prototype $\boldsymbol{p}^i_j$.

A color class region can be accurately bounded by spherical influence fields; they are covered with the overlapping influence fields of a prototype set drawn from the color class training set. In this case, the influence field may extend into the regions of some different color classes to the point of incorrect classification or class confusion. It can be modified by reducing the threshold $\lambda$ of the color prototype until its region of influence just excludes the disputed class. The spherical influence field is able to develop proper separating boundaries for nonlinear separable problems, as shown in Figure 12.5. Moreover, it can handle the case of nonseparable color distribution by probability estimation of color prototypes.

### *Adaptive Segmentation*

Adaptive segmentation aims to obtain the best segmentation result by adjusting the segmentation algorithm to meet the variations of segmentation objects, it requires that the algorithm has the ability

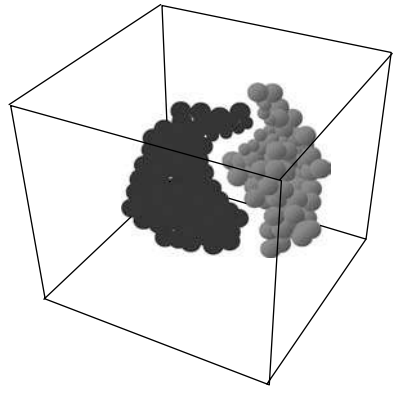


**Figure 12.5** Nonlinear distribution bounded by spherical influence fields.

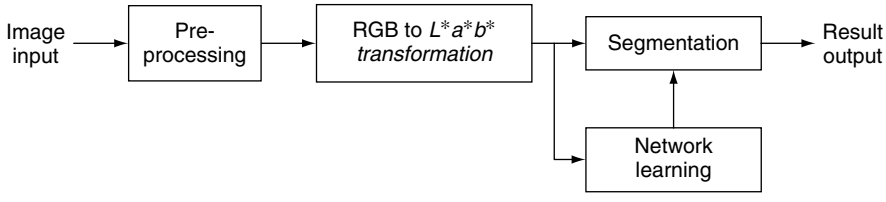



**Figure 12.6** Neural network-based adaptive segmentation.

to interact with the environment. A supervised neural network could perform this procedure due to its learning abilities; a block diagram of adaptive segmentation by a supervised neural network is shown in Figure 12.6.

The improved Restricted Coulomb Energy (RCE) neural network proposed for road/traffic color image segmentation [31,32] (initially introduced by D.L. Reilly [33]) is used to meet the objective of developmental learning of facial image detection.

### 3.5.2 RCE-based Segmentation

An RCE neural network can perform adaptive pattern classification by applying a supervised training algorithm to expand the number of color prototypes and define values for their network connections. The mechanisms of network training and classification make it applicable for implementing face/nonface color image segmentation. During the training procedure, color prototypes of color classes are input into the network for clustering, representative color features of each color class are extracted in clustering, the knowledge of color classes is then stored in the network as a set of color prototypes for segmentation. The network generates segmentation results in fast response mode or output probabilities mode, based on color prototypes and their probability estimations.

#### *RCE Network Description*

The RCE neural network is a general purpose, adaptive pattern classification engine; it is inspired by systems of charged particles in three-dimensional space [34]. In our project, the architecture of the RCE network contains three layers of neuron cells, with a full set of connections between the first and

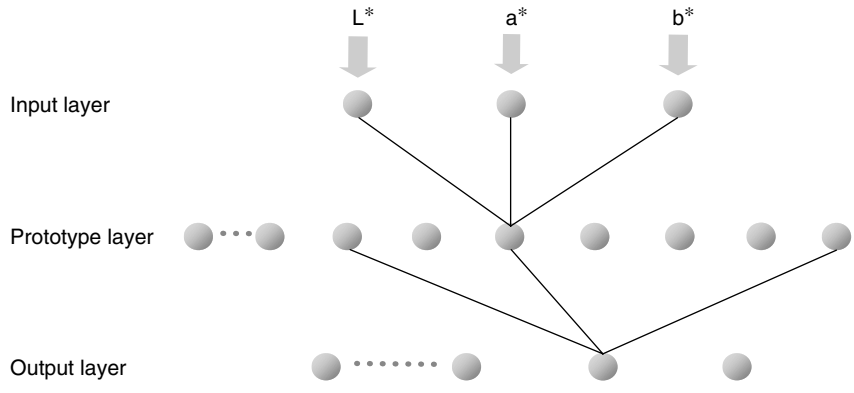



**Figure 12.7** Architecture of our RCE neural network.

second layers, and a partial set of connections between the second and third layers, its architecture is shown in Figure 12.7.

In the input layer, three cells are designed to receive the L*a*b* color values of a pixel in the color image. A middle-layer cell named the color prototype cell contains all prototypes of learned color classes that occurred in the training data. Each cell on the output layer corresponds to a different color class represented in the training data set. In the middle layer, a color prototype $P$ is characterized by five elements: color class $\boldsymbol{C}$, weight vector $\boldsymbol{W}$, cell thresholds $\lambda_{\min}$ and $\lambda_{\max}$, pattern count $\boldsymbol{K}$, and smoothing factor $\sigma$.

The color prototype cell weight vector $\boldsymbol{W}$ represents the set of weighted connections between the color prototype cell and each input layer cell. A prototype cell weight vector defines a point in L*a*b* color space, while the thresholds define the size of the spherical influence field. In response to the input signal X, each color prototype cell calculates a distance between the input values and color prototype stored in its weights, as follows:

$$d_i = \sum_{j=1}^{3} \left| W_{ij} - \mathrm{X}_j \right| \tag{12.4}$$

During network training, if $\boldsymbol{d}_i$ is less than the cell threshold $\lambda_i$, a color prototype $\boldsymbol{C}_i$ becomes active to trigger the node $\boldsymbol{O}_i$ in the output layer as follows:

$$\begin{aligned} \boldsymbol{O}_i &= 1 \quad \text{if} \quad \boldsymbol{d}_i < \lambda_i \\ \boldsymbol{O}_i &= 0 \quad \text{if} \quad \boldsymbol{d}_i \geq \lambda_i \end{aligned} \tag{12.5}$$

The pattern counter $\boldsymbol{K}$ stores the number of times the color prototype has been correctly triggered in response to its color class.

### *Network Training*

RCE network training involves three mechanisms: color prototype cell commitment, color prototype threshold modification and the increment of color prototype pattern counts. During network training, the RCE network allocates the position and size of the spherical influence field corresponding to each prototype cell by prototype cell commitment and prototype threshold modification, and covers the feature regions of each color class present in the training data. Prototype pattern counts are used to solve the problem of nonseparable color distribution.

*The First Color Prototype Commitment*

Let the first input signal $X_1$ ($L_1, a_1, b_1$) be presented to the network. The color class of $X_1$ is $C_1$, this signal causes a new color prototype cell to be committed in the network, its spherical influence field is centered in $X_1$ (see Figure 12.8). The network reactions are as follows:

1. The input signal is loaded into the weight vector of the color prototype:

$$\boldsymbol{W}_1 \leftarrow \boldsymbol{X}_1 \tag{12.6}$$

2. The color prototype is assigned a cell threshold $\lambda_1$, that is the radius of the spherical influence field:

$$\lambda_1 \leftarrow \lambda_m \tag{12.7}$$

3. A connection is made between the color prototype cell and output cell with color class $C_1$:

$$\boldsymbol{X}_1 \leftarrow \boldsymbol{C}_1 \tag{12.8}$$

It carries a pattern counter which is set to one:

$$\boldsymbol{K}_1 = 1 \tag{12.9}$$

As long as subsequent input signals with color class $C_1$ fall within the spherical influence field of the color prototype, there are no additional color prototype cells committed, and no change occurs to the spherical influence field. The pattern counter of color prototypes is incremented.

$$\text{If} \quad \text{Class}(X) = \boldsymbol{C}_1 \text{ and } \boldsymbol{O}_1 = 1 \quad \text{then} \quad \boldsymbol{K}_1 \leftarrow \boldsymbol{K}_1 + 1 \tag{12.10}$$

*Additional Color Prototype Commitments*

An input signal with color class $C_1$ that falls outside the spherical influence fields of existing color prototypes causes an additional color prototype to be committed for color class $C_1$, as shown in Figure 12.9. The same commitment process occurs as in the above description, a connection is made between this new color prototype and the output cell with color class $C_1$.

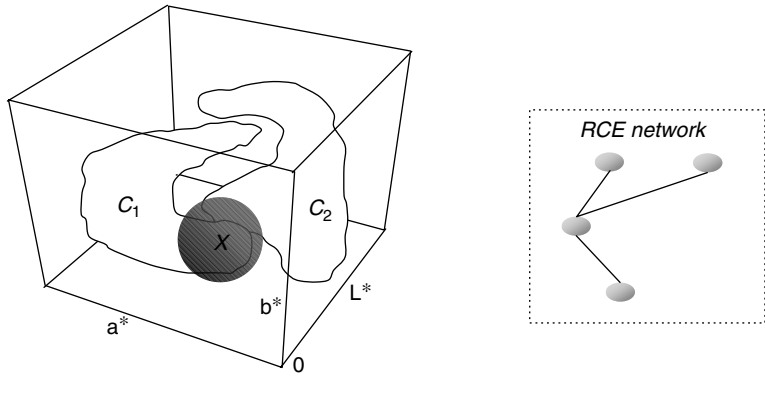



**Figure 12.8** Network response for the first input signal.

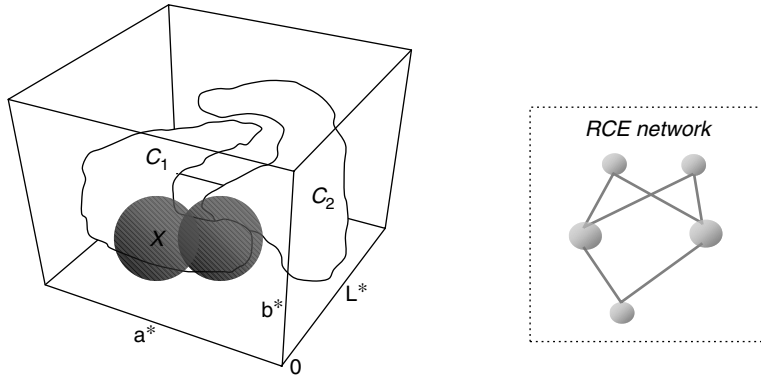



**Figure 12.9** Network response for an input signal. The signal has $\boldsymbol{C}_1$ color class, it falls outside of the existing spherical influence fields belonging to color class $\boldsymbol{C}_1$.

*Color Prototype Commitments with color class* $\boldsymbol{C}_2$

Suppose an input signal with a new color class $\boldsymbol{C}_2$ is presented to the network. Assume it falls outside the spherical influence fields of any existing class $\boldsymbol{C}_1$ color prototypes. Then, the input causes a new color prototype with color class $\boldsymbol{C}_2$ to be committed, an output cell representing color class $\boldsymbol{C}_2$ is also committed.

As we can see in Figure 12.10, the spherical influence field of the new color prototype may overlap the spherical influence fields of existing prototypes with different color classes. In this case, the initial threshold of the new color prototype is determined as follows:

$$\lambda_{\text{initial}} = \min\left(\lambda_{\max}, \max\left(\lambda_{\min}, d_{\min}\right)\right) \tag{12.11}$$

where, $d_{\min}$ is the distance to the closest opposing class prototype.

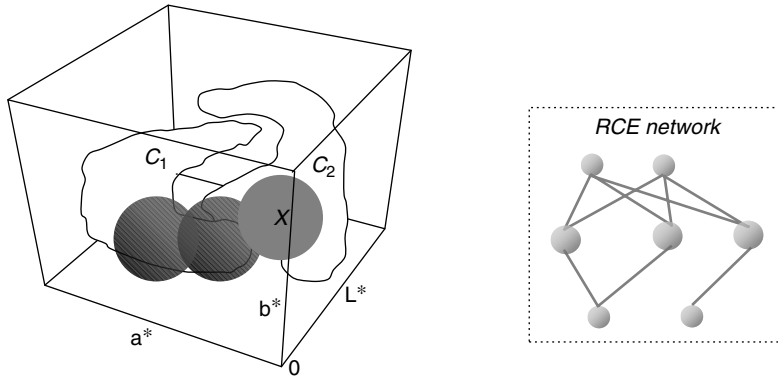



**Figure 12.10** Network response for an input signal. The signal has color class $\boldsymbol{C}_2$, it falls outside of all spherical influence fields belonging to color class $\boldsymbol{C}_1$.

*The Modification of Incorrect Prototype Commitments*

Assume a new input signal with color class $\boldsymbol{C}_1$ is presented to the network and falls into the spherical influence field of a color prototype with color class $\boldsymbol{C}_2$. The $\boldsymbol{C}_2$ color prototype is incorrectly triggered and causes the output cell with color class $\boldsymbol{C}_2$ to be triggered (Figure 12.11). In network training, this mistake is corrected by reducing the threshold of the $\boldsymbol{C}_2$ prototype to such a point that the spherical influence field of the $\boldsymbol{C}_2$ color prototype just excludes the input signal (Figure 12.12). The new threshold of $\boldsymbol{C}_2$ is modified as follows:

$$\lambda_{\text{modified}} = \max(\lambda_{\min}, d) \tag{12.12}$$

where, $d$ is the distance between the input signal and the $\boldsymbol{C}_2$ color prototype.

***Representative Color Extraction***

The color distribution of natural color objects is a non-even distribution, only the colors with high dense distributions can represent the color features of the color object, these are named representative colors.

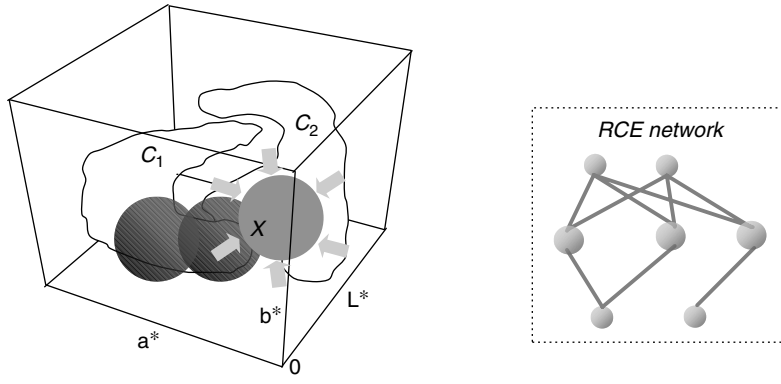



**Figure 12.11** Threshold modification of the color prototype: input signal with color class $\boldsymbol{C}_1$.

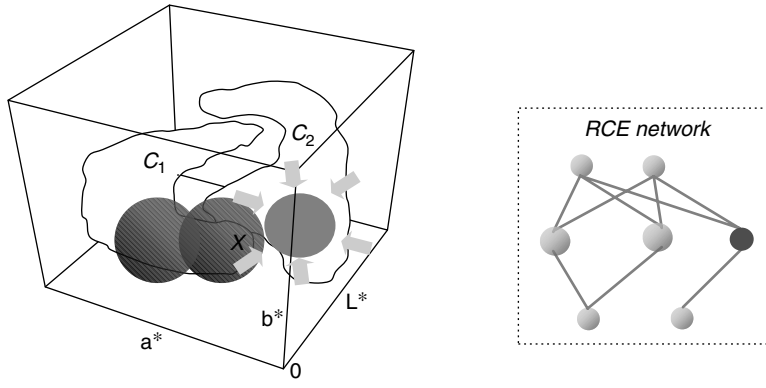



**Figure 12.12** Threshold modification of the color prototype. The threshold of the prototype with color class $\boldsymbol{C}_2$ is modified, so that the spherical influence field just excludes the input signal.

In an RCE network, each natural object is defined as one color class, its representative color can be extracted by estimating the color density of the color prototype with a given color class in the L*a*b* color space. The function of the pattern counter is extended to estimate the local density of each color prototype for a given color class in the L*a*b* color space, because the pattern counters that developed during the last training pass are the most accurate color density estimators of color classes. Suppose the network is trained by giving training sets of natural color objects until no new color prototypes are committed and no color prototypes have their cell threshold modified. Assume color class $\boldsymbol{C}_i$ represents the $i$th color object, define color prototypes $\boldsymbol{P}^i(\boldsymbol{p}_1^i, \ldots, \boldsymbol{p}_m^i)$ belonging to color class $\boldsymbol{C}_i$, their pattern counters are $\boldsymbol{K}_1, \ldots, \boldsymbol{K}_m$. The procedure of representative color extraction for color class $\boldsymbol{C}_i$ is as follows:

1. Calculate the density value $\boldsymbol{Q}_i$ of each color prototype $\boldsymbol{p}_i$,

$$\boldsymbol{Q}_i = \frac{\boldsymbol{K}_i}{\sum_{i=1}^{m} \boldsymbol{K}_i} \tag{12.13}$$

2. Select representative colors by a given threshold value $\boldsymbol{Q}_{\text{threshold}}$,

$$\begin{aligned} &\text{if} \quad \boldsymbol{Q}_i < \boldsymbol{Q}_{\text{threshold}} \quad \text{then } \boldsymbol{p}_i \text{ is reserved} \\ &\text{if} \quad \boldsymbol{Q}_i \geq \boldsymbol{Q}_{\text{threshold}} \quad \text{then } \boldsymbol{p}_i \text{ is eliminated} \end{aligned} \tag{12.14}$$

***Color Prototype Segmentation***

Facial color image segmentation is accomplished by two network response modes: *fast response mode* and *probabilities mode*.

*Fast Response Mode*

The network computes the distances $\boldsymbol{D}$ between input color signal and color prototypes in L*a*b* color space and compares these with the threshold values stored in each color prototype. If only one color prototype $p_i$ with color class $\boldsymbol{C}_i$ is responding to the input color signal, a single cell in the output layer is the response, which means,

$$\text{if} \quad D < \lambda_i \quad \text{then} \quad \boldsymbol{O}_i < 1 \tag{12.15}$$

*Output Probabilities Mode*

If multiple output cells are active, or none are active, output probabilities mode is invoked to determine the probabilities that the input signal belongs to each color class. The response of the output cell is a conditional probability estimation of the input color signal relative to each color class. To compute the output response, a decaying exponential function is defined for each color prototype:

$$\boldsymbol{f}_i = \boldsymbol{e}^{-\sigma_i \boldsymbol{D}_i} \tag{12.16}$$

where, $\boldsymbol{D}_i$ is the distance between input signal and the $i$th color prototype and $\sigma_i$ is the color prototype smoothing factor controlling the decay rate of the function.

Assume the output layer has $n$ output cells; the color class of the $j$th output cell is $\boldsymbol{C}_j$. The response of the $j$th output cell is given by:

$$\boldsymbol{O}_j = \sum_{\boldsymbol{f}_i \in \boldsymbol{C}_j} \boldsymbol{K}_i \boldsymbol{f}_i \tag{12.17}$$

The actual conditional probability of the input signal being the on $j$th output cell is computed as follows:

$$S(C_j|X) = \frac{O_j}{\sum_{j=1}^{n} O_j} \tag{12.18}$$

The output cell with maximum conditional probability generates an optimal segmentation result related to the input signal.

## 3.6 Implementation

The improved RCE network can perform adaptive segmentation by applying the supervised learning algorithm to expand the number of color prototypes and define the weights of their connections. The mechanism of network training makes it applicable to extracting representative colors of given color objects. The knowledge of color objects is then stored in the network as the reference set for segmentation. The segmentation results are generated by the network response.

### 3.6.1 Description of the Segmentation Algorithm

The procedure of facial color image segmentation by an RCE neural network includes two processes: extracting representative colors from given color objects and segmenting face/nonface color images based on the network responses. The operation of RCE-based adaptive segmentation is represented as follows:

1. Network training
   - — Select color objects from natural scenes and acquire their RGB color images. In these color images, representative colors must be included.
   - — Transform the original color space into the L*a*b* color space for each color image.
   - — Construct the training set of color objects by labeling the color class of each color object.
   - — Train the RCE neural network using the training set of color objects.
   - — Extract the representative colors of each color class from the prototype layer by density estimation of color prototypes.
2. Segmentation
   - — Acquire the face/nonface color image for segmentation and transform its RGB color space into the L*a*b* color space.
   - — Input the L*a*b* color image into the RCE neural network, segment this color image by the two RCE mode responses.

Diagrams of the adaptive segmentation algorithm are shown in Figures 12.13 and 12.14.

RCE-based adaptive segmentation represents an approach to color image segmentation with a good application potential. Its application is not confined to developmental learning of facial image detection, and can be used for color image analysis in many other fields, such as meteorology, medicine, industry, the military, etc. To facilitate the application in other fields, an adaptive segmentation program can be designed for general purposes, e.g. color learning by humanoid robots.

## 3.7 Questions to Ponder

- Why is the learning of colors important to robots?
- What are the drawbacks of neural networks?
- What other features (excluding skin segmentation) can be used for face detection (human presence detection/face detection)?

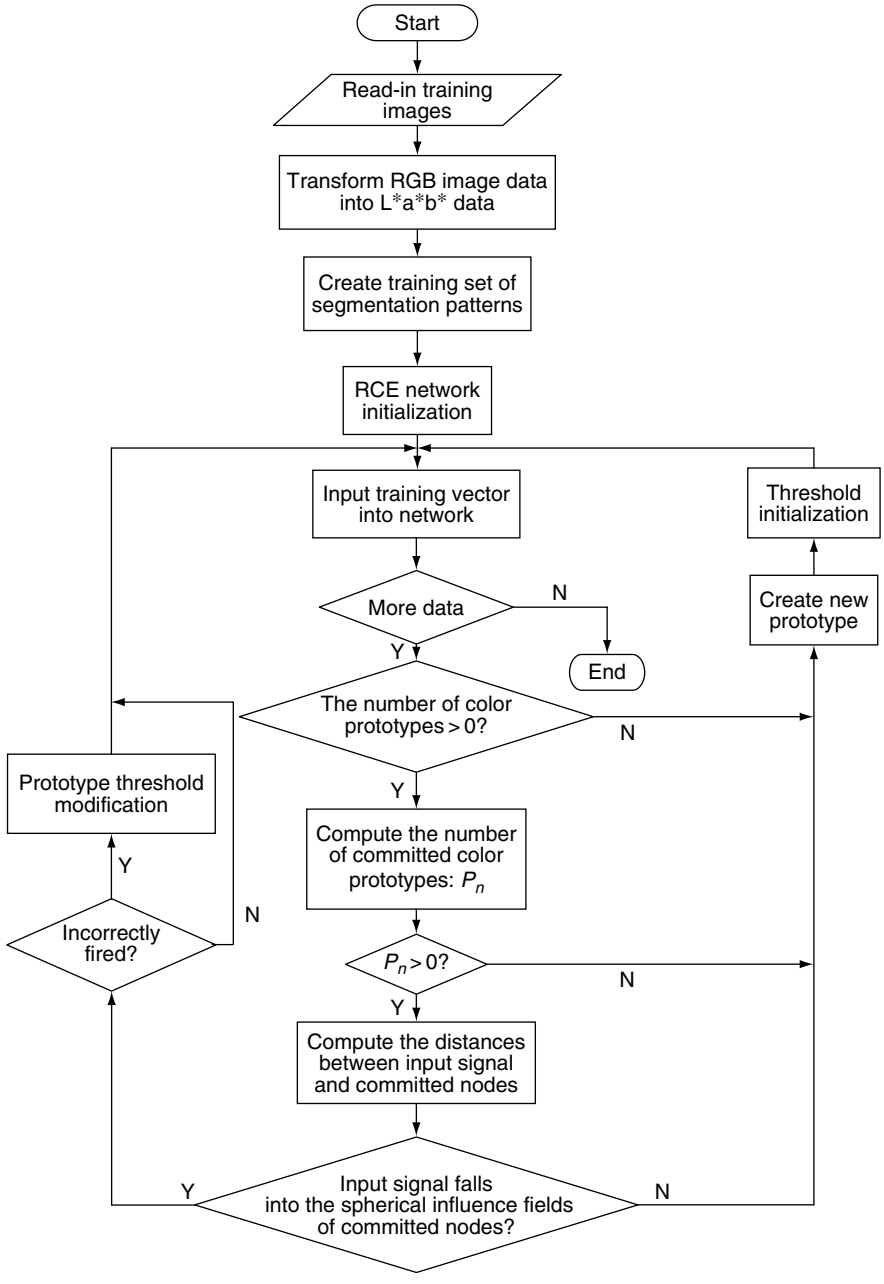



**Figure 12.13** Flowchart of the network training algorithm.

- What are the computational issues involved, i.e. the effects of the training data set, image size, etc?
- Is the detection of human presence important? In what areas will it be useful?
- What can and should be achieved during the detection phase?

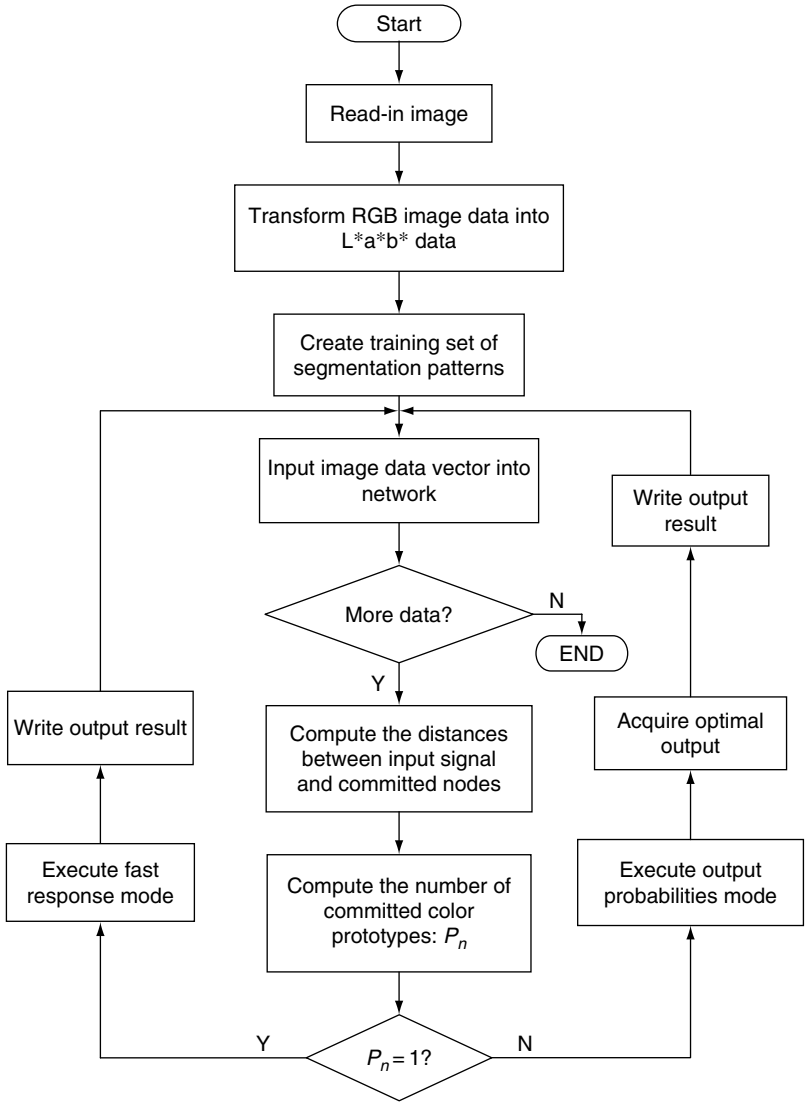



**Figure 12.14** Flowchart of the segmentation algorithm.

## *3.8 Experimental Results*

The segmentation algorithm has been performed on several color images of size $256 \times 256$. Some of them were acquired through a web-cam, representing the live capture of scenes. Figure 12.15 is selected because it can demonstrate important aspects of segmentation. The selected color images have the following color characteristics:

- The color object has a wide range of color distribution.
- The color object has color distribution with additional effects from variable illumination.

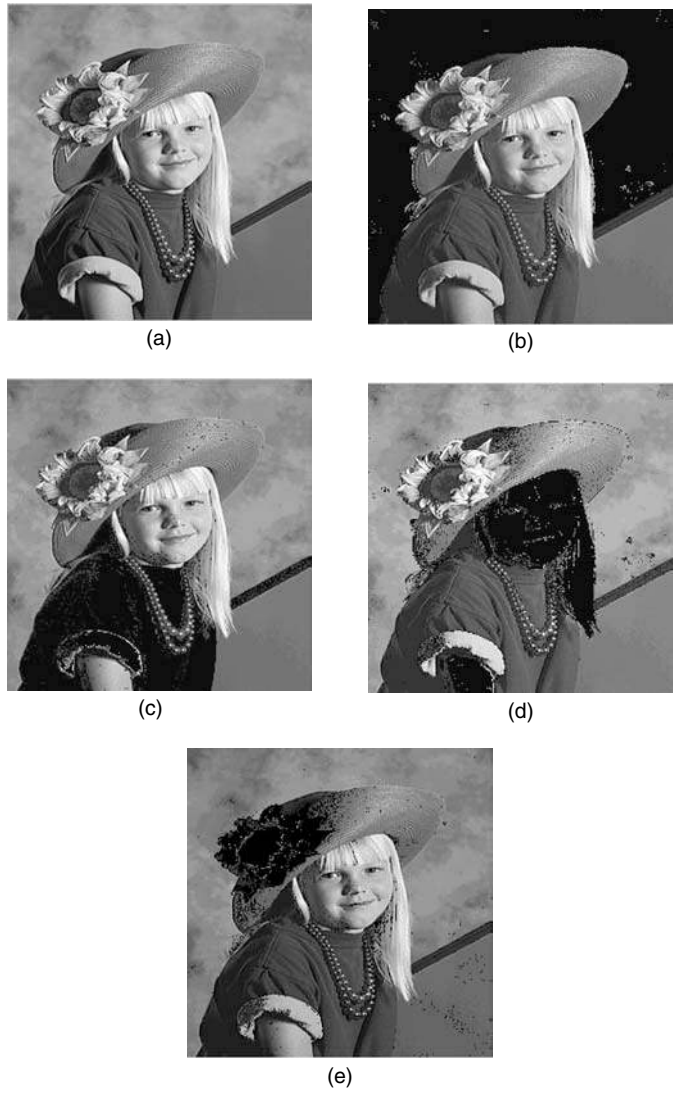



**Figure 12.15** Segmentation of a portrait. (a) Original color image; (b) Textured background segmentation; (c) clothing segmentation; (d) skin segmentation; (e) flower segmentation.

- Color objects have overlapping color distributions.
- The color object has a textured color distribution.

Segmentation results are shown in Figures 12.15 and 12.16. The experiment results state that RCE-based adaptive segmentation can succeed in segmenting different kinds of color image. The proposed segmentation algorithm is an adaptive segmentation algorithm, visual navigation is one of its most important applications. In our experiment, four images are continuously grabbed to test the performance of continuous segmentation.

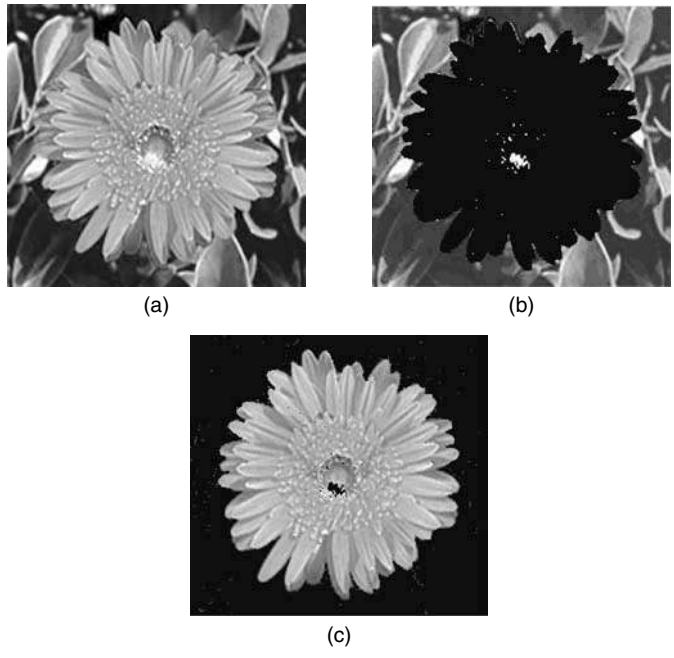



**Figure 12.16** Segmentation of plants. (a) Original color image; (b) flower segmentation; (c) leaf background segmentation.

## 4. Developmental Learning of Facial Image Recognition

There are a lot of face recognition techniques that are successful in their application areas. The objective of this section is to provide a technique for developmental learning of facial image recognition, which is a follow-up to the face detection phase. Face detection allows the detection of human presence by applying skin segmentation using an RCE neural network. The face recognition is facilitated mainly by applying wavelet packet analysis to perform feature extraction of skin-tone color areas [8,35] from the face detection results obtained in Section 3. Developmental learning of facial image recognition, however, is achieved using Hidden Markov Models (HMMs) [36,37]. The symmetric flow will be as proposed in Figure 12.4, but with different techniques involved, as shown in Figure 12.17.

### *4.1 Wavelets*

Wavelet analysis is a relatively new method, although its mathematical underpinnings date back to the work of Joseph Fourier in the nineteenth century. Fourier laid the foundations with his theories of frequency analysis, which proved to be enormously important and influential. Some of the popular wavelet families are the Haar wavelet, Daubechies wavelets, Mexican Hat wavelet, etc. What makes wavelets useful is that they have scale and time aspects. The scale aspect follows the notion of local regularity, while the time aspect studies short-time phenomena as transient processes or identification of 'domains'.

Wavelet analysis is useful in signal analysis as time information is not lost, as compared with Fourier analysis, which has this serious drawback when transforming to the frequency domain. Wavelet analysis allows the use of long time intervals where we want more precise low-frequency information, and

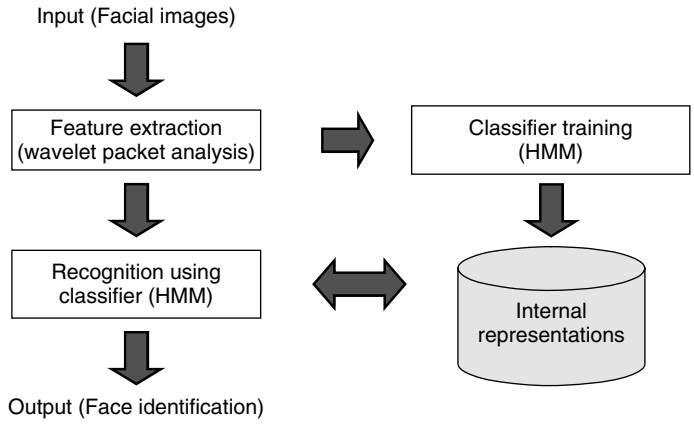



**Figure 12.17** Flowchart for developmental learning of facial image recognition.

shorter regions where we want high-frequency information, i.e. a windowing technique with variable sized regions.

A wavelet is a waveform of effectively limited duration that has an average value of zero, or it can also be defined as a function with two versions of different 'frequency' that are orthogonal to each other. More information about wavelet analysis and their applications can be found in [38–40].

Applications involving wavelet techniques can be generally classified into sound (one-dimensional), image (two-dimensional) and video (three-dimensional) processing that include compression, noise reduction, progressive transmission, etc. Wavelet analysis is hence applied in many signal processing applications, including speech and audio processing, communications, geophysics, finance, medicine, etc.

## *4.2 Wavelet Packet Analysis*

Classical wavelet decomposition divides an image into an approximation and detailed images; the approximation is then further divided into a second-level approximation and details. We are going to adopt partially the wavelet packet decomposition as proposed in [8,35], which is a generalization of the classical wavelet decomposition. In this approach, details as well as approximations can be split, resulting in a wavelet decomposition tree (Figure 12.18).

In the wavelet packet framework, compression and denoising ideas are exactly the same as those developed in the wavelet framework. The only difference is that wavelet packets offer a more complex

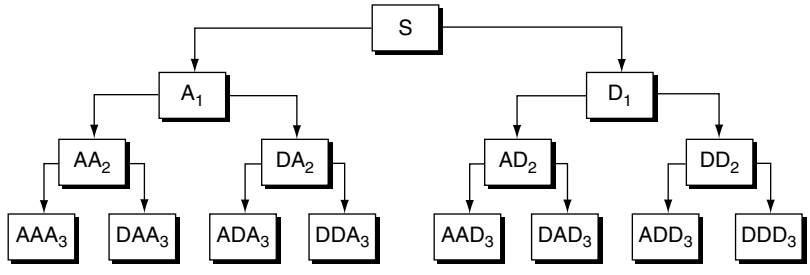



**Figure 12.18** Wavelet packet decomposition.

and flexible analysis because the details, as well as the approximations, are split (approximations are the high-scale, low-frequency components of the signal; details are the low- scale, high-frequency components).

Single wavelet packet decomposition gives a lot of bases from which you can look for the best representation with respect to a design objective by finding the 'best tree' based on an entropy criterion.

The typical wavelet packet denoising, or compression, procedure involves four steps:

1. *Decomposition*. For a given wavelet, compute the wavelet packet decomposition of signal $x$ at level $N$.
2. *Computation of the best tree*. For given entropy, compute the optimal wavelet packet tree.
3. *Thresholding of wavelet packet coefficients*. For each packet (except for the approximation), a threshold is selected and thresholding is applied to coefficients. In general, refinement of threshold is by trial and error so as to optimize the results to fit particular analysis and design criteria.
4. *Reconstruction*. Compute wavelet packet reconstruction based on the original approximation coefficients at level $N$ and the modified coefficients.

## *4.3 Feature Extraction by Wavelet Packet Analysis*

This section will discuss how to locate candidate areas in skin color filtered images, and follows the technique proposed in [8]. Garcia and Tziritas proposed building potential candidate facial areas by iteratively merging adjacent homogeneous skin color regions so that the skin color areas were not treated as a whole binary mask. Candidate facial areas, when iteratively constructed, can allow a certain level of distinguishing between different skin tones in the skin color areas. This permits a certain degree of segmenting of the faces from surrounding skin color backgrounds.

### 4.3.1 Wavelet Packet Decomposition of Facial Images

The discrete wavelet series for a continuous signal $s(t)$ is defined by the following equation:

$$c_{n,k} = \frac{1}{2^{n/2}} \int s(t)\varphi^*\left(2^{-n}t - k\right) \mathrm{d}t \tag{12.19}$$

The series use a dyadic scale factor, with $n$ the scale level and $k$ the localization parameter. The function $\varphi(t)$ is the mother wavelet which satisfies some admissibility criteria and ensures a complete, nonredundant and orthogonal representation of the signal. The Discrete Wavelet Transform (DWT) results from the above series, and is equivalent to the successive decomposition of the signal by a pair of filters $h(\cdot)$ and $g(\cdot)$, as shown in Figure 12.19. The low-pass filter $h(\cdot)$ provides the approximation of the signal at coarser resolutions, while the high-pass filter $g(\cdot)$ provides the details of the signal at coarser resolutions. Dyadic scales and positions mean choosing scales and positions based on powers of two to reduce computations.

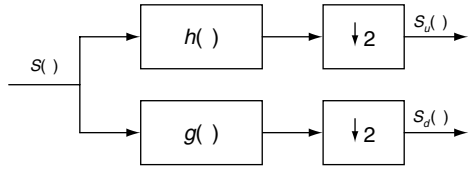



**Figure 12.19** Dyadic decomposition of a signal.

The extension to the 2D case is usually performed by applying these two filters separately in the two directions. The convolution with the low-pass filter results in a so-called *approximation* image and the convolutions with the high-pass filter in specific directions result in so-called *detail* images.

In classical wavelet decomposition, the image is split into an approximation and details. The approximation is then split into a second-level of approximation and details. For $n$-level decomposition, the signal is decomposed in the following way:

$$\begin{aligned} A_n &= \left[h_x{}^* \left[h_y{}^* A_{n-1}\right]_{\downarrow 2,1}\right]_{\downarrow 1,2} \\ D_{n1} &= \left[h_x{}^* \left[g_y{}^* A_{n-1}\right]_{\downarrow 2,1}\right]_{\downarrow 1,2} \\ D_{n2} &= \left[g_x{}^* \left[h_y{}^* A_{n-1}\right]_{\downarrow 2,1}\right]_{\downarrow 1,2} \\ D_{n3} &= \left[g_x{}^* \left[g_y{}^* A_{n-1}\right]_{\downarrow 2,1}\right]_{\downarrow 1,2} \end{aligned} \tag{12.20}$$

where $^*$ denotes the convolution operator, $\downarrow 2, 1 (\downarrow 1, 2)$ subsampling along the rows (columns), and $A_0 = I(x, y)$ is the original image. $A_n$ is obtained by low-pass filtering and is the approximation image at scale $n$. The detail images $D_{ni}$ are obtained by band-pass filtering in a specific direction ($i = 1, 2, 3$ for vertical, horizontal and diagonal directions respectively) and thus contain directional details at scale $n$. The original image $I$ is thus represented by a set of subimages at several scales: $\{A_n, D_{ni}\}$.

As stated in [8], the choice of the most suitable $h$ and $g$ filters for a given application is mainly ruled by experimental results. The constraints that have been considered in order to build these filters are closely related to the work of Smith and Barnwell concerning exact reconstruction techniques for tree-structured sub-band coders [41,42]. Hence, $h$ and $g$ are built as conjugate quadrature filters. The coefficient values and the filter support size have been chosen during series of experiments.

The $z$-transform of the pair of filters is

$$\begin{aligned} H(z) &= 0.853 + 0.377(z + z^{-1}) - 0.111(z^2 + z^{-2}) \\ &\quad -0.024\left(z^3 + z^{-3}\right) + 0.038\left(z^4 + z^{-4}\right) \\ G(z) &= -z^{-1} H\left(-z^{-1}\right) \end{aligned} \tag{12.21}$$

With the low-pass filter symmetric and the filter pair orthogonal:

$$\sum_n h(n - 2k) g(n - 2l) = 0, \forall (k, l) \tag{12.22}$$

To achieve decorrelated filter responses, two measures need to be considered.

The anti-aliasing coefficient is defined and calculated as:

$$\eta = \frac{\int_0^{1/4} |H(f)|^2 \mathrm{d}f}{\int_0^{1/2} |H(f)|^2 \mathrm{d}f} = 0.87 \tag{12.23}$$

where $H(f)$ is the Fourier transform and $h(n)$ the ideal value, equal to 1.

The correlation coefficient between the approximation and the detail signals is as follows:

$$\rho = \frac{\sum_{k,l} h(k) g(l) \gamma_s(k - lt)}{\sqrt{\sum_{k,l} h(k) h(l) \gamma_s(k - l)} \sqrt{\sum_{k,l} g(k) g(l) \gamma_s(k - l)}} \tag{12.24}$$

where $\gamma_s(\cdot)$ is the auto-covariance function of the input signal. For a noncorrelated (white noise) or fully correlated input signal, it is obvious that $\rho$ is zero valued. $\rho$ is calculated under the assumption of a first order Markov process for $s(t)$ and the maximum value of $\rho$ is computed as 0.06 [8].

The wavelet packet decomposition that is performed in the proposed approach [8] is a generalization of the classical wavelet decomposition that offers a richer signal analysis (discontinuity in higher derivatives, self-similarity, etc.) as the details, as well as the approximations, can be split. This will result in a wavelet decomposition tree applied in Matlab, as shown in Figure 12.20, with the correspondent wavelet coefficients of level 2 displayed in Figure 12.21.

The depth of the wavelet packet decomposition tree is ruled by the size of the areas which are processed. A database of manually extracted facial areas which contains a large number of different cases (size, chrominance, intensity, position and orientation) has to be built for study. The recommended database criteria by Garcia and Tziritas are as follows [8]:

- The processed frames have a size of $288 \times 352$ pixels.
- The samples are to be classified into two categories according to their respective sizes to be used in estimating the prototype vectors. A face is classified as *medium* if its height is smaller than 128, and as *large* if its height is bigger than 128.
- Candidate face intensity images are decomposed with discrete wavelet packet analysis until level 3 for large areas and until level 2 for medium areas. Deeper decomposition will not be required as the coefficient images will become too small to provide more valuable information.

According to its size, a candidate facial image is described by a set of wavelet coefficient matrices belonging to the deepest level of decomposition ($n = 16$, i.e. one approximation image and 15 detail images for *medium* areas and $n = 64$, i.e. one approximation image and 63 detail images for *large* areas) which will represent quite a huge chunk of information (equivalent to the size of the input image). Dimensionality reduction is performed by extracting discriminatory information from these images.

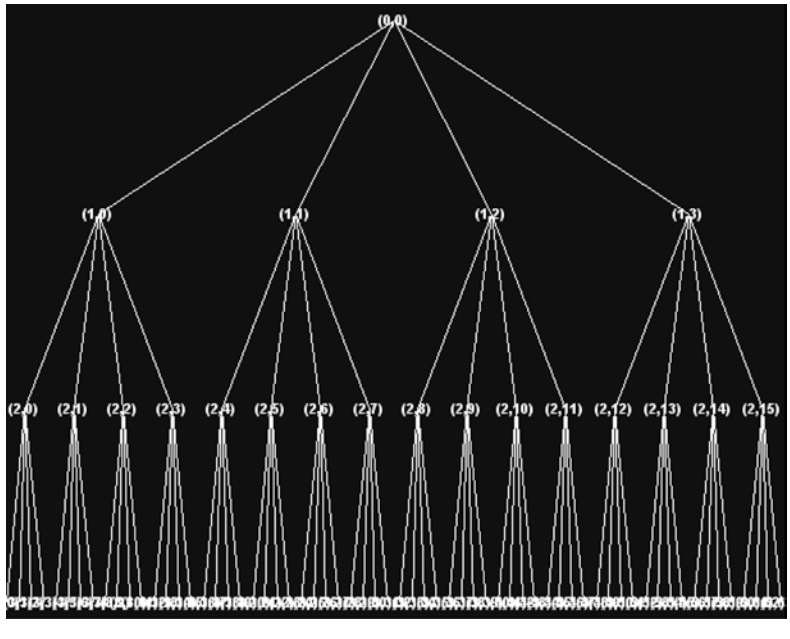



**Figure 12.20** Wavelet packet tree.

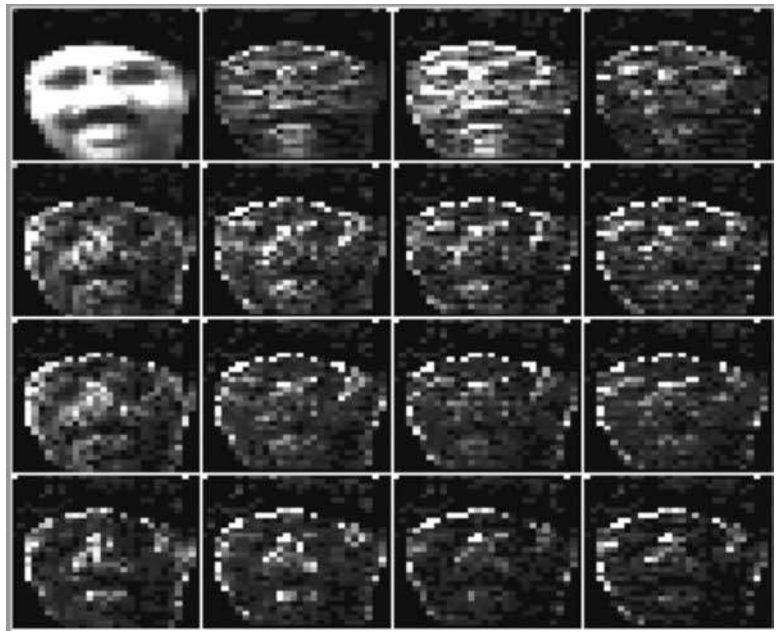


**Figure 12.21** Level 2 coefficients of the wavelet packet tree.

### 4.3.2 Experimental Results

In our approach to feature extraction, we extracted statistical information from the wavelet coefficients after performing wavelet packet decomposition on various skin-tone segmentation areas of the approximation image, and statistical information from the wavelet coefficients of each whole detail image.

We first located the face bounding box, but we did not divide it into multiple areas as proposed in [8,35] to perform the wavelet packet analysis. We treated it as a whole single entity for simplicity and investigation purposes. By extracting moments from wavelet coefficients in these areas, we were able to obtain information about the face texture, related to different facial parts like the eyes, the nose and the mouth, and including facial hair, all in one context. Standard deviations have been chosen as face texture descriptor coefficients. Hence, the corresponding standard deviations of the wavelet coefficients contained in the approximation image of the selected level of decomposition were extracted. We used images of size $100 \times 120$ pixels instead of $288 \times 352$ pixels recommended in [8] to further reduce computation. The result of the feature extraction is a two-dimensional array of vectors $O(x, y)$ with dimensionality 15 where,

$$O(x, y) = \text{A 2D observation sequence to be used for the face recognition process with pseudo 2D hidden Markov models (see Section 4.4.2 and [36]).}$$

## *4.4 Hidden Markov Models*

Preliminary studies were made using Hidden Markov Models (HMMs) to facilitate learning and feature classification for facial image recognition. The HMMs are finite stochastic automata used for the stochastic modeling of nonstationary vector time series. They were initially introduced and studied by

Baum and his colleagues in the late 1960s and early 1970s [43]. With the increasingly widespread understanding of HMM theory, it has been successfully used for pattern recognition in time series such as speech [36] and action recognition [44], where the data are modeled as one-dimensional vectors. In the field of computer vision, researchers have recently started to consider the application of stochastic models in space series, one-dimensional HMMs were used for image recognition (e.g. handwriting recognition [45], optical character and text recognition [46], face recognition [47–49], shape classification [50], etc.)

Although successful recognition performances were given, it was limited by the fact that only one dimensional shift was allowed in the image, an accurate alignment was still required in another dimension, which was guaranteed by constraining the training and test set. In this case, the 2D structure of HMMs was motivated to enhance the recognition performance. The fully connected 2D HMM was investigated by Levin, who formulated the image alignment problem as a Dynamic Plane Warping (DPW) problem [51], this led to the algorithm running in exponential time related to the dimension of the image, hence making the model impractical for real applications.

With its assumption that one dimensional distortion is independent of another dimensional distortion, the Pseudo Two-Dimensional Hidden Markov Model (P2DHMM) was able to facilitate the algorithm running in polynomial time.

P2DHMM has gained common interest in the community of computer vision due to its two-dimensional/elastic matching property in the image domain. It was first proposed for keyword recognition in poorly printed documents by Kuo in 1994 with a recognition accuracy of 96 % [45]. Realizing its promising performance on image alignment, [47] and [49] proposed to apply P2DHMM for face recognition in their PhD theses, and Nefian reported the accuracy of face recognition could reach 98 % based on P2DHMM models trained with 2D DCT coefficient features [52]. Rigoll revealed another important potential of image segmentation: that an integration of target segmentation and classification is possible even within a complex image background [53].

### 4.4.1 The One-Dimensional Hidden Markov Model

Deterministic and stochastic models are two types of signal model characterizing the properties of a given signal. The latter attempts to characterize the statistical properties of the signal when the signal can be well characterized as a parametric random process. The Hidden Markov Model (HMM) is one such stochastic model which had good success in modeling the speech process.

#### *Markov Chains*

Consider a system whose evolution may be described at any time as being in one of a set of $N$ distinct states, $\{S_n, n = 1, 2, \ldots, N-1, N\}$. The system undergoes a change of state (possibly back to the same state) according to a set of probabilities associated with the state. Let $t = 1, 2, \ldots$ denote the time instants associated with state changes, and $q_t$ denote the actual state at time $t$. If the evolution of the system is such that the conditional probability distribution of $q_t$ depends only on the value of $q_{t-1}$ and is independent of all previous values, the process is called a *Markov chain*, with the Markov property as follows:

$$P\left(q_t = S_{j|q_{t-1}} = S_{i,q_{t-2}} = S_k, \ldots\right) = P\left(q_t = S_{j|q_{t-1}} = S_i\right) \tag{12.25}$$

Therefore, a sequence of random variables $S_1, S_2, \ldots, S_{n-1}, S_n$ forms a Markov chain only if the probability that the system is in state $q_t$ at time $t$ depends exclusively on the probability that the system is in state $q_{t-1}$ at time $t-1$. In a Markov chain, the transition from one state to another is probabilistic. Let $a_{ij}$ denote the transition probability from state $i$ at time $t-1$ to state $j$ at time $t$,

$$a_{ij} = P\left(q_t = S_{j|q_t-1} = S_j\right) \tag{12.26}$$

which are conditional probabilities with the following properties:

$$\begin{aligned} p_{ij} &\geq 0 \quad \text{for all } (i, j) \\ \sum_j p_{ij} &= 1 \quad \text{for all } i \end{aligned} \tag{12.27}$$

The Markov chain may be regarded as a generative model (Markov model), consisting of a number of states linked together on a pair-wise basis for possible transitions. Each time a particular state is visited, the model outputs the symbol associated with that state.

### *Hidden Markov Models*

The Markov model is observable when each state of the model corresponds to an observable (physical) event, but this model is too restrictive to be applicable for many problems. The Hidden Markov Model (HMM) is a doubly embedded stochastic process with an underlying stochastic process that is not observable (hidden), but can only be observed through another set of stochastic processes that produce the sequence of observations. The one-dimensional HMM is a Markov chain with a finite number of unobservable states. Although the Markov states are not directly observable, each state has a probability distribution associated with the set of possible observations. The observation is determined according to a given conditional probability density function, e.g. Gaussian or Gaussian mixture. The HMM model is characterized by an initial state probability matrix, a state transition probability matrix and a set of probability density functions associated with the observations for each state.

The HMM provides a statistical model for a set of observation sequences. Let $o_1, o_2, \ldots, o_t$ denote an observation sequence with length $T$. An HMM consists of a sequence of states numbered 1 to $N$ which is best understood as a generator of observations. The states are connected together by arcs and lines (Figure 12.22), and one state is entered each time. An observation is generated according to the multivariate Gaussian distribution $b_j(o_t)$ with mean $\mu$ and covariance matrix $\boldsymbol{V}_j$ associated with that state. The arcs have transition probabilities associated with them such that a transition from state $i$ to state $j$ has the probability $a_{ij}$. The initial probability of the model being in state $j$ is $\pi_j$. Then, the HMM model is characterized by the following notation:

$N$ : The number of states in the model.
$M$ : The number of distinct observation symbols.
$T$ : The length of the observation sequence.
$\boldsymbol{V} = \{v_1, \ldots, v_M\}$ : The discrete set of possible observation symbols.
$\boldsymbol{O} = \{o_1, \ldots, o_T\}$ : The observation sequence with length $T$, where $o_t$ is the observation symbol observed at time instant $t$.
$\boldsymbol{Q} = \{q_1, \ldots, q_T\}$ : The state sequence with length $T$, where $q_t$ is the actual state at time instant $t$.
$\Pi = \{\pi_i\}$ : The initial state probability distribution, where $\pi_i = P\left(q_1 = S_j\right)$.

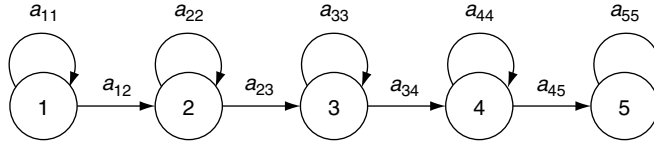



**Figure 12.22** The five-state hidden Markov model.

$\boldsymbol{A} = \{a_{ij}\}$ : The state transition probability distribution of being in state $j$ at time $t$ given that we were in state $i$ at time $t-1$ :
$\boldsymbol{B} = \{b_j(k)\}$ : The observation symbol probability distribution in state $j\,(1 \le j \le N, 1 \le k \le M)$, where $b_{j(k)}$ is the probability of observing the symbol $v_k$ given that we are in state $j$.
$\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$ : The compact notation of an HMM model.

Given the form of the HMM, three main problems must be solved for the model to be useful in real applications, these are:

1. *The evaluation problem*: Given an HMM model $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$ and the observation sequence $\boldsymbol{O} = \{o_1, \ldots, o_T\}$, how do we compute the probability of this observation sequence $P(\boldsymbol{O}|\lambda)$ efficiently?
2. *The decoding problem*: Given an HMM model $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$ and the observation sequence $\boldsymbol{O} = \{o_1, \ldots, o_T\}$, how do we choose a corresponding state sequence $\boldsymbol{Q} = \{q_1, \ldots, q_T\}$, such that the joint probability of the observation and state sequences is maximized?
3. *The learning problem*: How do we adjust the HMM model parameters $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$ such that $P(\boldsymbol{O}|\lambda)$ or $P(\boldsymbol{O}, \boldsymbol{I}|\lambda)$ is maximized?

The evaluation problem allows us to choose the model which best matches the observations; the decoding problem attempts to uncover the hidden part of the model, i.e. to find the optimal state sequence; and the learning problem optimizes the model parameters to best describe how a given observation sequence comes out. The first two problems are analysis problems, while the last problem is a typical model training or identification problem. Much detail can be found in [47–49].

### *The Forward–Backward Algorithm*

Given the HMM model $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$, the probability of the observation sequence $\boldsymbol{O} = \{o_1, \ldots, o_T\}$ can be simply computed by enumerating every possible state sequence of length $T$, but such a direct computation will require $(2T-1)N^T$ multiplications and $N^T - 1$ additions, which is computationally unfeasible, even for small values of $N$ and $T$. An efficient computation is implemented by the forward–backward procedure [36].

#### *The Forward Procedure*

Define the forward variable $\alpha_t(i)$ as:

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = S_i|\lambda) \tag{12.28}$$

which is the probability of the partial observation sequence $o_1, o_2, \ldots, o_T$ up to time $t$ and state $S_i$ at time $t$. Given the model $\lambda$, it can be solved inductively as follows:

1. Initialization:

$$\alpha_t(i) = \pi_i b_i(o_1), \quad 1 \le i \le N \tag{12.29}$$

2. Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}) \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix} \tag{12.30}$$

3. Termination:

$$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{12.31}$$

Step 1 initializes the forward probabilities as the joint probability of state $S_i$ and initial observation $o_1$. Induction computes the probability of partial observation sequence up to time $t+1$ and state $j$ at time $t+1$, the state $j$ can be reached independently from any of the $N$ states at time $t$ with the probability $a_{ij}$. In step 3, all possible ways of realizing the given observation sequence are summed up. The computation of the forward procedure requires only $N(N+1)(T-1)+N$ multiplications and $N(N-1)(T-1)$ additions, and involves of the order of $N^2T$ computations. To solve the evaluation problem, only the forward procedure is required, the backward procedure is used to help in solving the learning problem.

### *The Backward Procedure*

In a similar manner, define a backward variable $\beta_t(i)$ as:

$$\beta_t(i) = P\left(o_{T+1}, o_{T+2}, \ldots, o_T, q_t = S_i \,|\, \lambda\right) \tag{12.32}$$

which is the probability of the partial observation sequence from $t+1$ to the end. Given the state $S_i$ at time $t$ and the model $\lambda$, it can be solved inductively as follows:

1. Initialization:

$$\beta_t(i) = 1, \qquad 1 \le i \le N \tag{12.33}$$

2. Induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \qquad \begin{array}{l} t = T-1, T-2, \ldots, 1 \\ 1 \le i \le N \end{array} \tag{12.34}$$

The initial step arbitrarily defines $\beta_T(i)$ to be 1 for all $i$. The induction step shows that in order to have been in state $S_i$ at time $t$, and to account for the observation sequence from time $t+1$ on, you must consider all possible states $S_j$ at time $t+1$, accounting for the transition from $S_i$ to $S_j$–$a_{ij}$, as well as the observation $o_{t+1}$ in state $j$–$b_j(o_{t+1})$, and then account for the remaining partial observation sequence from state $j$–$\beta_{t+1}(j)$.

### *The Viterbi Algorithm*

Given the HMM model $\lambda$, the decoding problem is solved by searching the state sequence $\boldsymbol{Q} = \{q_1, q_2, \ldots, q_T\}$ such that the occurrence probability of the observation sequence $\boldsymbol{O} = \{o_1, o_2, \ldots, o_T\}$ from this sequence is greater than that from any other state sequence. There are several possible ways to search the optimal state sequence, depending on the optimality criteria of the state sequence.

The most widely used criterion is to find the single best state sequence to maximize $P(\boldsymbol{Q}\,|\,\boldsymbol{O}, \lambda)$ which is equivalent to maximizing $P(\boldsymbol{Q}, \boldsymbol{O}\,|\,\lambda)$. The Viterbi Algorithm is used for finding the single best state sequence that exists, based on dynamic programming. It is an inductive algorithm in which at each time instant you keep the best possible state sequence for each of the $N$ states as the intermediate state for the desired observation sequence $\boldsymbol{O} = \{o_1, o_2, \ldots, o_T\}$. In this way, the best path for each of the $N$ states as the last state for the desired observation sequence is found, and the one with the highest probability is selected.

#### *The Reformulation Procedure*

Given the probability of the observation sequence $\boldsymbol{O}$ for the state sequence $\boldsymbol{Q}$ – $P(\boldsymbol{O}\,|\,\boldsymbol{Q}, \lambda)$, the joint probability of observation and state sequences – $P(\boldsymbol{O}, \boldsymbol{Q}\,|\,\lambda)$

$$P(\boldsymbol{O}, \boldsymbol{Q}\,|\,\lambda) = P(\boldsymbol{O}\,|\,\boldsymbol{Q}, \lambda) P(\boldsymbol{Q}\,|\,\lambda)$$

is derived by:

$$\pi_{q_1} b_{q_1} a_{q_1 q_2} b_{q_2}(o_2), \ldots, a_{q_{T-1} q_T} b_{q_T}(o_T) \tag{12.35}$$

Define $\boldsymbol{U}(q_1, q_2, \ldots, q_T)$ such that

$$\boldsymbol{U}(q_1, q_2, \ldots, q_T) = -\left[\ln\left(\pi_{q_1} b_{q_1}(o_1)\right) + \sum_{t=2}^{T} \ln\left(a_{q_{t-1} q_t} b_{q_t}(o_t)\right)\right] \tag{12.36}$$

Then $P(O, Q|\lambda)$ can be written as

$$P(O, Q|\lambda) = \exp\left(-U(q_1, q_2, \ldots, q_T)\right) \tag{12.37}$$

The problem of optimal state estimation, namely,

$$\max_{\{q_t\}_{t=1}^{T}} P(\boldsymbol{O}, q_1, q_2, \ldots, q_T|\lambda) \tag{12.38}$$

becomes equivalent to

$$\min_{\{q_t\}_{t=1}^{T}} \boldsymbol{U}(q_1, q_2, \ldots, q_T) \tag{12.39}$$

The Viterbi Algorithm is used to find the optimum state sequence. Suppose the system is currently in state $i$ and is going to visit state $j$ next. The weight on the path from state $i$ to state $j$ is $-\ln(a_{ij} b_j(o_t))$, which is the negative logarithm of the probability of going from state $i$ to $j$ and selecting the observation symbol $o_t$ in state $j$. The weight of state sequence is defined as the sum of the weights on the adjacent states, which means the multiplication of the corresponding probabilities. Thus, searching for the optimal sequence means finding the path with the minimum weight, as follows:

1. Initialization:

$$\begin{aligned} &\delta_1(i) = -\ln(\pi_i) - \ln(b_i(o_1)), \qquad 1 \le i \le N \\ &\psi_1(i) = 0 \end{aligned} \tag{12.40}$$

2. Recursion:

$$\begin{aligned} &\delta_t(j) = \min_{1 \le i \le N} \left\lfloor \delta_{t-1}(i) - \ln(a_{ij}) \right\rfloor \quad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix} \\ &\psi_t(j) = \arg\min_{1 \le i \le N} \left\lfloor \delta_{t-1}(i) - \ln(a_{ij}) \right\rfloor \quad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix} \end{aligned} \tag{12.41}$$

3. Termination:

$$\begin{aligned} P^* &= \min_{1 \le i \le N} [\delta_T(i)] \\ q_T^* &= \arg\min_{1 \le i \le N} [\delta_T(i)] \end{aligned} \tag{12.42}$$

4. Path backtracking:

$$q_t^* = \psi_{t+1}\left(q_{t+1}^*\right), \quad t = T-1, T-2, \ldots, 1 \tag{12.43}$$

The optimal state sequence is $\boldsymbol{Q}^* = \{q_1^*, q_2^*, \ldots, q_T^*\}$ with the probability of $\exp(-p^*)$. The computation of the Viterbi Algorithm has the order of $N^2 T$, similar to the forward–backward procedure.

### 4.4.2 The Pseudo Two-Dimensional Hidden Markov Model

The one-dimensional HMM may be extended to a two-dimensional structure by allowing each of its states to be an HMM. The P2DHMM consists of a set of superstates; each superstate includes an embedded HMM. In this way, superstates may be used to model two- dimensional data along one direction, and embedded HMMs model the data along the other direction. The P2DHMM is a pseudo 2D model, which means transition between states in different superstates is not allowed, because a fully connected 2D HMM model leads to the model running in exponential time. The difference compared to a real 2D HMM is that the state sequences in the columns are modeled independently of the state sequences of neighboring columns. A pseudo 2D HMM is actually a nested one-dimensional HMM.

The P2DHMM is a stochastic automata with a special two-dimensional arrangement of the states; one of its structures is outlined in Figure 12.23. The states along the vertical direction are superstates; each superstate consists of a one-dimensional HMM along the horizontal direction.

In the horizontal direction, transitions are only allowed among the states of the superstate. In the vertical direction, transitions occur among the superstates. The P2DHMM model is characterized by the following notation:

$N$ : The number of superstates in the vertical direction.

$\bar{\boldsymbol{S}} = \{\bar{S}_j : 1 \le j \le N\}$ : The superstate set.

$\boldsymbol{A} = \{a_{kj} : 1 \le k, j \le N\}$ : The superstate transition probability matrix, where $a_{kj} = P\left(q_{y+1} = \bar{S}_{j|q_y} = \bar{S}_k\right)$.

$\Pi = \{\pi_j : 1 \le j \le N\}$ : The initial superstate probability distribution, where $\pi_j = P(q_1 = S_j)$.

$\Lambda = \{\lambda^j : 1 \le j \le N\}$ : The parameter set of left–right 1D HMMs within each superstate. In the *jth* superstate, each $\lambda^j$ is specified by the standard 1D HMM parameters as follows :

$N^j$ : The number of states in the horizontal direction with the *jth* superstate.

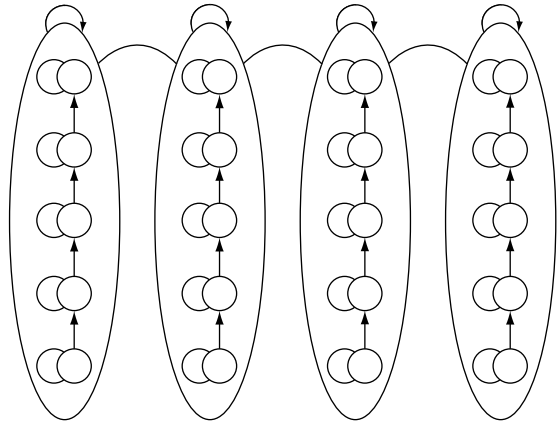


**Figure 12.23** The P2DHMM model with four superstates.

$$\boldsymbol{A}^j = \left\{ a_{ki}^j : 1 \le k, i \le N^j \right\} : \text{The state transition probability matrix,}$$
$$\text{where } a_{kj}^j = P(q_{x+1,y} = S_j^i \,\big|\, q_{xy} = S_k^j).$$
$$\boldsymbol{B}^j = \left\{ b_i^j \left(o_{xy}\right) : 1 \le i \le N^j \right\} : \text{The observation symbol probability distribution for each state,}$$
$$\text{where } b_i^j \left(o_{xy}\right) = P(o_{xy} \,\big|\, q_{xy} = S_i^j).$$
$$\Pi^j = \left\{ \pi_i^j : 1 \le i \le N^j \right\} : \text{The initial state probability distribution,}$$
$$\text{where } \pi_i^j \left(o_{xy}\right) = P(q_{1y} = S_i^j).$$
$$\eta = \{N, \boldsymbol{A}, \Pi, \Lambda\} : \text{The compact notation of a P2DHMM model.}$$

A pseudo 2D HMM for each person consists of a pseudo 2D lattice of states, each describing a distribution of feature vectors belonging to a particular area of the face. Samaria [47] used a multivariate Gaussian as a model of the distribution of feature vectors for each state. During testing, an optimal alignment of the states was found for a given image (i.e. the likelihood of each pseudo 2D HMM was maximized). Person identification was achieved by selecting the pseudo 2D HMM which obtained the highest likelihood. Due to the alignment stage, the pseudo 2D HMM approach is inherently robust to translation.

## *4.5 Feature Classification by Hidden Markov Models*

The primary objective is to introduce HMM as one of the techniques to enable developmental learning of facial image recognition. It is considered to be very suitable for face recognition due to the two-dimensional warping capabilities of P2DHMMs [37]. Also, like artificial neural networks, HMMs can learn from training data. With an HMM for feature classification, using a wavelet feature map as the 2D observation sequence $O(x, y)$ from the input image, the proposed classification approach partially follows Eickeler's [37] proposed statistical classification approach, with slight changes to the selection of algorithm for training the HMM for each person.

For *training (learning)* purposes, i.e. to train the HMM for each person, the Viterbi Procedure is applied to determine the most likely state sequence of an HMM $\lambda$, and to determine the probability of each face model for the test image for a given observation sequence. It will be able to provide the HMM parameters corresponding to a local maximum of the likelihood function, depending on the initial model parameters [36] (Section 4.4.1, using Equations (12.35) to (12.43)).

For *recognition*, the probability of an observation sequence $O(x, y)$ for a given face model is computed by the forward–backward procedure [36] to determine the probability of $\lambda$ (Section 4.4.1, using Equations (12.28) to (12.34)).

Through *training* using HMMs, humanoid robots can 'learn' to incrementally recognize facial images of human individuals, facilitating the first step to an effective dual-nature interaction. It will be analogous to young children forming 'feature maps,' i.e. distinctive features in their minds, and performing 'probabilistic optimization' to relate the input facial image with known individuals' facial images. The humanoid robot, likewise, should foremost establish a platform for developmental vision (lifelong acquisition of knowledge) by equipping itself with the capabilities to model, optimize and represent information.

## *4.6 Questions to Ponder*

- What are the components of a full face recognition system? [54]
- What other methodologies can be used for feature extraction (e.g. Gabor wavelets, eigenfaces, etc.) and feature classification (Gaussian mixture models, Kalman filters, etc.) in face recognition systems?
- What are the critical issues in face recognition systems (training set size, initial model)? Can they developmentally learn? How can they be implemented into humanoid robots?

- Why does a face recognition system not use a single technique? Can the detection and recognition techniques be combined as one?
- On what grounds can a user justify that his/her face recognition system is robust? [55]

## *4.7 Experimental Results*

The presented facial image recognition system was used on facial images captured using a web-cam in the AIM laboratory at Nanyang Technological University. Multiple facial images for each of ten people were taken at different times with slightly varying lighting conditions and different facial expressions. The heads of the people in the images were slightly tilted or rotated. Four images of each person were used for *training* of the models, while any images could be used for testing. Figure 12.24 shows the flowchart of the proposed face recognition system.

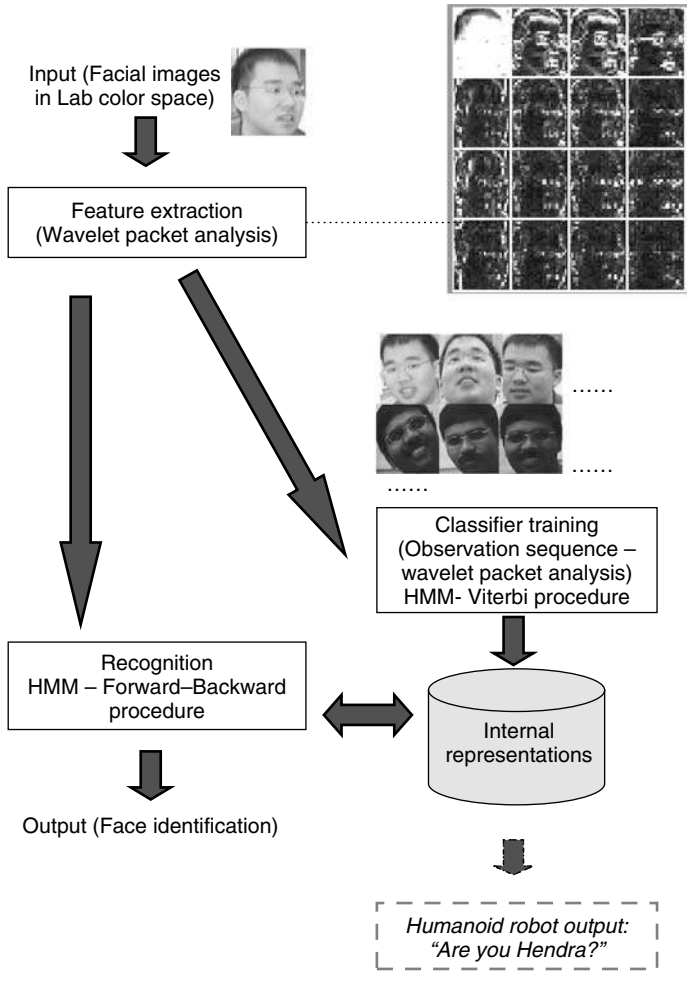



**Figure 12.24** Flowchart for developmental learning of facial image recognition.

## 5. Discussion

With many successful face recognition systems available, the objective of this chapter is not to compete with these face recognition systems, but to present the problem of developmental learning of face recognition in a new perspective; developmental vision by humanoid robots. The physical–mental development in humans should be applicable to humanoid robots to a certain extent, such that humanoid robots can exceed their human masters' physical and mental limitations. The developmental approach is motivated by human mental development from infancy to adulthood, during which each human individual develops his cognitive and behavioral capabilities through interactions with the environment [56]; it is hence believed that developmental vision systems by humanoid robots will be one of the motivations in the machine vision research arena for the next ten years. Developmental learning capabilities incorporated in humanoid robots will not only be beneficial for a dual interaction with their human masters but may also extend human capabilities in both physical and mental aspects–by, autonomous behaviors.

An RCE neural network implements color image segmentation based on color prototypes, its performance is greatly affected by the threshold of the color prototype. The accuracy of image segmentation is increased when a small threshold value is selected, but the computation time is also greatly increased. In general, we use different prototype threshold values to segment one color image. The computation time is decreased when the threshold value is increased. This is therefore a problem of equilibrium between computation time and segmentation accuracy for threshold selection [31].

The advantage of neural networks is that when applying supervised learning algorithms (e.g. an RCE neural network), they are able to adjust synaptic weights using input–output data to match the input–output characteristics of a network to desired characteristics, providing us with an efficient technique for learning color prototypes. Yet, we feel this is not sufficient for a robust developmental vision system of humanoid robots with color as its primary source of information. Further research will be needed in facilitating developmental learning of shapes by humanoid robots and, at the same time, fine-tuning the developmental learning of color algorithm when encountering the problem of backgrounds with similar color tones. Hence, we feel that color information will be more informative if color can be embedded with semantic entities. Embedding learned prototypes (e.g. colors, shapes, etc.) with semantic entities (meanings) will allow us to travel from meanings to autonomous actions by humanoid robots [57].

We selectively adopted part of the proposed wavelet packet decomposition [8] for feature extraction, and an HMM [36,37] for feature classification and recognition. The facial image database is considered insufficient for a practical application, but good enough to provide some preliminary insights into issues of establishing developmental learning of facial image recognition. Wavelet packet analysis is applied in a Matlab environment using the wavelet toolbox. First, we found that wavelet decomposition allows relatively fast computation for feature extraction compared to HMM recognition and the learning phase, but the wavelet coefficients obtained after wavelet packet decomposition are sometimes too small or nonapplicable as observation states for the HMM optimization (recognition) phase, hence resulting in falsely learned and recognized facial images. One suggestion would be to increase the size of the facial image database, which will be necessary to test the robustness of the face recognition system. Another suggestion would be to attempt normalizing or postprocessing of wavelet coefficients to yield a better result. Fully implementing wavelet packet analysis, as Garcia and Tziritas [3] have proposed for the facial feature extraction phase, could also be feasible. Hence, developmental learning of human facial expressions using wavelet analysis to obtain feature maps will be a challenge for future work. HMMs have demonstrated pleasing results when applied to the feature classification and face recognition problem; they are able to treat the face recognition problem as a learning problem which attempts to uncover 'hidden' features. They are considered to be well-suited for developmental learning of facial image recognition, analogous to young children executing 'self- patching' (imagination) of unobserved information. Hence, HMMs with proven usefulness in pattern recognition [36,44–53] are a good choice for adaptive recognition of human faces by humanoid robots.

# References


[1] Steels, L. and Kaplan, F. "Bootstrapping grounded word semantics," in Briscoe, T. (Ed.) *Linguistic Evolution Through Language Acquisition: Formal and Computational Models*, Cambridge University Press, 2002.

[2] Buxton, H. "Learning and Understanding Dynamic Scene Activity: A Review," *Image and Vision Computing* **21**, pp. 125–136, 2003.

[3] Lew, M. S. and Huijsmans, N. "Information Theory and Face Detection," *Proceedings of International Conference on Pattern Recognition*, pp. 601–605, 1996.

[4] Colmenarez, A. J. and Huang, T. S. "Face detection with information based maximum discrimination," *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 782–787, 1997.

[5] Yang, M.-H. and Ahuja, N. "Detecting Human Faces in Color Images," *Proceedings of IEEE International Conference on Image Processing*, pp. 127–139, 1998.

[6] Wu, H., Chen, Q. and Yachida, M. "Face Detection From Color Images Using a Fuzzy Pattern Matching Method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(6), pp. 557–563, 1999.

[7] Maio, D. and Maltoni, D. "Real-time Face Location on Gray-scale Static Images," *Pattern Recognition*, **33**(9), pp. 1525–1539, 2000.

[8] Garcia, C. and Tziritas, G. "Face detection using quantized skin color regions merging and wavelet packet analysis," *IEEE Transactions on Multimedia*, **MM-1** (3), pp. 264–277, 1999.

[9] Eleftheradis, A. and Jacquin, A. "Model-assisted coding of video teleconferencing sequences at low bit rates," in *Proceedings of IEEE International Symposum on Circuits and Systems*, pp. 3.177–3.180, 1994.

[10] Yang, G. and Huang, T. S. "Human face detection in a complex background," *Pattern Recognition*, **27**(1), pp. 55–63, 1994.

[11] Burl, M. C., Leung, T. K. and Perona, P. "Face localization via shape statistics," presented at *International Workshop on Automatic Face and Gesture Recognition*, June 1995.

[12] Yow, K. C. and Cipolla, C. "Feature-based human face detection," *Image and Vision Computing*, **15**, pp. 713–735, 1997.

[13] Jeng, S.-H., Yao, H. Y. M., Han, C. C., Chern, M. Y. and Liu, Y. T. "Facial feature detection using a geometrical face model: An efficient approach," *Pattern Recognition*, **31**(3), pp. 273–282, 1998.

[14] Pentland, A., Picard, R. W. and Sclaroff, S. "Photobook: Content-based manipulation of image databases," in *Proceedings of SPIE, Storage and Retrieval and Video Databases II*, 1994.

[15] Wiskott, L., Fellous, J. M., Kruger, N. and Von der Malsburg, C. "Face recognition by elastic bunch graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**, pp. 775–779, 1997.

[16] Sinha, P. "Object recognition via image invariants: A case study," *Investigative Ophthalmology and Visual Science* **35**, pp. 1.735–1.740, 1994.

[17] Lin, S.-H., Kung, S.-Y. and Lin, L.-J. "Face recognition/detection by probabilistic decision-based neural network," *IEEE Transactions on Neural Networks*, **8**, pp. 114–131, 1997.

[18] Rowley, H. A., Baluja, S. and Kanade, T. "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp.23–28, 1998.

[19] Sung, K. K. and Poggio, T. "Example-based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp. 39–51, 1998.

[20] Rowley, H. A., Baluja, S. and Kanade, T. "Rotation Invariant Neural Network-Based Face Detection," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 38–44, 1998.

[21] Féraud, R., Bernier, O. J., Viallet, J. -E. and Collobert, M. "A Fast and Accurate Face Detection Based on Neural Network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(1), pp. 42–53, 2001.

[22] Hsu, R.-L., Abdel-Mottaleb, M. and Jain, A. K. "Face Detection in Color Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(5), pp. 696–706, 2002.

[23] http://en.wikipedia.org/wiki/Neural_networks.

[24] Wesolkowski, S., Jernigan, M. E. and Dony, R. D. "Comparison of Color Image Edge Detectors in Multiple Color Space," *IEEE*, 2000.

[25] Carron, T. and Lambert, P. "Color Edge Detector Using Joint Hue, Saturation and Intensity," *1st IEEE International Conference on Image Processing*, pp. 977–981, 1994.

[26] Cumain, A. "Edge Detection in multi-spectral images," *Graphical Models and Image Processing*, **53**(1), pp. 40–51, 1991.

[27] Lee, H. C. and Cok, D. "Detecting Boundaries in a Vector Field," *IEEE Transactions on Signal Processing*, **39**(5), pp. 1181–1194, 1991.

[28] Robertson, A. L. "The CIE 1976 color difference formulae," *Color Research*, App. **2**(1), pp. 7–11, 1977.
[29] http://en.wikipedia.org/wiki/Color_space.
[30] Ohta, Y. I., Kanade, T. and Sakai, T. "Color Information for Region Segmentation," *CGIP 13*, pp. 222–241, 1980.
[31] Guo, D. and Xie, M. "Road/Traffic Color Image Segmentation Using RCE Neural Network," *SPIE International Symposium on Intelligent Systems and Advanced Manufacturing: Intelligent Robots and Computer Vision XVII Conference*, Boston, 1998.
[32] Xie, M. *Fundamentals of Robotics – Linking Perception to Action*, World Scientific Publishing Co Inc., 2003.
[33] Reilly, D. L., Cooper, L. N. and Elbaum, C. "A Neural Mode for Category Learning," *Biological Cybernetics*, **45**, pp. 35–41, 1982.
[34] Hundak, M. J. "RCE Classifiers: Theory and Practice," *Cybernetics and Systems: An International Journal*, **23**, pp. 483–515, 1992.
[35] Garcia, C., Zikos, G. and Tziritas, G. "Face detection in color images using wavelet packet analysis," *Multimedia Computing and Systems*, IEEE, **1**, pp. 703–708, 1999.
[36] Rabiner, L. "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of IEEE*, **77**(2), pp. 257–286, 1989.
[37] Eickeler, S., Müller, S. and Rigoll, G. "Recognition of JPEG compressed face images based on statistical methods," *Image and Vision Computing*, **18**, pp. 279–287, 2000.
[38] http://www.mathworks.com/access/helpdesk/help/toolbox/wavelet/wavelet.shtml.
[39] Mallat, J. S. "A theory of multi-resolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**, pp. 674–693, 1989.
[40] Daubechies, I. "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory*, **36**, pp. 961–1005, 1990.
[41] Smith, M. and Barnwell, T. "Exact reconstruction techniques for tree structured sub-band coders," *IEEE Transactions on Acoustics, Speech and Signal Processing*, **ASSP-34**, pp. 434–441, 1986.
[42] Westerink, P. H., Biemond, J. and Boekee, D. E. "Sub-band coding of color images," in Woods, J. W. (Ed.), *Sub-band Image Coding*, Kluwer, Boston, MA, pp. 193–228, 1991.
[43] Baum, L. E. and Petrie, T. "Statistical inference for probabilistic functions of finite state Markov chains," *Annals of Mathematical Statistics*, **37**, pp. 1554–1563, 1966.
[44] Starner, T., Weaver, J. and Pentland, A. "Real-time American sign language recognition using desk and wearable computer-based video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, pp. 1371–1375, 1998.
[45] Kuo, S. S. and Agazzi, O. E. "Keyword spotting in poorly printed documents using pseudo 2-d hidden Markov model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**(8), pp. 842–848, 1994.
[46] Kundu, A., He, Y. and Bahl, P. "Recognition of handwritten words: First and second order hidden Markov model based approach," *Pattern Recognition*, **22**(3), pp. 283–297, 1989.
[47] Samaria, F. S. *Face Recognition Using Hidden Markov Model*, PhD thesis, Cambridge University, 1994.
[48] Nefian, A. V. *A hidden Markov model-based approach for face detection and recognition*, PhD thesis, Georgia Institute of Technology, 1999.
[49] Hisham, O. H. A. *A novel reduced-complexity approach to hidden Markov modeling of two-dimensional processes with application to face recognition*, PhD thesis, University of Ottawa, Canada, 2002.
[50] He, Y. and Kundu, A. "2-D shape classification using hidden Markov model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**(11), pp. 1172–1184, 1991.
[51] Levin, E. and Pieraccini, R. "Dynamic planar warping for optical character recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp 149– 152, 1992.
[52] Nefian, A. V. and Hayes, M. H. "An embedded hmm-based approach for face detection and recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 3553–3556, 1999.
[53] Rigoll, G., Müller, S. and Winterstein, B. "Robust person tracking with non-stationary background using a combined pseudo-2d-hmm and Kalman-filter approach," in *Proceedings of IEEE International Conference on Image Processing*, pp. 242–246, 1999.
[54] Sanderson, C. *Face Processing and Frontal Face Verification*, IDIAP–RR 03-20, April, 2003.
[55] Eickeler, S., Jabs, M. and Rigoll, G. "Comparison of Confidence Measures for Face Recognition," *IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 257–262, 2000.

[56] Weng, J., Hwang, W. S., Zhang, Y. and Evans, C. H. "Developmental Robots: Theory, Method and Experimental Results," *Proceedings of 2nd International Symposium on Humanoid Robots*, Tokyo, Japan, pp. 57–64, 1999.

[57] Xie, M., Kandhasamy, J. S. and Chia, H. F. "Robot Intelligence: Towards machines that understand meanings," *International Symposium of Santa Caterina on Challenges in the Internet and Interdisciplinary Research*, Amalfi Coast, Italy, January 2004.

# 13

# Empirical Study on Appearance-based Binary Age Classification

**Mohammed Yeasin**
Department of Computer Science,
State University of New York, Institute of Technology Utica,
NY-13504, USA

**Rahul Khare**
**Rajeev Sharma**
Computer Science and Engineering Department
Pennsylvania State University, University Park, PA-16802, USA

*This chapter presents a systematic approach to designing a binary classifier using Support Vector Machines (SVMs). To exemplify the efficacy of the proposed approach, empirical studies were conducted in designing a classifier to classify people into different age groups using only appearance information from human facial images. Experiments were conducted to understand the effects of various issues that can potentially influence the performance of such a classifier. Linear data projection techniques such as Principal Component Analysis (PCA), Robust PCA (RPCA) and Non-Negative Matrix Factorization (NMF) were tested to find the best representation of the image data for designing the classifier. SVMs were used to learn the underlying model using the features extracted from the examples. Empirical studies were conducted to understand the influence of various factors such as preprocessing, image resolution, pose variation and gender on the classification of age group. The performances of the classifiers were also characterized in the presence of local feature occlusion and brightness gradients across the images. A number of experiments were conducted on a large data set to show the efficacy of the proposed approach.*

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

# 1. Introduction

Intelligent systems for monitoring people and collecting valuable demographics in a social environment will play an increasingly important role in enhancing the user's interaction experience with a computer, and can significantly improve the intelligibility of Human–Computer Interaction (HCI) systems. For example, a robust age classification system is expected to provide a basis for the next generation of *parental control tools*. Robust age classification, along with gender and ethnicity information, can have a profound impact in advertising (i.e. narrowcasting and gathering demographic data for marketing), law enforcement, education, security, electronic commerce, gathering consumer statistics, etc. This would directly impact the consumer electronics and personal computer hardware markets since they are the key distribution channels for delivering content in today's society. The algorithm discussed in this chapter may be integrated into the VLSI circuit of media devices, along with features to customize the desired type of control.

Current methods for gathering vital consumer statistics for marketing involve tedious surveys. If it were possible to gather data such as age information about the customers within the marketing establishments, such information could prove to be vital for deciding which sections of society need to be targeted. This could potentially lead to enormous savings in time as well as money. Current methods of advertising follow broadcasting techniques wherein the advertisement is for all people and not targeted specifically towards the person watching the advertisement. In many practical scenarios, such a method may not be the best way, and *automated narrowcasting* could be used. Such targeted advertisements have a deeper impact as they are targeted specifically to the tastes of the person watching the advertisement. Tastes, needs and desires of people change with age and if it were possible to determine the age category of the person, then narrowcasting could be used effectively in such scenarios. Several different visual cues could be used for determining the age category of a person in an automated manner. Height, skin texture, facial features and hair color are some of the visual features that change with age.

Cranio-facial research [1] suggests that as people grow older there is an outgrowing and dropping of the chin and the jaw. This change in the skull with an increase in age changes the T-ratios (transverse ratios of the distance between the eyes to the distance between lines connecting the eyes from the nose/chin/top of head). This theory was validated in a study in [2], where it was found that children up to the age of 18 have different ratios compared to people beyond the age of 18. After the age of 18 the skull stops growing and so there is no change in the ratios. Kwon *et al.* [2] used these results to differentiate between children and adults after using facial feature detectors on high-resolution images. The presence and absence of wrinkles was also used for differentiating between seniors and young adults (refer to [3] for details). However, these methods are inadequate for use in applications that require real-time performance with low-resolution images. The use of snakes for finding wrinkles in seniors is difficult to compute from low-resolution facial images and takes time to stabilize the snakes.

Recent advances in image analysis and pattern recognition open up the possibility of automatic age classification from only facial images. Classifying age with a very high degree of accuracy is a relatively hard problem, even for human experts [1]. Automatic classification of age from only facial images presents a number of difficult challenges. In general, five main steps can be distinguished in tackling the problem in an holistic manner.

1. An automated robust face detection system for segmenting the facial region.
2. Preprocessing of image data to eliminate the variabilities that may occur in image data capture.
3. Suitable data representation techniques that can keep useful information and at the same time reduce the dimensionality of the feature vector.
4. Design of a robust classifier for the classification of age.
5. Characterization of performance of the classifier.

This chapter provides an holistic approach by taking into account the five steps outlined to classify humans according to age groups of over 50 and under 40 using only visual data. The rest of the

chapter is organized as follows. Section 2 describes the state of the art in understanding factors that may be used to estimate the age and automatic age classification. Following this, a brief description of the classifier design is presented in Section 3. Section 4 presents the results of empirical analysis conducted to understand the effects of various factors that may affect the performance of a classifier. Finally, Section 5 ends the chapter with a few concluding remarks and ideas for future work.

## 2. Related Works

Automated classification of age from appearance information is largely underexplored. In [2], Kwon *et al.* made an attempt to classify images of people into three age categories: children, adults and senior adults. This work was based on the cardioidal strain transformation to model the growth of a person's head from infancy to adulthood, obtained from cranio-facial research [1]. The revised cardioidal strain transformation describing head growth can be visualized as a series of ever-growing circles all attached at a common tangent 'base' point, in this case the top of the head. According to this transformation, with an increase in age, the growth of lower parts of the face is more pronounced than that of the upper part. Thus, for example, within the top and bottom margins of the head, the eyes occupy a higher position in an adult than in an infant due to an outgrowing and dropping of the chin and jaw. This result was used in distinguishing between adults and babies. In order to do this the authors used facial feature detectors to detect the eyes, nose and the mouth. These feature positions were then used to find out certain ratios for each face and if the ratio was greater than a particular value, the face was classified as belonging to an adult, else it was classified as belonging to a baby. In order to distinguish between young adults and seniors the authors used snakes to determine the presence of wrinkles on the forehead of the person. The main problem with this method is the need to detect facial features, which is extremely error-prone. The costs and processing time of this approach limit its practicality for use in real applications.

In [1,4] O'Toole *et al.* used three-dimensional information for building a parametric 3D face model. They used caricature algorithms to exaggerate or de-emphasize the 3D facial features. In the resulting images the perceived age is changed based on whether the feature is exaggerated or de-emphasized. This result suggested that in older faces, the 3D facial features are emphasized. These results were verified by a number of observers. In a related work, Lanitis *et al.* [5] proposed a method for simulating aging in facial images, thus allowing them to 'age-normalize' faces before using them for 'face recognition'. For this purpose the authors used statistical model parameters that contain shape and intensity information. Some of these parameters were then used for simulating aging effects. Empirical results reported in this paper suggest that the face recognition accuracy improved when 'age normalization' was carried out. These results indicate that the appearance of the face contains sufficient information regarding the age of the person that in turn could be used for classification purposes.

In [2] it was shown that wrinkles could be used for differentiating between senior citizens and adults below a certain age. From this result it could be argued that the presence and absence of wrinkles, as indicated by the skin texture, could be detected even at low resolutions. Also, in [6] it has been stated that after age 45, the skin begins to thin, partially because of hormonal changes, resulting in a loss of volume and smoothness. In [3] it was found that it is easy to notice that there is a very big change in the facial skin texture of the person from age 40 to age 50. Keeping these points in mind, the two age categories used for classification were the age groups above 50 and below 40, keeping the ten years in between as a buffer zone where the performance of the classifier is unpredictable.

## 3. Description of the Proposed Age Classification System

This section provides a detailed description of the proposed system. Figure 13.1 shows the block diagram consisting of the main blocks used in the age category classification system. The proposed system crops the face in the current scene and decides the age category of the person. The output

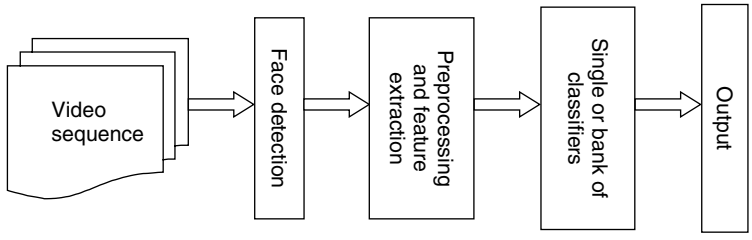



**Figure 13.1** Block diagram of the age classification system.

of the face detector passes through the preprocessing algorithms, such as histogram equalization and brightness gradient removal, in order to present images of uniform brightness to the classifier. Before the image is fed to the classifier, the image is passed through a feature extractor algorithm. Principal component analysis, robust principal component analysis and non-negative matrix factorization were used for experimentation. This representation of the image is finally fed to the classifier that decides the age category of the person.

To design the classifier, publicly available SVMLight [7] software was used. Bootstrapping techniques were used to learn a useful model based on available examples (see Figure 13.2 for details). The design of the classifier was undertaken in several stages. First, the training examples were used to train the SVMs to learn the underlying model which acts as a naïve classifier. Using the naïve classifier, one meticulously performs a series of experiments to pick a representative set of examples to bootstrap the classifier. A series of such new classifiers were tuned and tested on a test data set. The best classifier was chosen for the final testing for a given resolution and all other possible combinations that may influence the output of the classifier. A four-fold cross validation strategy was employed to report the classification results. For each of the training and testing sets, a radial basis function was used as the kernel function that had a fixed value of gamma, and the cost factor was varied with values of 0.1, 0.5, 1, 5, 10, 50 and 100. Thus, for each of the training and testing sets, seven different accuracies were obtained, depending on the parameter settings. The accuracies were averaged out for all the four sets and the best average accuracies were used as the final results. The output of the classifier was then used in the decision fusion process in the case of the parallel paradigm of classification, or fed to the next level of classifier in the case of the serial paradigm of classification. A detailed description of the proposed approach is presented in the following subsections.

## *3.1 Database*

A large database was collected using the resources from Advanced Interfaces Inc. The facial portions of the images were segmented automatically using the face detector developed in [8] and originally

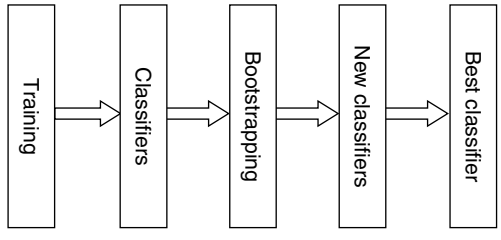



**Figure 13.2** Bootstrapping process to find the best classifier for testing.

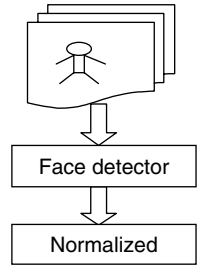



**Figure 13.3** Data collection and face normalization.

proposed in [9]. The set-up used for obtaining the facial images was as shown in Figure 13.3. The pose variation in the data set was determined by the face detector, which allowed a maximum in-plane and/or out-of-plane rotation of about 30 degrees. The data set consisted of about 4100 grayscale images with a minimum resolution of $29 \times 29$ distributed across the age categories of ages 20 to 40 and 50 to 80. The database was divided up into four groups of males and females above 50 years old and below 40 years old. Table 13.1 summarizes the distribution of the database.

In the process of data collection, facial images of people of different age groups were collected. All these images were appropriately labeled with the age of the person in the image. These labels were used as ground truths to be used during the training of the classifiers. This data set was divided into three parts – the training set, the bootstrapping set and the testing set, all of them mutually disjoint.

## *3.2 Segmentation of the Facial Region*

A biological face detection system developed by Yeasin *et al.* [8] was used to segment the face from the rest of the image. The following criteria were emphasized while developing the face detector:

1. The algorithm must be robust enough to cope with the intrinsic variabilities in images.
2. It must perform well in an unstructured environment.
3. It should be amenable to real-time implementation and give few or no false alarms.

An example-based learning framework was used to learn the face model from a meticulously created representative set of face and nonface examples. The key to the success of the method was the systematic way of generating positive and negative training examples using various techniques, namely: preprocessing, lighting correction, normalization, the creation of virtual positive examples from a reasonable number of facial images and the bootstrap technique to collect the negative examples for training. A successful implementation of face detection was made using a retinally connected neural network architecture reported in [9], this was refined later to make it suitable for real-time applications. The average performance of the system is above 95 % on facial images having 30–35 degree deviation from the frontal facial image.

**Table 13.1** Database of images.

| Gender | Age category | Number of images |
|---|---|---|
| Female | Over 50 | 631 |
| Male | Over 50 | 1042 |
| Female | Under 40 | 1283 |
| Male | Under 40 | 1214 |

## 3.3 Preprocessing

The localized face from the previous stage is preprocessed to normalize the facial patterns. Several techniques, such as illumination gradient correction, were performed to compensate or to reduce the effect of lighting variations within the window where the face was detected. Also, histogram equalization was performed to reduce the nonuniformity in the pixel distributions that may occur due to various imaging situations. Additionally, normalization of the 'training set' was performed to align all facial features (based on manual labeling) of the face with respect to a canonical template, so that they formed a good cluster in the high-dimensional feature space.

## 3.4 Feature Extraction

The feature extraction stage is designed to obtain a meaningful representation of observations and also to reduce the dimension of the feature vector. It is assumed that a classifier that uses a smaller dimensional feature vector will run faster and use less memory, which is very desirable for any real-time system. Besides increasing accuracy by removing very specific information about the images, the feature extraction method also improves the computational speed of the classifier – an important criterion for a real-time classifier system. In the proposed method, the following techniques were implemented to construct a feature vector from the observations. Experiments were conducted with the aim of achieving two goals, namely: to select the components of the PCA and NMF; and to make comparisons between PCA and NMF and use the suitable one for classification of age. A weighted NMF [10] scheme was used in the experimentation to alleviate the limitation of standard NMF in optimizing the local representation of the image. Several experiments were conducted with different numbers of bases to compare the classification results.

## 3.5 Classifying People into Age Groups

For the training of the classifier, about 50 % of the data collected from all the age categories was used. A method of cross validation was used to get the best possible classifier. The different parameters that could be changed were the kernels, kernel parameters and the cost factor. Once the best classifier was found from the cross validation method, the misclassified examples could be used in the bootstrapping process and to further refine the classifier, thus finding the optimum classifier.

In order to improve the performance of the classifier, either the parallel or the serial or a combination of the two paradigms could be used. The parallel paradigm is based on the fact that examples misclassified by one classifier could be classified correctly by another, thus giving a better overall accuracy if both classifiers are used. The classifiers can vary either in the type of parameters used or the type of feature extraction used for them. Another way to improve the accuracy is to use the fact that there are big differences in the facial features of different genders, as well as different ethnicities. For example, the face of an adult female could be misclassified as a person from a lower age category. Hence, different sets of images could be used for training the classifier for female age categories and for male age categories. The same logic could be extended for people of different ethnicities, leading to the need to use an ethnicity classifier before the age category classifier. Using the parallel and the serial paradigms simultaneously would give the best possible performance.

Age category classification could be a binary age category classifier using the serial paradigm for classification. In this example, the image fed by the camera is used by the face detector software to detect the face in it. This face is then resized to 20 by 20 and histogram equalization and brightness gradient removal is carried out on the image. Following the image processing, the image is passed through a feature detector having a set of 100 basis vectors, thus giving a feature vector with 100 values. This is then fed to a gender classifier and, depending on the gender output, it is either fed to a male age classifier or a female age classifier. The final output of the age classifier gives the age category of the person as belonging either to the adult age category or the minor age category.

## 4. Empirical Analysis

To understand the effects of various factors that may influence the performance of the classifier, a number of experiments were conducted in order to characterize the performance of the classifier. The tests conducted were:

1. Performance of dimensionality reduction methods.
2. The effect of preprocessing and image resolution.
3. The effect of pose variation.
4. Characterization of brightness gradients.
5. Characterization of occlusion.
6. The impact of gender on age classification.
7. Classifier accuracies across the age groups.

### *4.1 Performance of Data Projection Techniques*

The problem known as the 'curse of dimensionality' has received a great deal of attention from researchers. Many techniques have been proposed to project data. These techniques are based on the mapping of high-dimensional data to a lower dimensional space, such that the inherent structure of the data is approximately preserved. In this work, two linear projection methods for dimensionality reduction were used to find a suitable representation of the image data. Three different representations, namely: Principal Component Analysis (PCA), Robust Principal Component Analysis (RPCA) [11] and Non-Negative Matrix Factorization (NMF) [7] were used for the experimentation. The two forms of PCA are supposed to capture the global features of the facial image, while NMF is supposed to capture the local image features of the facial image. Classical PCA is very sensitive to noise, so to nullify the effect of noise, RPCA methods were tested.

PCA is designed to capture the variance in a data set in terms of principal components. In effect, one is trying to reduce the dimensionality of the data to summarize the most important (i.e. defining) parts, whilst simultaneously filtering out noise present in the image data. However, the decision regarding which component of PCA captures the meaningful information for a particular problem is not well understood in general. Hence, it poses a problem in experimentation, because whether the age information is present in the eigenvectors corresponding to the highest eigenvalues, or those with low eigenvalues is not known. To overcome this problem, the experiments involved varying the number of principal components from 10 to 100 in steps of ten. A similar strategy was also used for the RPCA-based representation. The second method for dimensionality reduction that was used is Non-Negative Matrix Factorization (NMF) [7]. The advantage of this method is that in cases where the age information is present in certain features of the face and not the entire face, NMF would be able to capture that information. However, it has the same problem as PCA, i.e. basis vector determination, and was also handled in the same way as PCA. In each case, the basis vectors were obtained using the set of training examples.

From the experiments it was found that all representational schemes, namely: PCA, RPCA and NMF, yielded very low accuracies of around 50 % for the age classification problem. On the other hand, classification done using raw image values for the same set of data gave accuracies around 70 %. This indicated that when the images were projected along the vector subspace for PCA, RPCA and NMF, data crucial for age classification was lost. While this may sound counterintuitive, it is believed that the lower accuracy has some important significance to choosing appropriate data projection techniques used in the experimentation. According to the central limit theorem, low-dimensional linear projections tend to be normally distributed as the original dimensionality increases. This would mean that little information could be extracted by linear projection when the original dimensionality of data is very high. To overcome this problem, nonlinear data projection methods (for example, self-organizing maps [12], distance-based approaches such as multidimensional scaling [13] and local linear embedding [14]) should be investigated.

## 4.2 The Effect of Preprocessing and Image Resolution

Preprocessing algorithms were designed to get rid of any variations in the lighting conditions in the images. The two preprocessing methods tested were histogram equalization and histogram equalization followed by brightness gradient removal. Brightness gradient removal cannot be used as a stand-alone preprocessing step and so was used in conjunction with histogram equalization to check if it has any significant impact on the classifier accuracies. Hence, the data set available was preprocessed to get three sets – unprocessed images, histogram equalized images and images processed through histogram equalization with brightness gradient removal.

In order to find out the optimum image resolution to be used for classification, the available image set was downsampled to obtain $25 \times 25$ and $20 \times 20$ size images, along with the existing image resolution of $29 \times 29$. The reason behind trying out lower image resolutions was that by downsampling, certain high-frequency peculiarities specific to people, such as moles, etc., would be lost, leading to better classification accuracies. Figure 13.4 shows the plot of classification accuracy at various image resolutions under different image preprocessing conditions. The result in Figure 13.4 represents the model obtained using male examples only. It may be noted that similar results were obtained for models trained using females, as well as for models obtained by using combined male and female examples. From the plot it can be seen that there is an improvement in the classification accuracy of about 4–5 % for an increase in the facial image resolution from $20 \times 20$ to $29 \times 29$. This improvement can be seen for all image preprocessing conditions.

## 4.3 The Effect of Pose Variation

In order to be effective for real-life applications, the age classifier should provide stable classification results in the presence of significant pose variations of the faces. In order to test the performance of the classifier in the presence of pose variations, the following test was conducted. The faces of 12 distinct people were saved under varying poses from frontal facial image to profile facial image. The best classifiers obtained for each resolution were used on these sets of faces to check for the consistency of classification accuracy. Thus, if, for a given track having ten faces, eight faces were classified correctly, the accuracy for that track was tabulated as 80 %. In this way, the results were tabulated for all three classifiers.

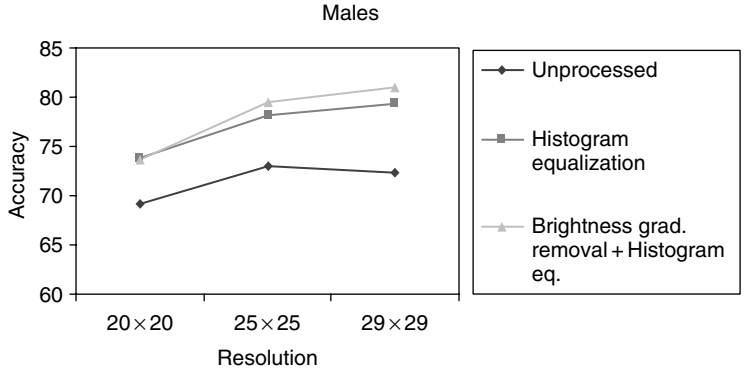



**Figure 13.4** The effect of preprocessing on age classification. Age classification accuracy (male only database) at various image resolutions under different preprocessing methods.

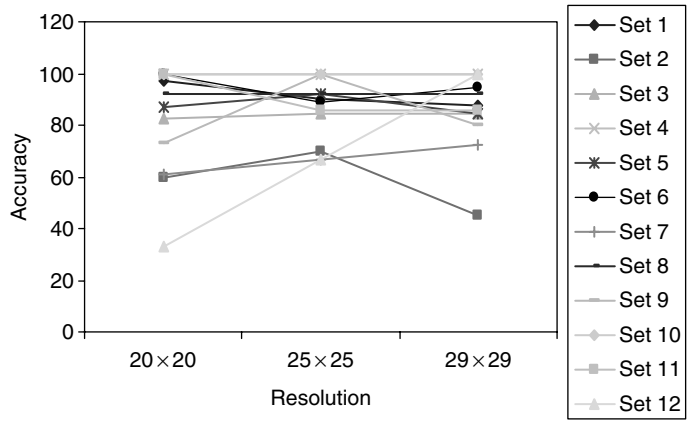



**Figure 13.5** The effect of pose. Age classification accuracy at various image resolutions for 12 sets of images with pose variations.

The results for the performance of the classifier under pose variation are summarized in Figure 13.5. From this experiment it is easy to see that for all three classifiers at least 11 out of 12 sets of faces gave an accuracy of greater than or equal to 60 %. This result is significant as it points to a possibility of using the sampled classifier output for all the faces obtained in a given track to classify the person in the track. Figure 13.5 indicates that for all the resolutions, at least 11 of the 12 sets of images with pose variation gave an accuracy of greater than 50 %.

## *4.4 The Effect of Lighting Conditions*

In real-life scenarios, there are changes in the lighting conditions that can produce diverse intensity gradients across the facial images. The intensity gradients could be across the entire face or only across half the face, depending on the position of the light source. To test the performance of the classifier under such circumstances, gradual intensity gradients were introduced in the available set of images and tested with the available classifiers trained from the images without the brightness gradients. The purpose of the testing was to check the effectiveness of the preprocessing steps in taking care of such gradients. Figure 13.6 indicates that, as expected, the brightness gradient removal method for preprocessing worked best in the presence of vertical and horizontal gradients in the image. However, in the presence of diagonal gradients in the image, histogram equalization gave results equally as good as the results obtained using histogram equalization with brightness gradient removal as the preprocessing step.

## *4.5 The Effect of Occlusion*

Often, in the presence of crowds, there is occlusion of the face. To test the efficacy of the classifier under such circumstances, the available set of images was occluded by darkening a certain region and then tested with the available classifiers. Three different types of occlusion were made to simulate the occlusion of the left eye, the right eye and the mouth. Figure 13.7 presents the results obtained from this experimentation. From Figure 13.7 it is evident that histogram equalization as the preprocessing step is best able to handle occlusions. This plot also indicates the fact that occlusion of the mouth region causes a drastic reduction in the classification accuracy.

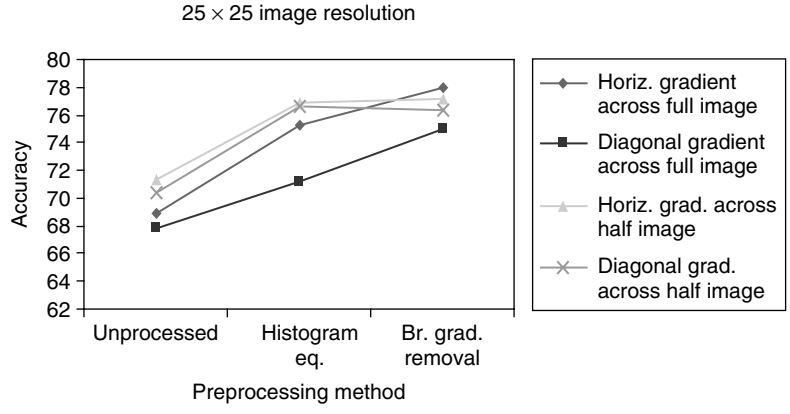



**Figure 13.6** The effect of light. Age classification accuracy for different simulations of brightness gradients in the images for $25 \times 25$ image resolution.

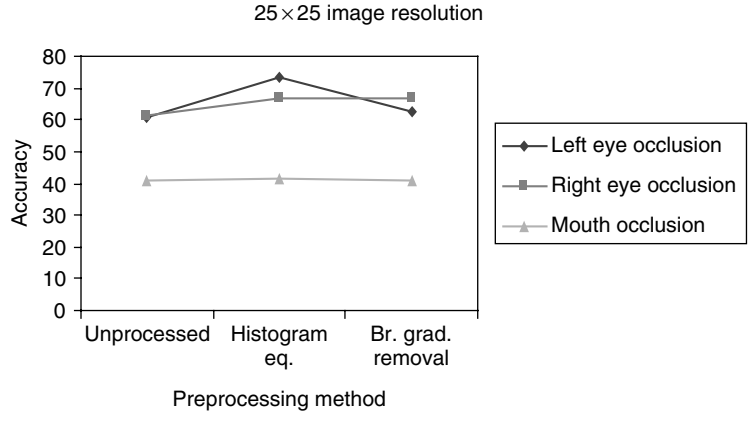



**Figure 13.7** The effect of occlusion. Age classification accuracy for different simulations of occlusion in the images for $25 \times 25$ image resolution.

## *4.6 The Impact of Gender on Age Classification*

Studies have been conducted which indicate that men and women age at different rates. In order to check if these variable rates of aging affect the classifier in any way, two tests were conducted. In the first test, classifiers were trained and tested separately for each gender. In the second case, examples of males and females were used together in the training, and in the testing, the accuracy rate for males and females was found separately. Figure 13.8 shows the effect of gender classification on age classification. From the plot it is evident that using a combined set of males and females for training gives better results for female classification but a poorer result for male classification when compared to using only females or only males respectively in the training set.

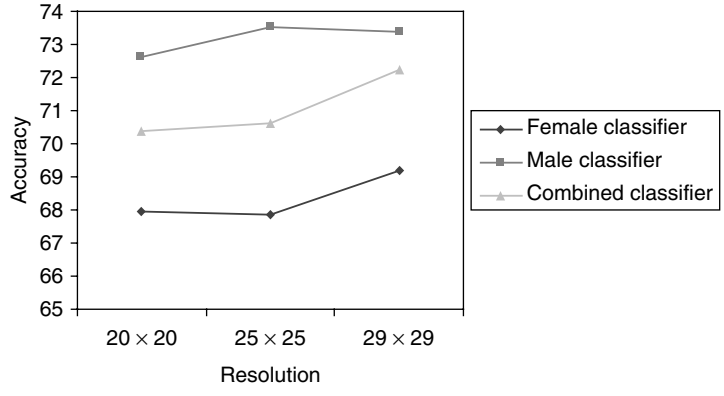



**Figure 13.8** The effect of gender. Age classification at various image resolutions for gender-specific age classifier models.

## *4.7 Classifier Accuracies Across the Age Groups*

In order to find out the position of the classifier hyperplane and the way the data was clustered around it, a test was carried out to check the accuracies of the classifier for the age groups from 20s to 80s. Thus, for the three classifiers, one for each resolution, facial images of people belonging to the age groups from 20s to 80s were used as the test cases, and the accuracy for each group was found and plotted. Figure 13.9 shows the plot of classification for different age groups and indicates that the classifier hyperplane lies somewhere in the 35 to 40 age group. Figure 13.10 shows representative sample images used for testing and also samples for failure cases.

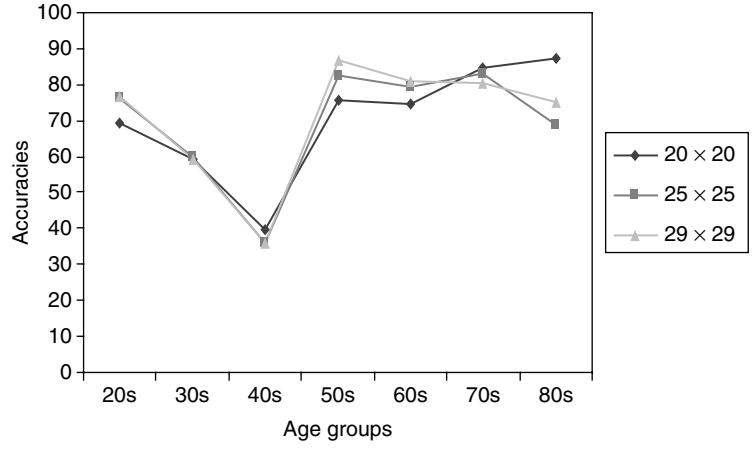



**Figure 13.9** Age classification accuracy across various age groups at different image resolutions.

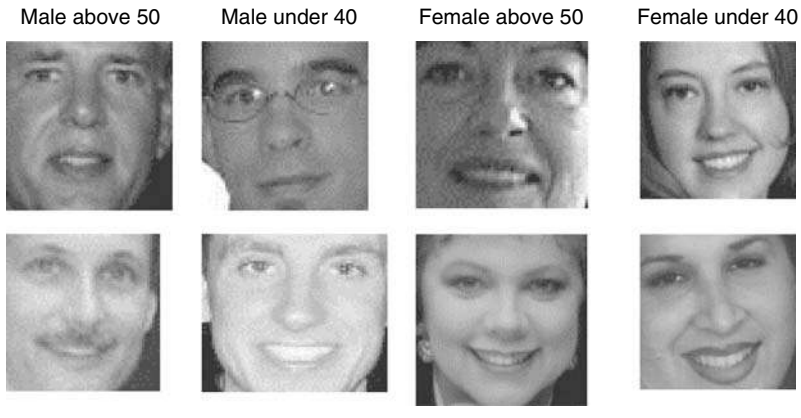



**Figure 13.10** Sample images used for testing. Top row: images that were classified correctly; bottom row: sample images of the failure cases.

## 5. Conclusions

This chapter conducted a systematic study on designing an appearance-based age classifier using SVMs. The performance of the classifier was characterized under varying conditions of data resolution, data representation, preprocessing methods, pose variations, brightness conditions and the presence of occlusion. Empirical results of the experiments suggests that:

1. Classifiers employing preprocessing and normalization show a definite improvement in performance as compared to classifiers developed using no preprocessing methods.
2. The intuitive idea of higher resolution images giving better classifier accuracy is mostly true but not always.
3. The use of linear data projection techniques such as PCA or NMF did not improve the performance compared to using just the plain image intensity values for classification.
4. The brightness gradient removal method along with histogram equalization gives a performance improvement only in the presence of brightness gradients, otherwise it does not provide any significant improvement.
5. Occlusion of the mouth area causes the maximum degradation of performance of the classifiers, indicating that pixels around the mouth are used in a significant way for classification.

While the proposed system performs reliably in many real-world situations, it may be useful to combine it with other modalities to achieve more robustness. Further studies may focus on finding the causes for the failure of dimensionality reduction techniques in age classification. This work only investigated the linear projection methods of PCA and NMF. According to the central limit theorem, low-dimensional linear projections tend to be normally distributed as the original dimensionality increases. This would mean that little information could be extracted by linear projection when the original dimensionality of data is very high. To overcome this problem, nonlinear data projection methods (for example, self-organizing maps [12], distance-based approaches such as multidimensional scaling [13] and local linear embedding [14]) should be investigated. This work empirically studied two age categories which are visually distinct in their skin textures. However, more work needs to be done to check the performance of this method using lower age categories, such as people under 18 years old, as a class. Better accuracy rates could be obtained by utilizing some form of sampling of the classifier outputs, to be undertaken as future work that will make the system viable for commercial application.

# Appendix A. Data Projection Techniques

There are many methods for estimating intrinsic dimensionality of data without actually projecting the data [15]. In addition to intrinsic dimensionality estimation, data projection methods are further required to generate lower dimensional configurations. There are generally two categories of method for projecting data to a low space: linear projection and nonlinear projection. Linear projection is characterized by a projection matrix. Examples are Principal Component Analysis (PCA) and Non-Negative Matrix Factorization (NMF). This appendix briefly discusses only the linear data projection techniques.

## *A.1 Principal Component Analysis (PCA)*

Due to the high dimensionality of data, similarity and distance metrics are computationally expensive and some compaction of the original data is needed. Principal component analysis is an optimal linear dimensionality reduction scheme with respect to the Mean Squared Error (MSE) of the reconstruction. For a set of $N$ training vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, the mean ($\mu = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$) and covariance matrix ($\sum = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \mu)(\boldsymbol{x}_i - \mu)^{\mathrm{T}}$) can be calculated. Defining a projection matrix $\mathbf{E}$ composed of the $K$ eigenvectors of $\Sigma$ with highest eigenvalues, the $K$-dimensional representation of an original, $n$-dimensional vector $\boldsymbol{x}$, is given by the projection $\boldsymbol{y} = \mathbf{E}^{\mathrm{T}}(\boldsymbol{x} - \mu)$.

## *A.2 Non-Negative Matrix Factorization (NMF)*

NMF is a method of obtaining a representation of data using non-negativity constraints. These constraints lead to a part-based representation because they allow only additive, not subtractive, combinations of the original data [10]. Given an initial database expressed by an $n \times m$ matrix $\mathbf{V}$, where each column is an $n$-dimensional non-negative vector of the original database ($m$ vectors), it is possible to find two new matrices ($\mathbf{W}$ and $\mathbf{H}$) in order to approximate the original matrix $\mathbf{V}_{i\mu} \approx (\mathbf{WH})_{i\mu} = \sum_{a=1}^{r} \mathbf{W}_{ia}\mathbf{H}_{a\mu}$. The dimensions of the factorized matrices $\mathbf{W}$ and $\mathbf{H}$ are $n \times r$ and $r \times m$, respectively. Usually, $r$ is chosen so that $(n+m)r < nm$.

Each column of matrix $\mathbf{W}$ contains a basis vector, while each column of $\mathbf{H}$ contains the weights needed to approximate the corresponding column in $\mathbf{V}$ using the basis from $\mathbf{W}$. In the PCA context, each column of matrix $\mathbf{W}$ represents an eigenvector and the factorized matrix of $\mathbf{H}$ represents the eigen projections. In contrast to PCA, NMF does not allow negative entries in the factorized matrices $\mathbf{W}$ and $\mathbf{H}$ permitting the combination of multiple basis images to represent an object. In order to estimate the factorization matrices, an objective function has to be defined. A possible objective function is given by $F = \sum\sum[\mathbf{V}_{i\mu}\log(\mathbf{WH})_{i\mu} - (\mathbf{WH})_{i\mu}]$. This objective function can be related to the likelihood of generating the images in $\mathbf{V}$ from the basis $\mathbf{W}$ and encoding $\mathbf{H}$. An iterative approach to reaching a local maximum of this objective function is given by the following rules:

$$\mathbf{W}_{ia} \leftarrow \mathbf{W}_{ia}\sum_{\mu}\frac{\mathbf{V}_{i\mu}}{(\mathbf{WH})_{i\mu}}\mathbf{H}_{a\mu}, \mathbf{W}_{ia} \leftarrow \frac{\mathbf{W}_{ia}}{\sum_j \mathbf{W}_{ja}}, \mathbf{H}_{a\mu} \leftarrow \mathbf{H}_{a\mu}\sum_i \mathbf{W}_{ia}\frac{\mathbf{V}_{i\mu}}{(\mathbf{WH})_{i\mu}}$$

Initialization is performed using positive random initial conditions for matrices $\mathbf{W}$ and $\mathbf{H}$. The convergence of the process is also ensured.

# Appendix B: Fundamentals of Support Vector Machines

The foundations of Support Vector Machines (SVMs) have been developed by Vapnik [16–18], and are gaining popularity due to their many attractive features and promising empirical performance. The

formulation embodies the Structural Risk Minimization (SRM) principle, as opposed to the Empirical Risk Minimization (ERM) approach commonly employed in nonparametric methods. SRM minimizes an upper bound on the generalization error, as opposed to ERM that minimizes the error on the training data. SVMs can be applied to both classification and regression problems. They condense all the information contained in the training set relevant to classification in the support vectors. This reduces the size of the training set identifying the most important points, and makes it possible to efficiently perform classification. Finally, SVMs are quite naturally designed to perform classification in high-dimensional spaces, even in the presence of a relatively small number of data points.

In a two-class pattern classification problem, the task of learning from examples can be formulated in the following way. Given a set of decision functions: $\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}$, $f_\lambda : \Re^N \to \{-1, 1\}$, where $\Lambda$ is a set of abstract parameters, and a set of examples, $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, $\mathbf{x}_i \in \Re^N$, $y_i \in \{-1, 1\}$, drawn from an unknown probability distribution $P(\mathbf{x}, y)$, i.e. the data are assumed independently drawn and identically distributed. We want to find the function $f_\lambda$ that provides the smallest possible value for the expected risk:

$$R(\lambda) = \int |f_\lambda(\mathbf{x}) - y| \, P(\mathbf{x}, y) \mathrm{d}\mathbf{x} \mathrm{d}y \tag{13.1}$$

The functions $f_\lambda$ are usually called hypotheses, and the set $\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}$ is called the hypothesis space and is denoted by H. The expected risk is a measure of how good a hypothesis is at predicting the correct label $y$ for a point $\mathbf{x}$. The probability distribution $P(\boldsymbol{x}, y)$ is unknown, and impossible to compute. However, since we have access to samples of $P(\mathbf{x}, y)$, we can compute a stochastic approximation of $R(\lambda)$, the so-called empirical risk, as follows:

$$R_{\mathrm{emp}}(\lambda) = \frac{1}{m} \sum_{i=1}^{m} |f_\lambda(\mathbf{x}_i) - y_i| \tag{13.2}$$

where $m$ is the number of examples. There is no probability distribution here and $R_{\mathrm{emp}}$ is a fixed number for a particular choice of $\lambda$ and for a particular training set $\{\mathbf{x}_i, y_i\}$. The approach now is to converge the empirical risk rather than expected risk. As shown by Vapnik and Chervonenkis [17], consistency takes place if and only if convergence in probability of $R_{\mathrm{emp}}$ to $R$ is replaced by uniform convergence in probability. The theory of uniform convergence in probability developed by Vapnik and Chervonenkis [17] also provides bounds on the deviation of the empirical risk from the expected risk. For more details please refer to [19].

## Acknowledgments

This work was partially supported by NSF CAREER grant IIS-97-33644 and NSF Grant IIS-0081935. The authors would like to thank Advanced Interfaces Inc. for providing the funding, database and software needed for carrying out this research work.

## References

[1] O'Toole, A. J., Vetter, T., Volz, H. and Salter, E. M. *As we get older, do we get more distinct*, Technical Report no. 49, Max Planck Institut fur biologische kybernetic, March 1997.

[2] Kwon, Y. H. and da Vitoria Lobo, N. "Age Classification from Facial Images," *Computer Vision and Image Understanding*, **74**(1), pp. 1–21, 1999.

[3] Flores, G. M. "Senility of the face–Basic study to understand its causes and effects," *Plastic Reconstructive Surgery*, **36**, pp. 239–246, 1965.

[4] O'Toole, A. J., Vetter, T., Volz, H. and Salter, E. M. "Three dimensional caricatures of human heads: distinctiveness and the perception of age," *Perception*, **26**, pp. 719–732, 1997.

[5] Lanitis, A., Taylor, C. J. and Cootes, T. F. "Towards Automatic Simulation of Aging Effects on Face Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(4), pp. 442–455, 2002.
[6] Miller, H. M. and Seltzer, R. B. "The skin game: Keeping your skin younger-looking at any age," *LA Daily News*, Sunday, June 08, 2003.
[7] Joachims, T. "Making Large-scale SVM Learning Practical," in Schölkopf, B., Burges, C. and Smola, A. (Eds), *Advances in Kernel Methods – Support Vector Learning*, MIT Press, 1999.
[8] Yeasin, M. and Kuniyoshi, Y. "Detecting and tracking a human face using a space-varying sensor and an active head," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, South Carolina, USA, pp. 168–173, 2000.
[9] Rowley, H. A., Baluja, S. and Kanade, T. "Neural Network-Based Face Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(1), pp. 23–38, 1998.
[10] Lee, D. D. and Seung, H. S. "Algorithms for Non-negative Matrix Factorization," in *Proceedings of Neural Information Processing Systems*, pp. 556–562, 2000.
[11] De la Torre, F. and Black, M. J. "Robust Principal Component Analysis for Computer Vision" in *International Conference on Computer Vision (ICCV'2001)*, Vancouver, Canada, July 2001.
[12] Kohonen, T. *Self-Organizing Maps*, 2nd edition, Springer, 1997.
[13] Kambhatla, N. and Leen, T. K. "Dimension reduction by local principal component analysis," *Neural Computation*, **9**(7), pp. 1493–1516, 1997.
[14] Bruske, J. and Sommer, G. "Intrinsic dimensionality estimation with optimally topology preserving maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(5), pp. 572–575, 1998.
[15] Roweis, S. T. and Saul, L. K. "Nonlineaar dimensionality reduction by locally linear embedding," *Science*, **290**, pp. 2323–2326, 2000.
[16] Vapnik, V. N. *The Nature of Statistical Learning Theory*, Springer, 1995.
[17] Vapnik, V. N. and Chervonenkis, A. Y. "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability and its Applications*, **17**(2), pp. 264–280, 1971.
[18] Vapnik, V. N. *Estimation of Dependencies Based on Empirical Data*, Springer Verlag, 1982.
[19] Burges, C. "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, **2**(2), pp. 121–167, 1998.

# 14

# Intelligent Recognition in Medical Pattern Understanding and Cognitive Analysis

**Marek R. Ogiela**
**Ryszard Tadeusiewicz**
AGH University of Science and Technology, Institute of Automatics Al. 30 Mickiewicza, PL-30-059, Kraków, Poland

*This chapter presents a new approach to the application of picture languages for cognitive analysis and reasoning of selected medical visualization. We will show new opportunities for applying these methods to undertake the task of automatic understanding of image semantics in intelligent medical information or computer-aided diagnosis systems. These systems are applied in various tasks supporting decisions taken in the wide area of health care and medical imaging. The possibility of obtaining information about the semantic content of medical images may contribute considerably to the creation of new intelligent cognitive medical systems. This chapter shows that structural techniques of artificial intelligence may be applied in the case of tasks related to automatic classification and machine perception of semantic pattern content, in order to determine the medical meaning of the images. We describe some examples presenting ways of applying such techniques in the creation of cognitive vision systems for selected classes of medical image.*

## 1. Introduction

In this chapter, a new approach to the processing and analysis of medical images is proposed. We try to introduce the terms and methodology of medical data understanding as a new step along the way, starting from image processing, and followed by analysis and classification [1–5]. In the chapter we try to show that image-understanding technology, as the next step after image recognition, is useful (sometimes even necessary), possible and also effective. This will be done using four types of selected medical image. But in fact, the methods under consideration can also be used for many other kinds of medical image.

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

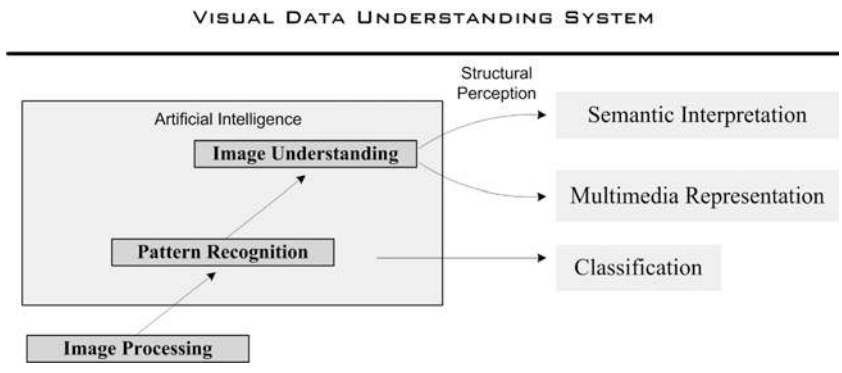



**Figure 14.1** Stages of image analysis and semantic classification in visual data understanding systems.

Further, we shall try to demonstrate that structural pattern analysis can be regarded as an effective method for medical image understanding, replacing simple recognition (Figure 14.1). Structural image analysis can be considered to be a totally new approach to the analysis and description of shapes of selected organs in medical imaging in general [6]. Examples of the application of syntactic methods of pattern recognition for the understanding and analysis of the selected medical images presented in this chapter show its usefulness for early diagnosis of some diseases of selected organs. The results of analysis of investigations, based on the structural analysis of selected types of medical image, confirm very good properties of our proposed methodology and algorithms.

The chapter will discuss, in particular, disease symptom recognition tasks for four selected areas of the body and types of medical image:

- analysis of coronary arteries seen on coronographic images [7,8]. This analysis is aimed at discovering the symptoms of ischemic heart disease.
- the renal pelvis with ureters visible on urograms of these structures [9]. An analysis of urograms allows diagnosis of some lesions characteristic of hydronephrosis or extrarenal uremia.
- pancreatic ducts visible on images obtained in the course of ERCP (Endoscopic Retrograde Cholangio-Pancreatography) examinations [10]. In this case, the objective of the analysis is early computer-aided diagnosis of neoplastic lesions and pancreatitis.
- the spine and spinal cord visualized on MR images. The objective is to detect and diagnose lesions that might evidence a whole range of various disease units (numerous forms of inflammatory conditions to the most serious cases of tumors).

The process of recognition will be based mainly on the use of context-free picture grammars and languages of description of shape features, as well as graph grammars used to recognize disease symptoms which can occur in these organs. This type of recognition may not only support the diagnosis of disease lesions, but it also constitutes intelligent information systems imitating the method of image interpretation and understanding used by qualified professionals.

The algorithms proposed in this chapter constitute a new proposal for an efficient and effective analysis of selected organ morphology, aimed at diagnosing pathological lesions in them. They also expand current analysis techniques to a considerable degree by offering possibilities to specify the semantic content of images, which can support diagnosis and further treatment directions. Moreover, they may also serve to provide a quick indexation and categorization of disease units in medical databases [11].

For a proper analysis of the mentioned changes, and for a verification of how advanced their level, an attributed context-free grammar of type LR and a graph grammar of type EDT [12,13] have been

proposed. These methods derive from mathematical linguistics and have been applied to detect changes in the width of different structures visible in graphs. These graphs are obtained from the application of a straightening transformation at the image preprocessing stage, which enables the production of graphs of straightened structures, while preserving the morphological lesions occurring in them [14–16].

The general methodology for the application of structural analysis to the creation of perceptual descriptions for analyzed structures and pathological signs is the following. First, simple shape elements are defined and the general grammar description of the analyzed organ is built. The basis for this description is a special kind of graph, tree or context-free grammar [13,17]. Using the actual shape of the organ in question, we can obtain a description in the form of sequences of terminal symbols. Such sequences belong to the languages generated by the introduced grammar. For each lesion, and in a healthy organ, the obtained description sentences are different, because every organ is unique. The main analysis and recognition of pathological signs are based on parsing algorithms, which analyze input sentences and reduce them to one of several known categories. For each case of illness, it is possible to obtain a unique parsing result, belonging to one of several classes. In the rare cases (from a diagnostic point of view) of equivalence of some classes, it is also necessary to establish the final result of recognition by applying simple methods based on specially defined semantic procedures [18].

Furthermore, this type of recognition in essence is an attempt to automate a specifically human process of understanding the medical meaning and the implications of the shapes of organs on a digital image. It is based on a perception, in other words, on a deeper understanding of the examined image [19,20].

The main advantage of using the presented grammars is that they offer a possibility to detect, on the obtained width profiles, both concentric stenoses, revealed on a regular cross-section by the monotonous stenosis of the whole lumen, and eccentric stenoses, revealed only on one side of the vessel. This property is especially useful for coronary artery diagnosis because it enables the determination of whether the detected symptom is characteristic of stable angina pectoris, in the case of concentric stenosis diagnosis, or unstable angina pectoris when an eccentric stenosis is revealed.

Further sections will present algorithms for initial analysis of the discussed images, as well as recognition methods and examples of results of disease lesion recognition in coronary arteries, upper urinary tracts, pancreatic ducts and the spinal cord.

## 2. Preliminary Transformation of Selected Medical Images

Cognitive analysis of the looked-for disease lesions on the discussed medical images, conducted with the use of the syntactic methods of image recognition proposed in this chapter, is possible only due to prior preparation of the examined image in a very special and highly targeted way. This means that the application of syntactic methods of image analysis must be preceded by the application of a special operation sequence in the preprocessing of the examined images. Due to this, the final objective of the analysis presented in this chapter will be to understand the essence of selected health problems connected with imminent myocardial infarction, urinary system atresia and pancreatic failure. Thus, the starting point for the analysis will be to visualize the pathological lesions of the respective organ shapes. Due to a need to eliminate various factors (e.g. individual variation) which change the appearance of the organs in question on an image, and thus change the meaning of the contents of the imaging (e.g. pointing to the occurrence of symptoms of a given disease), we looked for a form of preprocessing which would retain the information necessary to understand an image, and at the same time, eliminate information noise.

In the tasks presented in this chapter, we have the following examples of disturbing factors: the general course, size and shape, as well as location, of the analyzed anatomical structures on the image. All of these factors, ensuing both from the anatomical features of body build of an individual, and from the method used to take the concrete X-ray photograph, influence strongly the shape of structures visible on the image. Yet they do not determine the evaluation of the image as such (that is, the decision as to whether a given organ is healthy or if there are signs of pathology). A medical

doctor aiming to understand the essence of a patient's ailment takes out those unimportant features and focuses his/her attention on details important for the diagnosis. In the examples discussed here, those details are stenoses or dilations (sometimes also ramifications) of the appropriate vessels. This is why in the examples analyzed here, preprocessing of the examined images has been directed towards obtaining straightened (which means that the vessel course is independent of individual variations) and appropriately smoothed width diagrams of the vessel. They are the carriers of information necessary to understand if we are dealing with a healthy organ or one with pathological lesions. Later, we shall see examples of such width diagrams obtained as a result of preprocessing of the analyzed coronary arteries, ureters, main pancreatic ducts and spinal cord. Those diagrams are subsequently approximated by means of segments of an open polygon, and segments are characterized by means of symbols of a selected grammar. Owing to this, we obtain a description of the examined organ shape, limiting considerably the representation of those fragments (which usually take a lot of space on the image) which can be considered to be physiologically correct; it shows all pathological lesions of interest to us occurring on the examined structures.

The above-presented concept of preprocessing medical images in preparation for the analysis process, with the aim of reaching an automatic understanding of the essence of pathological lesions shown on them, contains many important and difficult details. Their solution was necessary for the functioning of the method described here. We shall comment briefly on some of the numerous problems associated with image preprocessing which had to be undertaken and solved before it was possible to describe attempts at automatic understanding of the medical content and the essence of information about deformations visible on images.

First, let us emphasize that in order to achieve the aimed-for objectives, it was necessary to obtain width diagrams of the analyzed structures in such a way that they illustrated both concentric and eccentric stenoses [21]. It is possible to achieve this objective by the application, at the preprocessing stage, of an approach to determining the central line and creating a width diagram slightly different than the ones proposed by other researchers [7,22]. The conducted research has shown that in order to obtain the central line of the analyzed vessel, rather than the operation of morphological erosion of the binarized vessel proposed by some researchers, or manual positioning (performed by a specialist) of the location of the said line, we should use a different technique which gives much better results. Similar research has demonstrated that in the course of attempts at automatic understanding of pathological lesions of the structure of the examined organ, and of deformations of the analyzed organ shape indicating those, the best results are obtained when the reference line is a curvilinear axis of the vessel. This axis can be laid out by the method of skeletonization of the analyzed anatomical structures used by the authors. The method is based on an appropriately modified Pavlidis algorithm [23]. The said method is efficient in calculations, and the central line obtained in its result, with precisely central location in relation to its walls (even in the case in which they are pathologically deformed), creates a certain path along which width profiles are determined.

If we take a numerically straightened axis line of the analyzed vessel for our reference point, then we obtain the diagrams of its unilaterals (the so-called top and bottom or left and right). These diagrams are obtained from the use of a specialist algorithm, the so-called straightening transformation, prepared by the authors; it allows one to obtain width diagrams of the analyzed structures, together with morphological lesions occurring in them [8]. These diagrams are subsequently approximated with a polygon, which constitutes the basis for the further-discussed process of automatic understanding of pathological deformations of the analyzed organs shown on images.

To summarize the above, we can say that the following operations should be conducted in the course of preprocessing of the examined images (in square brackets are references to bibliographic items supplying all the necessary technical details relating both to the algorithms and details of implementation):

- Segmentation and filtering of the analyzed images [24,25].
- Skeletonization of the analyzed structures of selected vessels shown on images, e.g. on images of coronary arteries, main pancreatic ducts, ureters and spinal cords [21,23].

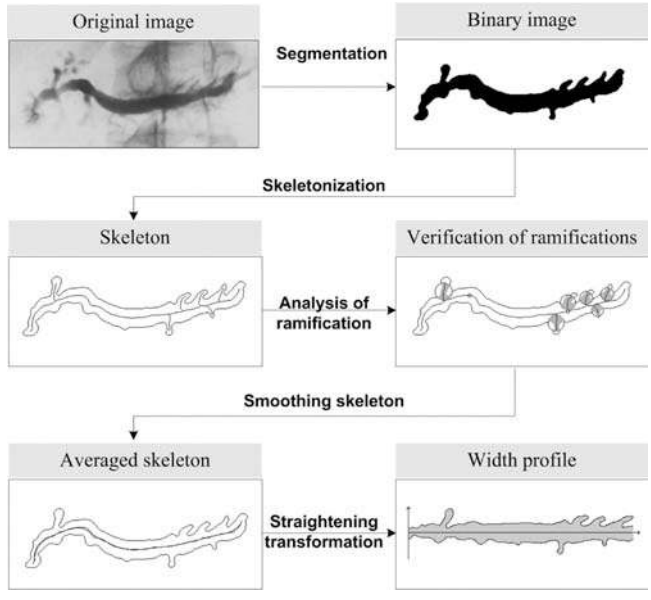



**Figure 14.2** Operations conducted in the course of preprocessing of the examined images. At the top is the original image of a pancreatic duct showing symptoms of chronic inflammation. At the bottom, a diagram of the pancreatic duct contours with visible morphological lesions.

- Analysis of real, and verification of apparent, skeleton ramifications [26,27].
- Smoothing the skeleton by averaging its elements [28].
- The application of a specially prepared straightening transformation, transforming the external contour of the examined structures in a two-dimensional space to the form of 2D width diagrams, showing contours of the 'numerically straightened' organ. This transformation retains all morphological details of the analyzed structures (in particular their deformations and pathological lesions) [29]. For this reason, it is a convenient starting point for further analysis of the properties of shape features of the analyzed structure and for detecting such deformations (with the use of syntactic methods of image recognition). This constitutes the basis for an automatic understanding of the nature of pathological lesions, and for a diagnosis of the disease under consideration.

Recognition and automatic understanding of the looked-for lesions of organ shapes with the use of syntactic methods of image recognition [30] is possible with a prior application of the specified sequence of operations, which together constitute the previously mentioned image preprocessing stage. Details relating to individual stages of preprocessing of the analyzed images were discussed at length in the above-quoted publications of the authors. These stages have also been specified in Figure 14.2.

## 3. Structural Descriptions of the Examined Structures

Syntactic image analysis, aimed at understanding the looked-for lesions, which are symptoms of some defined diseases, has been conducted on all examples analyzed here, based only on the obtained width diagrams of vessels shown on those images (following the above-described image preprocessing). To diagnose and describe the analyzed lesions in the examined structures, we have used context-free grammars of the LR(1)-type [13,30]. These grammars allow one (with an appropriate definition of

primitive components) to diagnose and describe in an unambiguous way all lesions and pathological deformations important from the point of view of the diagnosis tasks analyzed here. The key to success in every task performed was an appropriate definition of the primitive component set (alphabet of graph primitives), allowing the recording of every examined organ shape (both correct and pathologically changed) as a regular expression in the language defined by the analyzed grammar. As has already been stressed above, this task was simplified due to the fact that primitive components were defined on the vessel width diagrams obtained (as a result of preprocessing) after an execution of the straightening transformation. This did not mean, however, that indicating the necessary primitive components (and finding terminal symbols corresponding to those components) was a very easy task.

It is at the stage of selecting the primitive components that the author of the medical image understanding method described in this chapter must, for the first time (although not for the last time), cooperate closely with a team of experienced medical doctors. These experienced diagnosis experts are the only people competent to tell which features of the analyzed contour of a selected organ part are connected with some stages of natural diagnostic reasoning. The role of the producers of the optimum set of 'letters' of the created grammar is strictly divided: a doctor can (and should!) focus his/her attention on the largest number of details possible, treating every image discussed here as a totally new, separate case. A computer scientist must try to integrate and generalize the detailed information in such a way that the result is a maximally compact set of as few primitive components as possible; the components must allow the building of a computationally effective graph grammar [12,31].

To use a concrete example, we should state that in the tasks selected and discussed here, there was a need to determine primitive components which allowed the description of pathological deformations, i.e. which edges of the analyzed structures should appear on the obtained width profiles. In accordance with the opinions of doctors with whom we consulted, shape features important for diagnostics are: pathological stenoses or dilations of the examined vessels and local changes of their contour, such as side ramifications and cysts.

After analysis of many example images, it turned out that morphological lesions with very differing shapes played an identical role in the diagnostic reasoning of doctors. Therefore, for the aggregation of the various forms of deformation of the contour line of the examined vessels, it was possible to use the line approximation algorithm of the previously obtained vessel width diagram for identical sets (sequences) of primitive components. An approximation of the complete edge line of the diagram by means of a polygon is a method described in [32]. As a result of the application of this method for every diagram, we obtain a sequence of segments approximating its external contours. This sequence simplifies the analyzed image once again, yet it still retain all information important from the point of view of the constructed automatic diagnostic reasoning sequence. It is worth noticing that, at this stage, we have a very uneven compression of the primitive image information. This is beneficial for the 'concentration of attention' of the recognizing diagram operating on these contour fragments, which are really important. If a fragment of the examined vessel at some (even very long) segment of its course has a smooth form, deprived of morphological features, which can be important for the recognition process, then the whole approximated segment is one section of a polygon and it will be represented by one symbol in a notation based on a sequence of identifiers of the discovered graphic primitives. If, on the other hand, a contour fragment is characterized by big changes of the edge line, then the proposed form of description will attribute many segments of a polygon; this will mean a big representation in the final linguistic notification. This is rightly so, since, for diagnosis, it is an important fragment!

Next, depending on parameters, terminal symbols are attributed to each of the polygon segments. In the examples considered here, one parameter was enough to obtain a satisfactory specification of the description of the analyzed images. The parameter characterized every successive segment of the approximating polygon in the form of its inclination angle. To be more exact: the appropriately digitized values of the angle were given in the form of primitive components. The digitation pattern used could be treated as a counterpart of a dictionary of the introduced language of shape features. Yet, in more complex tasks, we can imagine a situation in which the set of primitive components can

also depend on further features of the polygon approximating the examined shape, e.g. on the length of individual segments or their mutual location.

Of course, languages describing separate classes of medical image are different [3,30,31]. Thus, concrete rules governing the creation of languages of primitive components for languages in each class are different. Formulae for concrete operations, aimed at building a specialist grammar and specialist language of description of shape features in everyone of the cases described here will be presented below, while the task of recognition of types of disease based on appropriate X-ray images is also described. It will be possible to notice easily that, in every case, the final result of the operation is a sequence of terminal symbols, which are the input into appropriately designed syntax analyzers [31,33] and syntactic transducers, the basic tools for the implementation of the process of automatic understanding of medical images described in this chapter.

## 4. Coronary Vessel Cognitive Analysis

In this section, we present methods of computer-aided diagnosis for the recognition of morphological lesions of coronary vessels with the use of syntactic methods of image recognition. Recognizing such lesions is extremely important from the point of view of correct diagnosis of myocardial ischemic states caused by coronary atheromatosis lesions resulting in stenoses of the artery lumen, which in turn lead to myocardial ischemic disease. This disease can take the form of either stable or unstable angina pectoris or myocardial infarction [7].

The objective of the methods described in this section will be to diagnose the stenoses of coronary arteries, in particular the so-called important stenoses: artery lumen stenoses, which exceed 50 % and occur in the left coronary artery trunk, as well as stenoses exceeding 70 % of the artery lumen in the remaining segments of coronary vessels. The importance of a correct diagnosis of such lesions is demonstrated by the fact that closing the lumen of one of the left coronary artery branches, e.g. the interventricular anterior artery, can constitute a threat to life due to the fact that it leads to ischemia or necrosis of more than 50 % of the left ventricle cardiac muscle. Examples of images of coronary arteries with stenoses are shown in Figure 14.3.

In order to diagnose correctly and define the degree to which lesions have advanced, we have proposed a context-free picture grammar of the LALR(1) type. This grammar allows one to diagnose effectively this type of irregularity shown on X-ray images obtained in the course of coronography examinations. The main advantage of the application of context-free grammars, as compared to analysis

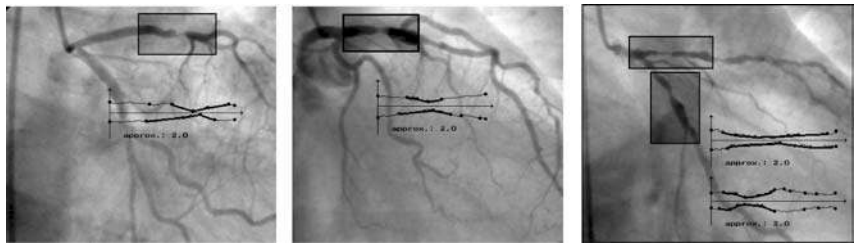


**Figure 14.3** Images of coronary arteries obtained in the course of coronography examination. The frames mark important strictures of the examined arteries, which will be localized and analyzed with the use of structural pattern recognition methods. The figures also show the results of recognition of pathological lesions. The graphs present width profiles of the examined artery sections with strictures. On the diagrams, bold lines mark the areas diagnosed by a syntax analyzer as the places of occurrence of pathological strictures.

methods proposed by other researchers [7,22], is a possibility to diagnose–on the obtained width profiles of the examined artery–both concentric strictures, which in a cross-section are seen as a monotonous stenosis of the whole lumen, and eccentric stenoses, which occur on only one vessel wall. This fact is important from the diagnostic point of view since it allows one to discover if the identified symptom is characteristic for stable angina pectoris (if a concentric stricture is discovered) or unstable angina pectoris (if an eccentric stenosis is discovered) [7].

The following attributed grammar has been proposed to diagnose various types of stenosis shape: $G_{CA} = (V_N, V_T, SP, STS)$, where:

$V_N = \{STENOSIS, U, H, D\}$–the set of non-terminal symbols
$V_T = \{h, u, d\}$for $h \in [-10°, 10°], u \in (10°, 90°), d \in (-10°, -90°)$–the set of terminal symbols
$STS = STENOSIS$–the grammar start symbol
Production set SP is defined in Table 14.1.

In the presented grammar, the first in the production set sequence defines the potential shapes of stenoses which can occur in the coronary vessel lumen. Further introductory steps in this grammar describe a linguistic formula defining the descending and ascending parts of the analyzed stricture. The last production defines a horizontal segment, which can occur between those parts. Semantic variables $h_e$ and $w_e$ define the height and length of the terminal segment labeled $e$. Their role in diagnosing and presenting the diagnosis to doctors is auxiliary. The considerable simplicity of the grammar presented here results from a small number of morphological lesions, which this grammar describes. It also proves the significant generation power of context-free grammars applied to analyze and recognize medical images. The use of attributes is aimed at determining additional numeric parameters of the diagnosed stricture, which allow the determination of the percentage rate of the coronary artery lumen stenosis, important for the prognosis of the patient's state.

With the application of the context-free grammar presented in this section, it is possible to diagnose various types of coronary stricture with great precision. In the case of syntactic analysis with the use of those grammars, we are dealing with a situation in which the recognizing software almost automatically supplies practically complete information about irregularities of the examined arteries.

The image set of test data, which was used in order to determine as a percentage the efficiency of correct recognition of the size of stenoses in coronary arteries, included 55 different images obtained for patients with heart disease. In this set, we considered image sequences of patients previously analyzed at the grammar construction stage and by the recognizing analyzer. In order to avoid analyzing identical images, we selected separate images occurring a number of positions before or after the ones used originally (from DICOM sequences). The remaining images in the test data were obtained for a new group of patients (25 people), including five people who had previously undergone angioplasty, and in whose cases a restenosis of the previously dilated vessel had occurred. The objective of an analysis of these data was to determine as a percentage the efficiency of correct recognition of an artery stenosis, and to determine its size using the grammar introduced. On the image data tested, the efficiency of

**Table 14.1** Definition of grammar rules from production set SP.

| Changes | Grammar rules | Semantic actions |
|---|---|---|
| Stenosis | 1. $STENOSIS \rightarrow D\ H\ U \mid D\ U \mid D\ H$ | Lesion = Stenosis |
| | 2. $H \rightarrow H\ h \mid h$ | $w_{sym} = w_{sym} + w_h; h_{sym} = h_{sym}$ |
| | 3. $D \rightarrow D\ d \mid d$ | $+h_h$ |
| | 4. $U \rightarrow U\ u \mid u$ | ... |

recognition amounted to 93 %. It is worth emphasizing that this value refers to *automatic* analysis and determination of the size of the stenoses using the procedure introduced, without the need to correct the external vessel contours manually. In this case, the value of the efficiency of recognition is determined by the percentage fraction of the accurately recognized and measured vessel stenoses compared to the number of all images analyzed in the test. The recognition itself meant locating and defining the type of stenosis, e.g. concentric or eccentric.

Figure 14.3 presents examples of recognizing the looked-for lesions in coronary artery images analyzed in this section. Recognized symptoms have been marked with a bold line.

## 5. Understanding of Lesions in the Urinary Tract

This section will present the methodology for analyzing and diagnosing morphological lesions occurring in upper urinary tracts. The diagnosis of these lesions has been conducted based on analysis of radiograms (urograms) of the renal pelvis and upper segments of urinary tracts [9]. In the case of analysis of renal radiograms, the main task is to recognize local stenoses or dilations of upper segments of urinary tracts (examples of these images are presented in Figure 14.4) and attempt to define the correct morphology of the renal pelvis and renal calyxes. Lesions in these structures can suggest the occurrence of renal calculi or deposits, which cause ureter atresia and can lead to diseases such as acute extrarenal uremia or hydronephrosis.

An analysis of the correct morphology of the ureter lumen will be conducted with the use of context-free attributed grammar. The analysis of the correctness of the renal pelvis and renal calyx will be conducted with the use of a tree grammar. Tree methods of syntactic image recognition are generally used to analyze more complex objects, textures or scenes in an image [13] and this is why

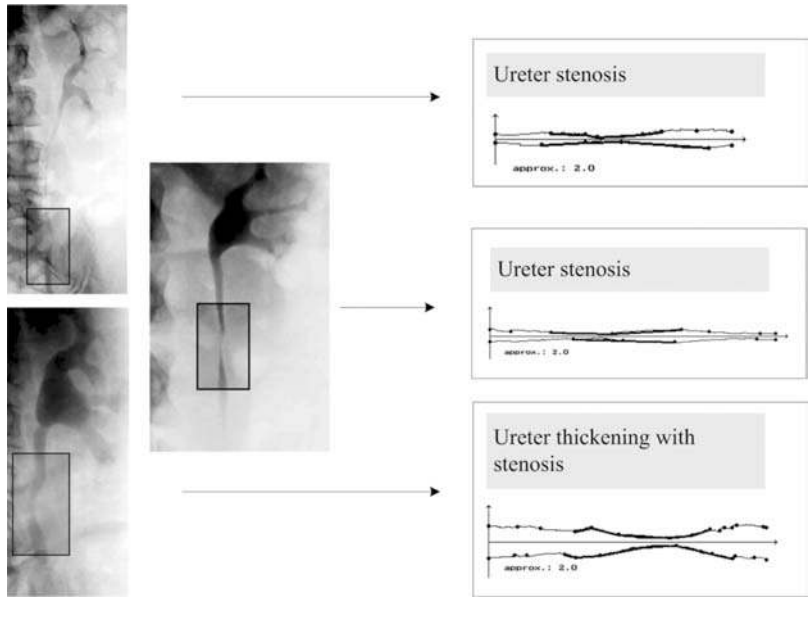



**Figure 14.4** Images obtained from urography examinations showing upper sections of the urinary tracts. Visible renal pelvises and urinary tract sections with ureterostenosis are shown in the frames. The results of diagnosing disease symptoms using syntactic methods are also presented.

for nephrogram analysis they have been used to verify the correctness of the morphology of the renal pelvis and renal calyx.

Diagnosing morphological lesions in the form of ureter stenoses or dilations has been conducted with the use of the following attributed grammar: $G_U = (V_N, V_T, SP, STS)$, where, as in the previous case, $V_N$ stands for a set of non-terminal symbols, $V_T$ a set of terminal symbols, SP the production set and STS the start symbol.

$$\begin{aligned} V_N &= \{\text{LESION, STENOSIS, DILATATION, HOR, SLOPE\_UP, SLOPE\_DOWN}\} \\ V_T &= \{\text{h, su, sd}\} \text{for } h \in [-8°, 8°], su \in (8°, 180°), sd \in (-8°, -180°) \\ STS &= \text{LESION} \end{aligned}$$

Production set SP is defined in Table 14.2.

The grammar presented here, like its counterpart used to diagnose stenoses of the coronary artery lumen, is not a very complex grammar. This is due to the fact that it defines a small number of morphological lesions, which can reveal ongoing disease processes. It allows one to diagnose only various forms of stenosis and dilation characteristic of various disease units. The use of attributes also allows determination of the numerical parameters of morphological lesions. The simplicity of grammars introduced so far results mainly from the large generation capacity of context-free grammars, understood mainly as possibilities to describe complex shapes by means of a small number of introductory rules, that is grammar productions. We shall see a totally different situation in analyzing and diagnosing pathological lesions in the main pancreatic ducts, where a larger number of symptoms to be recognized and the variety of their shapes will result in the fact that the complexity of the looked-for pathologies on the external contours can be ensured only by a more complex grammar, together with an auxiliary recognition procedure based on the use of languages of description of shape features.

The analysis of the renal pelvis and renal calyx was accomplished with the use of $G_{EDT}$ expansive tree grammar, generating trees with directed and labeled edges, that is EDT trees [13].

Even in the correct conditions, that is without visible lesions in the morphology, these structures are characterized by an extraordinary variability of shape. However, it is possible to find some common

**Table 14.2** Definition of grammar rules from production set SP.

| Changes | Grammar rules | Semantic actions |
|---|---|---|
| Stenosis | 1. LESION → STENOSIS | Lesion = Stenosis |
| | 2. STENOSIS → SLOPE_DOWN HOR SLOPE_UP | |
| | 3. STENOSIS → SLOPE_DOWN SLOPE_UP | |
| | 4. STENOSIS → SLOPE_DOWN HOR | |
| Dilatation | 5. LESION → DILATATION | Lesion = Dilatation |
| | 6. DILATATION → SLOPE_UP HOR SLOPE_DOWN | |
| | 7. DILATATION → SLOPE_UP SLOPE_DOWN | |
| | 8. DILATATION → SLOPE_UP HOR | |
| | 9. HOR → HOR h \| h | $w_{sym} = w_{sym} + w_h$; $h_{sym} = h_{sym} + h_h$ |
| | 10. SLOPE_DOWN → SLOPE_DOWN sd \| sd | ... |
| | 11. SLOPE_UP → SLOPE_UP su \| su | |

features characteristic for all correct structures [34]. Those features are, first of all, the number of smaller and bigger calyxes in the renal sinus. In correct structures there are usually only two or three bigger calyxes entering the renal pelvis; in turn, those are formed by eight to ten smaller calyxes occurring in them. Smaller calyxes end with concave renal papillae forming peaks of renal pyramids [27,34].

Analysis of such regularities was made by means of an expansive tree grammar defined especially to analyze the morphology of skeletons of the examined renal pelvis and renal calyx. Skeletonization was made with the use of the Pavlidis skeletonization algorithm. The skeleton obtained as a result of thinning was further analyzed in the following way: the first skeleton ramification point is the beginning of skeletons of bigger calyxes, thus, finding the number of ramifications originating in them allows one to determine the number of bigger calyxes. Determining in every bigger calyx the ramifications of the next order calyx allows one to determine the number of smaller calyxes originating from it.

On the other hand, in smaller calyxes, the analysis of end points in a skeleton allows one to determine if they ramify further or not. This can show if a given renal papilla is not concave towards the calyx interior but convex towards the renal pyramid, as is the case with hydronephrosis, cysts or neoplasm processes [31]. Skeletonization of convex renal papillae leads to the formation of skeletons of those structures without further ramifications. The result of skeletonization of correct renal papillae, that is concave renal papillae, is skeleton lines with visible fork-like ramifications [31].

A tree grammar describing the correct skeletons of the renal pelvis and renal calyx will be defined in such a way that the tree root will be defined by the location of the point where bigger calyxes ramify; its successors will be determined by ramification points of the 2nd order, that is, the beginning of smaller calyxes. The last layer of peaks is defined by ramification points of the 3rd order, that is, by ramifications occurring if a renal papilla has a concave shape.

To define primitive components used in the tree grammar, we can use the approximation algorithm of skeleton ramification in the renal pelvis and renal calyx, which allows one to identify every ramification with a single segment, whose ends are determined by the ends of the approximated ramification. Next, edge terminal labels are attributed to each of the determined segments, depending on their angles of inclination (see grammar definition). As a result of the operation we obtain a representation of the analyzed object in the form of a tree.

To diagnose morphological changes in renal pelvises, the following graph grammar was used: $G_{edt} = (\Sigma, \Gamma, r, P, Z)$, where $\Sigma = \Sigma_N \cup \Sigma_T$ is a set of terminal and non-terminal vertex labels, $r$ is a function which assigns to the graph vertex the number of its consequents, Z is a finite set of starting graphs, $\Gamma$ is a set of edge labels, and P is the production set.

$$\Sigma_T = \{\text{renal\_pelvis, bigger\_calyx, smaller\_calyx, papilla\_renalis}\}$$
$$\Sigma_N = \{\text{RENAL\_PELVIS, BIGGER\_CALYX, SMALLER\_CALYX}\}$$
$$\Gamma = \{x \in (30°, 180°), y \in [30°, -30°], z \in (-30°, -180°)\}$$
$$Z = \{\text{RENAL\_PELVIS}\}$$

Production set P is defined in Table 14.3.

The first production group defines the different kinds of normal renal pelvis, i.e. having two or three smaller calyxes. The succeeding productions define the form of bigger calyxes formed from two or more smaller calyxes. The last group defines the proper form of renal papillae, which give a fork form during the skeletonization, i.e. it finishes with short branches which arise only when it is concave to the interior of a smaller calyx. Convex forms during skeletonization are thinned to the line without end branches, which results from the properties of skeletonization algorithms.

Interpretation of disease lesions in the upper urinary tracts was conducted based on a test set of images composed of 30 X-ray images showing disease lesions. These images can be divided into two groups interpreted by a specialist as cases with in-born lesions and those with lesions acquired as a result of ongoing disease processes (renal calculi or hydronephrosis). All of them revealed ureter lumen stenosis or anomalies in the structure of the renal sinus. The test conducted showed that the recognition efficiency of such lesions on the test data set equals about 90 %. In the remaining cases,

**Table 14.3** Graph grammar production set P.

| Structure | Grammar rules |
|---|---|
| Renal pelvis | RENAL_PELVIS →<br>renal_pelvis (x BIGGER_CALYX y BIGGER_CALYX z BIGGER_CALYX)<br>\| renal_pelvis (x BIGGER_CALYX y BIGGER_CALYX)<br>\| renal_pelvis (x BIGGER_CALYX z BIGGER_CALYX)<br>\| renal_pelvis (y BIGGER_CALYX z BIGGER_CALYX) |
| Calyxes major | BIGGER_CALYX →<br>→ bigger_calyx (x SMALLER_CALYX y SMALLER_CALYX<br>z SMALLER_CALYX)<br>\| bigger_calyx (x SMALLER_CALYX y SMALLER_CALYX)<br>\| bigger_calyx (x SMALLER_CALYX z SMALLER_CALYX)<br>\| bigger_calyx (y SMALLER_CALYX z SMALLER_CALYX) |
| Calyxes minor | SMALLER_CALYX → smaller_calyx (x papilla_renalis) \| smaller_calyx<br>(y papilla_renalis)<br>\| smaller_calyx (z papilla_renalis) \| smaller_calyx (x papilla_renalis<br>ypapilla_renalis)<br>\| smaller_calyx (x papilla_renalis z papilla_renalis)<br>\| smaller_calyx (y papilla_renalis z papilla_renalis) |

interpretation difficulties resulted mainly from a change of the pathology character from local and distinct to blurred. The lesions in question are very large, originating as a result of neoplastic lesions, which lead to a complete change of shape or disappearance of the analyzed structures. Interpretation correctness depends also on the size of the approximation threshold, which, for considerably dilated ureters, should be selected in a way dependent on the size of the lesion observed. Such cases often remain not fully recognized and they can impact lowering the entirety of the method's efficiency. Figure 14.4 shows examples of diagnosing the looked-for lesions in urinary tracts.

## 6. Syntactic Methods Supporting Diagnosis of Pancreatitis and Pancreatic Neoplasm

The objective of ERCP image analysis is to diagnose morphological lesions of pancreatic ducts characteristic of neoplasm and chronic pancreatitis. The most important symptoms characteristic for pancreatic neoplasm are, first of all, the occurrence of local stenoses or dilations of the main pancreatic duct, as well as the occurrence of cysts or cavernous projections on the external edges of the pancreatic duct. Ducts with symptoms of chronic pancreatitis can be characterized by the occurrence of irregular side ramifications, as well as local stenoses or dilations (Figure 14.5). The enumerated symptoms will be recognized by a context-free attributed grammar specially defined for this purpose, and additionally by languages of description of shape features allowing one to diagnose quickly various types of pathological convexity or stenosis.

Analysis of the looked-for morphological lesions was conducted based on width diagrams obtained during image preprocessing, with the use of context-free attributed grammars of the LR(1)-type. Such grammars, like those in the previous tasks, with appropriate definition of primitive components on width diagrams and primitive terminal symbols corresponding to those, allow one to diagnose on an

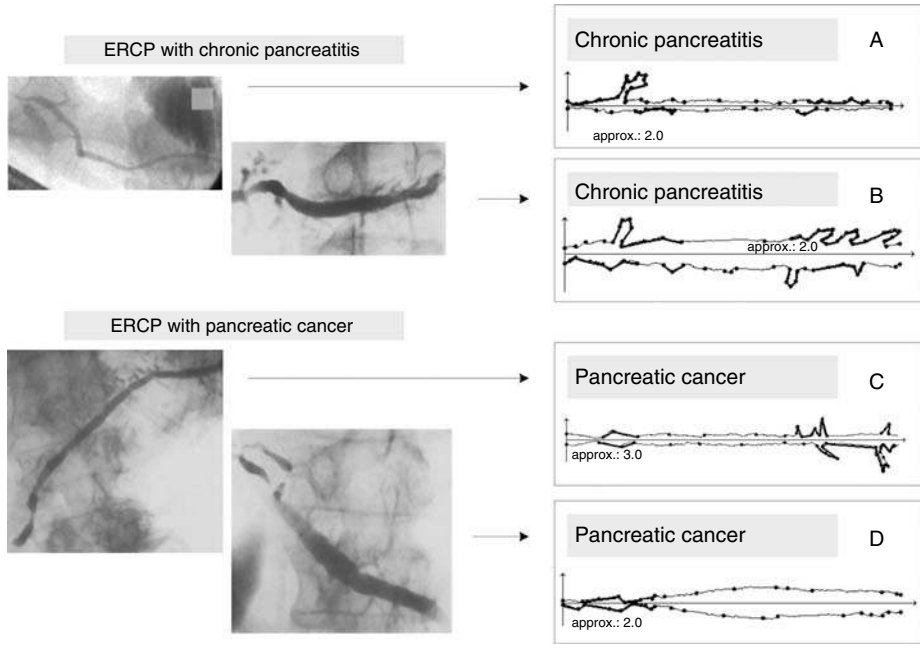



**Figure 14.5** ERCP images showing pancreatic ducts with symptoms of chronic pancreatitis and pancreatic cancer. The results of disease symptom recognition using syntactic methods are also shown.

ERCP image all pathological lesions important from a diagnostic point of view, that is cysts, strictures, dilations and ramifications.

The following attributed grammar has been used to describe morphological lesions in the examined pancreatic ducts: $G = (V_N, V_T, SP, STS)$, where:

$$V_N = \{\text{LESION, CYST, STENOSIS, DILATATION, BRANCH, P, S, G, I, N, NS, NG, NI, NN}\}$$
$$V_T = \{\text{p, s, ns, g, ng, i, ni, n, nn}\}$$
$$STS = \text{LESION}$$

Production set SP is defined in Table 14.4.

The semantic rules of the first productions specify the type of looked-for disease lesion, while the productions define the basic types of such diseases. Further productions define possible shapes of such pathologies. The last group specifies the shapes of descending and ascending arms in the morphology of such disease symptoms. The semantic variables of those productions allow specification of numerical parameters of the recognized lesion.

For a correct description and recognition of symptoms for which a 2D analysis is necessary on the obtained width diagrams (that is, for example, big cavernous projections), additional languages of description of shape features with multidirectional sinquad distribution have been used [26]. Those languages have also been used due to the impossibility of defining precisely all potential shapes which the looked-for lesions can take. The main advantage of those languages is that the recognition process can be based on an analysis of features describing the essence of those shapes, that is, the occurrence of lesions such as, for example, concavities or dilations. Those languages, together with the application of a sequential transducer with multiple input, allow one to recognize some lesions without a need

**Table 14.4** Production set SP defining pathological changes in pancreatic ducts.

| Lesion | Grammar rules | Semantic actions |
|---|---|---|
| Cyst | 1. LESION → CYST | Lesion = Cyst |
| | 2. CYST → I P NI \| G P NG \| I P NG \| G P NI \| I S NI \| G S NG \| I S NG \| G S NI \| I NS NI \| G NS NG \| I NS NG \| G NS NI | |
| Stenosis | 3. LESION → STENOSIS | Lesion = Stenosis |
| | 4. STENOSIS → NS S \| NS G \| NS P S \| NS P I \| NG S \| NI NS I \| NI S | |
| Dilatation | 5. LESION → DILATATION | Lesion = Dilatation |
| | 6. DILATATION → S P NG \| S G NS \| S NS \| G NS | |
| Branch | 7. LESION → BRANCH | Lesion = Branch |
| | 8. BRANCH → I NI \| I NS \| I P NI NN \| I NS NI NN \| G NI \| G S NN \| G P NN \| G S NI NN \| S NG \| S NS NN \| N G NG NI | |
| | 9. N → n \| N n | |
| | 10. NN → nn \| NN nn | . . . |
| | 11. I → i \| I i | |
| | 12. NI → ni \| NI ni | $w_{sym} = w_{sym} + w_{terminal\ symbol}$ |
| | 13. G → g \| G g | |
| | 14. NG → ng \| NG ng | $h_{sym} = h_{sym} + h_{terminal\ symbol}$ |
| | 15. S → s \| S s | |
| | 16. NS → ns \| NS ns | . . . |
| | 17. P → p | |

to define their morphology precisely, but based only on transitions between the so-called sinquads (that is, specified directions of the sectors approximating the width diagrams). This analysis is more universal in its character than an analysis using grammars, due to the fact that it allows identification of all types of convexity, ramification and stenosis–even in situations in which those lesions are not recognized by the previously described context-free grammar.

Despite the complicated grammatical reasoning process, in this case, the syntactic methods of image recognition also supply practically all information on morphological lesions on the external edges of the pancreatic ducts analyzed here.

As for the grammars analyzing coronary arteries and ureters, for the sequential grammar presented in this section, a syntax analyzer has been constructed in the form of a parser generated with the use of the YACC compiler. This analyzer was subject to a number of experiments; as a result it obtained very good results in recognizing the looked-for lesions on ERCP images. In the case of analysis of those images, for the shape feature analysis procedure, a sequential transducer was also implemented, a counterpart of the syntax analyzer for languages of description of shape features.

The efficacy of this method of computer-aided diagnosis of cancerous and inflammatory lesions based on ERCP pictures can be estimated at 90 %. This value stands for the percentage of correct recognitions of symptoms defined in the grammar: symptoms in the form of strictures, ramifications and lesions with cyst-like features. The remaining 10 % of the analyzed pancreatic images, which were

not fully correctly interpreted, can be considered to be cases difficult to interpret. Such images can present, among others, incorrect ramifications looping on the main image with the pancreatic duct, or show amputation (discontinuation) of the duct. Recognition of these may require either 3D analysis or independent interpretation of separated duct parts. Some images, as in the case of artery and ureter images, should be analyzed with a dynamically selected approximation threshold. These cases are also included in the group of images that are difficult to interpret. Figure 14.5 presents examples of recognition of looked-for lesions in ERCP images.

## 7. Semantic Analysis of Spinal Cord NMR Images

Another interesting application of the syntactic approach to automatic perception analysis involves the morphological classification of diseases of the spinal cord and surrounding spinal meninges [35]. This section provides an example of the analysis of such structures based on the scrutiny of congenital and developmental anomalies in the cord structure, well apparent in Magnetic Resonance Images (MRI).

There is a great variety of central nervous system and spinal cord diseases of variable etiology, including congenital and developmental anomalies revealed as lesions of spinal cord sections, or as dysfunctions of the nervous system. Apart from most common spinal cord diseases, the list includes cerebrospinal meningitis caused by bacteria, viruses or pathogenic parasites entering the central nervous system, polioencephalitis or poliomyelitis, inflammatory conditions in the whole cord width either diffused, disseminated or focused. These serious cases might be subjected to semantic analysis using structural classifiers. Diffuse diseases of the spinal cord may be caused by lesions in anterior and posterior spinal arteries. The occlusion of blood vessels might cause significant deterioration of the patient's condition, leading to a paralysis of sometimes even all limbs. Spinal cord damage caused by vascular problems might originate from the spinal cord tissue, sensory roots of a spinal nerve, meninges, a vascular rete in the spinal cord, the vertebrae or organs in the direct vicinity of disks. There might also be metastases from other organs.

In the case of analysis of backbone and spinal cord MRI images, the chief objective in the recognition stage is to detect and diagnose lesions that might evidence a whole range of various disease units: from myelomeningocele, numerous forms of inflammatory condition or cerebral or spinal cord ischemia, to the most serious cases of intra- and extramedullary tumors. An unambiguous identification of all units with the use of one recognizing software is extremely difficult, due to rather subtle differences that are decisive for the correct classification of every one of them. All the same, structural analysis proves to be extremely useful in the specification of the degree of the disease unit development by means of specifying the size of lesions in the morphology of the cord and by defining the compression of the spinal cord and meninges [35].

The analysis of this structure uses a developed context-free grammar. It allows us to identify the symptoms and to draw diagnostic conclusions relating to the inner nature of the visible pathology.

The grammar developed for the analysis of spinal cord images is defined as follows: $\mathrm{Gsc} = (V_N, V_T, \mathrm{STS}, \mathrm{SP})$, where:

$$V_N = \{\mathrm{LESION}, \mathrm{NARROWING}, \mathrm{ENLARGEMENT}, \mathrm{H}, \mathrm{E}, \mathrm{N}\}$$
$$V_T = \{\mathrm{h}, \mathrm{e}, \mathrm{n}\} \text{for } \mathrm{h} \in [-11°, 11°], \mathrm{e} \in (11°, 180°), \mathrm{n} \in (-11°, -180°)$$
$$\mathrm{STS} = \mathrm{LESION}$$

SP is defined in Table 14.5.

This grammar permits one to detect different forms of narrowing and enlargement which may characterize the different disease units (for example neoplasm or inflammation processes). The results of spinal cord morphology analyses aimed at locating lesions are provided in Figure 14.6.

**Table 14.5** Production set defining changes in the spinal cord.

| Lesion | Grammar rules | Semantic actions |
|---|---|---|
| Dilatation | 1. LESION → ENLARGEMENT | Lesion = enlargement |
| | 2. ENLARGEMENT → E H N \| E N \| E H | |
| Stenosis | 3. LESION → NARROWING | Lesion = narrowing |
| | 4. NARROWING → N H E \| N E \| N H | |
| | 5. H → h \| h H | $w_{sym} = w_{sym} + w_h$; $h_{sym} = h_{sym} + h_h$ |
| | 6. E → e \| e E | |
| | 7. N → n \| n N | . . . |

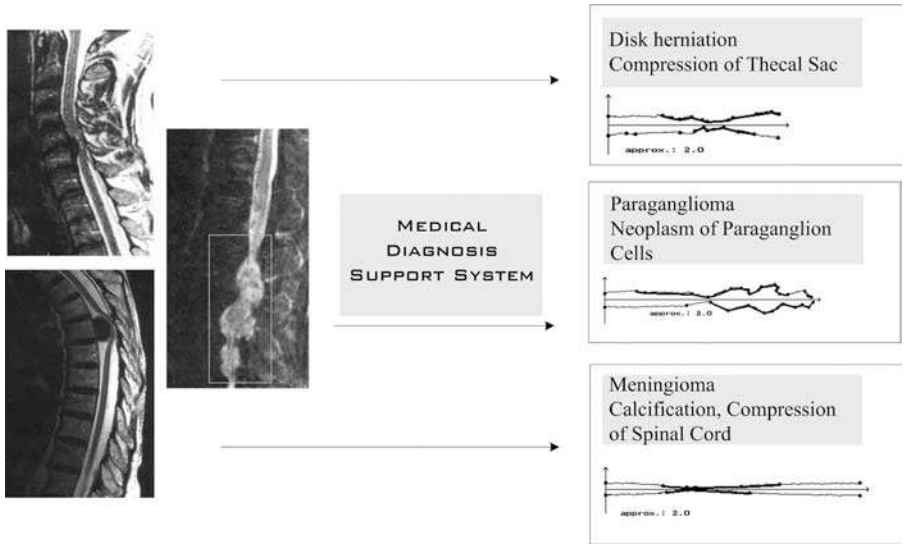



**Figure 14.6** Results of disease symptom recognition and understanding in the images of the spinal cord.

## 8. Conclusions

The presented methods of structural cognitive analysis are basically an attempt at automating the specifically human process of understanding the medical meaning of various organ shapes on a digital image; they are not only an attempt at simple recognition. In particular, a diagnosis may result from such an automatic understanding of shape; it is also possible to draw other numerous medical conclusions. This information may supply a method of treatment, that is, different types of therapy may be recommended depending on the shape and pathological localization described in the grammar. The results obtained from the application of the characterized methods confirm the immense potential of syntactical methods in the diagnosis of cardiac ischemic diseases, urinary tract disabilities and inflammation and neoplasm processes in the pancreas, as well as lesions of the central nervous system.

The syntactic methods of pattern recognition presented in this chapter have many applications in the field of artificial intelligence and medical IT, especially in the fields of computer medical imaging

and computer-aided diagnosis. The methods, originating from mathematical linguistics, allow us not only to diagnose and create formal and advanced descriptions for complicated shapes of disease symptoms carrying diagnostic information. They can also be used to create intelligent computer systems constructed for the purpose of image perception: allowing us to obtain a definition and machine-interpretation of the semantic contents of the examined image. These systems may assist the operation of medical robots widely used in the operational field in various surgeries. They can also constitute an integral part of CAD systems or intelligent information systems managing pictorial medical databases located (scattered) in various places [36–38].

## References


[1] Albus, J. S. and Meystal, A. M. and Aleksander, M. *Engineering of Mind: An Introduction to the Science of Intelligent Systems*, John Wiley & Sons, Inc., New York, 2001.

[2] Bankman, I. (Ed.) *Handbook of Medical Imaging: Processing and Analysis*, Academic Press, 2002.

[3] Davis, L. S. (Ed.) *Foundations of Image Understanding*, Kluwer Academic Publishers, Norwell, 2001.

[4] Duda, R. O., Hart, P. E. and Stork, D. G. *Pattern Classification*, 2nd edition, John Wiley & Sons, Inc., New York, 2001.

[5] Leondes, C. T. (Ed.) *Image Processing and Pattern Recognition*, Academic Press, San Diego, 1998.

[6] Ogiela, M. R. and Tadeusiewicz, R. "Image Understanding Methods in Biomedical Informatics and Digital Imaging," *Journal of Biomedical Informatics*, **34**(6), pp. 377–386, 2001.

[7] Khan, M. G. *Heart Disease Diagnosis and Therapy*, Williams & Wilkins, Baltimore, 1996.

[8] Ogiela, M. R. and Tadeusiewicz, R. "Syntactic reasoning and pattern recognition for analysis of coronary artery images," *Artificial Intelligence in Medicine*, **26**, pp. 145–159, 2002.

[9] Mandal, A. K. and Jennette, J. Ch. (Eds) *Diagnosis and Management of Renal Disease and Hypertension*, Carolina Academic Press, 1994.

[10] Silvus, S. E., Rohrmann, Ch. A. and Ansel, H. J. *Text and Atlas of Endoscopic Retrograde Cholangio-pancreatography*. Igaku-Shain, New York, 1995.

[11] Tadeusiewicz, R. and Ogiela, M. R. "Artificial Intelligence Techniques in Retrieval of Visual Data Semantic Information," in Menasalvas, E., Segovia, J., Szczepaniak, P. S. (Eds), *Advances in Web Intelligence, Lecture Notes in Artificial Intelligence*, 2663, Springer Verlag, pp. 18–27, 2003.

[12] Skomorowski, M. "Use of random graphs for scene analysis," *Machine Graphics & Vision*, **7**, pp. 313–323, 1998.

[13] Tadeusiewicz, R. and Flasinski, M. *Pattern Recognition*, Polish Scientific Publisher, Warsaw, 1991.

[14] Ogiela, M. R. and Tadeusiewicz, R. "Nonlinear Processing and Semantic Content Analysis in Medical Imaging," *Proceedings of IEEE International Symposium on Intelligent Signal Processing*, Budapest, pp. 243–247, 2003.

[15] Ogiela, M. R. and Tadeusiewicz, R. "Advanced image understanding and pattern analysis methods in Medical Imaging," *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing (SIP 2002)*, Kaua'i, Hawaii, USA, pp. 583–588, 2002.

[16] Ogiela, M. R., Tadeusiewicz, R. and Ogiela, L. "Syntactic Pattern Analysis in Visual Signal Processing and Image Understanding," *The International Conference on Fundamentals of Electronic Communications and Computer Science–ICFS 2002*, Tokyo, Japan, pp. 13:10 – 13:14, 2002.

[17] Meyer-Baese, A. *Pattern Recognition in Medical Imaging*, Elsevier–Academic Press, 2004.

[18] Ogiela, M. R. and Tadeusiewicz, R. "Visual Signal Processing and Image Understanding in Biomedical Systems," *Proceedings of the 2003 IEEE International Symposium on Circuits and Systems*, **5**, pp. V-17–V-20, 2003.

[19] Leś, Z., Tadeusiewicz, R. and Leś, M. "Shape Understanding: Knowledge Generation and Learning," *Proceedings of the Seventh Australian and New Zealand Intelligent Information Systems Conference (ANZIIS 2001)*, Perth, Western Australia, pp. 189–195, 2001.

[20] Tadeusiewicz, R. and Ogiela, M. R. "Automatic Understanding Of Medical Images–New Achievements In Syntactic Analysis Of Selected Medical Images," *Biocybernetics and Biomedical Engineering*, **22**(4), pp. 17–29, 2002.

[21] Ogiela, M. R. and Tadeusiewicz, R. "Artificial Intelligence Structural Imaging Techniques in Visual Pattern Analysis and Medical Data Understanding," *Pattern Recognition*, **36**(10), pp. 2441–2452, 2003.

[22] Sonka, M. and Fitzpatrick, J. M. (Eds) *Handbook of Medical Imaging: Vol. 2–Medical Image Processing and Analysis*. SPIE PRESS, 2000.
[23] Pavlidis, T. *Algorithms for Graphics and Image Processing*, Computer Science Press, Rockville, 1982.
[24] Bertrand, G., Everat, J.-Ch. and Couprie, M. "Image segmentation through operators based on topology," *Journal of Electronic Imaging*, **6**(4), 395–405, 1997.
[25] Mikrut, Z. "A Method of Linear Star-Sections Applied for Object Separation in ERCP Images," *Proceedings of International Conference on Image Processing*, Lausanne, pp. 363–366, 1996.
[26] Ogiela, M. R and Tadeusiewicz, R. "Syntactic pattern recognition for X-ray diagnosis of pancreatic cancer," *IEEE Engineering In Medicine and Biology Magazine*, **19**(6), pp. 94–105, 2000.
[27] Ogiela, M. R. and Tadeusiewicz, R. "Syntactic Analysis and Languages of Shape Feature Description in Computer Aided Diagnosis and Recognition of Cancerous and Inflammatory Lesions of Organs in Selected X-Ray Images," *Journal of Digital Imaging*, **12**(2), Suppl 1, pp. 24–27, 1999.
[28] Ogiela, M. R. and Tadeusiewicz, R. "Advances in syntactic imaging techniques for perception of medical images," *The Imaging Science Journal*, **49**(2), pp. 113–120, 2001.
[29] Tadeusiewicz, R. and Ogiela, M. R. "Examples of the Application of New Approach to Structural Analysis of Medical Images," *Archive of Theoretical and Applied Computer Science*, **13**(4), pp. 311–327, 2001.
[30] Miclet, L. *Structural Methods in Pattern Recognition*, Springer Verlag, 1986.
[31] Ogiela, M. R. and Tadeusiewicz, R. "New Aspects of Using the Structural Graph-Grammar Based Techniques for Recognition of Selected Medical Images," *Journal of Digital Imaging*, **14**(2), Suppl 1, pp. 231–232, 2001.
[32] Sklansky, J. and Gonzales, V. "Fast Polygonal Approximation of Digitized Curves," *Pattern Recognition*, **12**, pp. 327–331, 1980.
[33] Levine, J., Mason, T. and Brown D. *LEX & YACC*, O'Reilly & Associates, Inc., 1992.
[34] Netter, F. H. and Colacino S. *Atlas of Human Anatomy*, Novartis Medical Education, 1998.
[35] Burgener, F. A., Meyers, S. P. and Tan, R. *Differential Diagnosis in Magnetic Resonance Imaging*, Thieme, Stuttgart, 2001.
[36] Berchtold, S., Keim, D. A., Kriegel, H-P. and Seidl, T. "Indexing the solution space: a new technique for nearest neighbor search in high-dimensional space," *IEEE Transactions on Knowledge & Data Engineering*, **12**(1), pp. 45–57, 2000.
[37] Martinez, A. M. and Serra J. R. "A new approach to object-related image retrieval," *Journal of Visual Languages & Computing*, **11**(3), pp. 345–363, 2000.
[38] Ogiela, M. R. and Tadeusiewicz, R. "Semantic-Oriented Syntactic Algorithms for Content Recognition and Understanding of Images in Medical Databases," *2001 IEEE International Conference on Multimedia and Expo – ICME 2001*, Tokyo, Japan, pp. 621–624, 2001.

# 15

# The Roadmap for Recognizing Regions of Interest in Medical Images

**Sabah M.A. Mohammed**
**Jinan A.W. Fiaidhi**
**Lei Yang**
Department of Computer Science, Lakehead University, Ontario, Canada

*Segmentation of medical imagery is a challenging problem due to the complexity of the images, as well as to the absence of models of the anatomy that fully capture the possible deformations in each structure. Simple, low-level vision techniques have had limited success in this domain. This chapter provides the roadmap for automatically and effectively segmenting regions of interest. The roadmap starts by identifying three basic operations for image segmentation: convolution, thresholding and mathematical morphology. Based on these operations, one can develop a variety of hybrid region identification techniques which we call 'Visualization Operators'. The final step of the roadmap is to add intelligent capabilities to the recognition of these regions, based on techniques like fuzzy logic or neural networks.*

## 1. Introduction

Image segmentation is a partitioning of an image into related regions. The segmentation process is perhaps the most important step in image analysis since its performance directly affects the performance of the subsequent processing steps in image analysis. Despite its importance, segmentation still remains an unsolved problem in the general sense, as it lacks a general mathematical theory. The two main difficulties of the segmentation problem are its underconstrained nature [1] and the lack of definition of the 'correct' segmentation [2]. Perhaps as a consequence of these shortcomings, a plethora of

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

segmentation algorithms has been proposed in the literature [3]. There is a variety of such segmentation methods:

- histogram thresholding;
- edge following;
- tree/graph-based segmentation;
- region growing;
- clustering;
- probabilistic and Bayesian segmentation;
- neural network segmentation.

One major problem present in most of the above segmentation techniques is that they do not perform well with medical images, as their gray levels of different objects are relatively similar. The other major problem is choosing a suitable approach for isolating different objects from the backgrounds. The combination of some of the above techniques may produce better segmentation results for some images, but nothing is guaranteed.

In contrast to the heuristic nature of these methods, three primitive segmentation operations remain intrinsic and suggest a more algorithmic tack if they have been tuned and geared well for medical images. These three primitive operations are based on convolution, thresholding and mathematical morphology. Hybridizing such operations and adding more blocks to the integration will contribute to drawing the roadmap for segmenting medical images.

## 2. Convolutional Primitive Segmentation

Convolution is a mathematical operation that is fundamental to many common image processing processes. Convolution provides a mechanism for edge detection through 'multiplying together' two arrays of numbers to produce a third array of numbers of the same dimensionality. This can be used in image processing to implement operators whose output pixel values are simple linear combinations of certain input pixel values. In an image processing context, one of the input arrays is normally an image. The second array is usually much smaller, and is also two-dimensional, and is known as the *kernel*. Figure 15.1 shows an example image and a kernel that we will use to illustrate the convolution operation.

The convolution is performed by sliding the kernel over the image, generally starting at the top left corner, so as to move the kernel through all the positions where the kernel fits entirely within the boundaries of the image. Each kernel position corresponds to a single output pixel, the value of which is calculated by multiplying together the kernel value and the underlying image pixel value for each

| | | |
|---|---|---|
| $K_{00}$ | $K_{01}$ | $K_{02}$ |
| $K_{10}$ | $K_{11}$ | $K_{12}$ |
| $K_{20}$ | $K_{21}$ | $K_{22}$ |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $I_{00}$ | $I_{01}$ | $I_{02}$ | $I_{03}$ | $I_{04}$ | $I_{05}$ | $I_{06}$ | $I_{07}$ | $I_{08}$ |
| $I_{10}$ | $I_{11}$ | $I_{12}$ | $I_{13}$ | $I_{14}$ | $I_{15}$ | $I_{16}$ | $I_{17}$ | $I_{18}$ |
| $I_{20}$ | $I_{21}$ | $I_{22}$ | $I_{23}$ | $I_{24}$ | $I_{25}$ | $I_{26}$ | $I_{27}$ | $I_{28}$ |
| $I_{30}$ | $I_{31}$ | $I_{32}$ | $I_{33}$ | $I_{34}$ | $I_{35}$ | $I_{36}$ | $I_{37}$ | $I_{38}$ |
| $I_{40}$ | $I_{41}$ | $I_{42}$ | $I_{43}$ | $I_{44}$ | $I_{45}$ | $I_{46}$ | $I_{47}$ | $I_{48}$ |
| $I_{50}$ | $I_{51}$ | $I_{52}$ | $I_{53}$ | $I_{54}$ | $I_{55}$ | $I_{56}$ | $I_{57}$ | $I_{58}$ |
| $I_{60}$ | $I_{61}$ | $I_{62}$ | $I_{63}$ | $I_{64}$ | $I_{65}$ | $I_{66}$ | $I_{67}$ | $I_{68}$ |
| $I_{70}$ | $I_{71}$ | $I_{72}$ | $I_{73}$ | $I_{74}$ | $I_{75}$ | $I_{76}$ | $I_{77}$ | $I_{78}$ |

**Figure 15.1** The convolution process requires an image and a kernel.

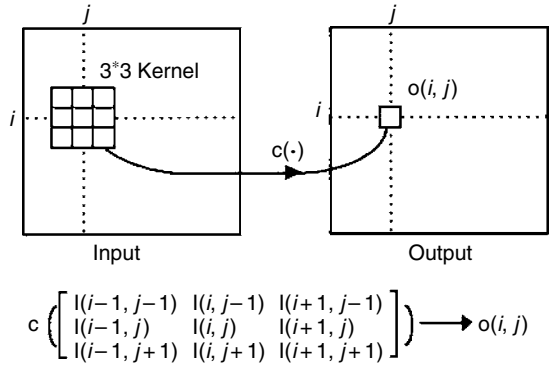



**Figure 15.2** Illustrating the process of convolution.

of the cells in the kernel, and then adding all these numbers together [4]. Figure 15.2 illustrates this operation. Using convolution, the value of the output pixel $O_{22}$ can be calculated as follows:

$$O_{22} = I_{11}{}^{*}K_{00} + I_{12}{}^{*}K_{01} + I_{13}{}^{*}K_{02} + I_{21}{}^{*}K_{10} + I_{22}{}^{*}K_{11} + I_{23}{}^{*}K_{12} + I_{32}{}^{*}K_{20} + I_{32}{}^{*}K_{21} + I_{33}{}^{*}K_{22} \quad (15.1)$$

The convolution process is a highly researched issue and it is affected by many factors. There are many types of convolution operator used in the literature [5] which either use a single kernel or multiple kernels. There are many single kernels that are only used for special effects (e.g. horizontal line detection, smoothing). Figure 15.3 illustrates the effect of two single kernels upon an X-ray image using smoothing and high-pass kernels:

$$\frac{1}{16} * \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \quad \text{(Smoothing)} \qquad \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 9 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \quad \text{(High-pass)}$$


**Figure 15.3** Using a single convolution kernel for smoothing.

However, multiple kernels are applied on the image in multiple stages. In particular, there are some notable multiple kernel convolution operators that we list herewith.

The *Roberts Cross* operator performs a simple, quick to compute, 2D spatial gradient measurement on an image. The operator consists of a pair of $2 \times 2$ convolution kernels. One kernel is the other rotated by 90°.

$$G_x: \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array} \quad G_y: \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array} \qquad |G| = |G_x| + |G_y|$$

These kernels are designed to respond maximally to edges running at 45° to the pixel grid, one kernel for each of the two perpendicular orientations. We can see that the result image is very dark when applying the Roberts Cross operator to a mammogram.

The *Laplacian* is a 2D isotropic measure of the second spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection.

$$\begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \quad \text{and} \quad \begin{array}{|c|c|c|}\hline -1 & -4 & -1 \\ \hline -4 & 20 & -4 \\ \hline -1 & -4 & -1 \\ \hline \end{array}$$

The *Sobel* operator performs a 2D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. Typically, it is used to find the approximate absolute gradient magnitude at each point in an input image. Sobel kernels are:

$$\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \quad \text{and} \quad \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these $G_x$ and $G_y$). These can then be combined together to find the absolute magnitude of the gradient at each point $|G| = \sqrt{G_x^2 - G_y^2}$ and the orientation of that gradient. An approximate magnitude can be computed using: $|G| = |G_x| + |G_y|$.

*Compass edge detection* is an alternative approach to the differential gradient edge detection. When using compass edge detection, the image is convoluted with a set of (in general eight) convolution kernels, each of which is sensitive to edges in a different orientation. For each pixel, the local edge gradient magnitude is estimated with the maximum response of all eight kernels at this pixel location: $|G| = \max(|G_i| : i = 1 \text{ to } n)$ where $G_i$ is the response of the kernel $i$ at the particular pixel position and $n$ is the number of convolution kernels. The local edge orientation is estimated with the orientation of the kernel that yields the maximum response. Various kernels can be used for this purpose, and best of all are the Prewitt and the Kirsch kernels. Two templates out of the set of eight are:

$$0°: \begin{array}{|c|c|c|}\hline -1 & 1 & 1 \\ \hline -1 & -2 & 1 \\ \hline -1 & 1 & 1 \\ \hline \end{array} \qquad 45°: \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline -1 & -2 & 1 \\ \hline -1 & -2 & 1 \\ \hline \end{array}$$

The *Kirsch* edge detector [6] is another compass kernel. The masks given by these templates try to model the kind of gray-level change seen near an edge having various orientations. There is a mask for each of eight compass directions. For instance, K0 implies a vertical edge (horizontal gradient) at the pixel corresponding at the center of the mask. To find the edge, I is convolved with the eight masks at each pixel position. The response is the maximum of the responses of any of the eight masks and the directions quantified into eight possibilities.

K0 :

| | | |
|---|---|---|
| −3 | −3 | 5 |
| −3 | 0 | 5 |
| −3 | −3 | 5 |

K1 :

| | | |
|---|---|---|
| −3 | 5 | 5 |
| −3 | 0 | 5 |
| −3 | −3 | −3 |

The *Canny* edge detection algorithm [7] is known to many as the optimal edge detector. The Canny edge detector first smoothes the image to eliminate noise. It then finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (nonmaximum suppression). The gradient array is now further reduced by hysteresis. Hysteresis is used to track along the remaining pixels that have not been suppressed. Hysteresis uses two thresholds and if the magnitude is below the first threshold, it is set to zero (made a nonedge). If the magnitude is above the high threshold, it is made an edge. And if the magnitude is between the two thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above T2. Figure 15.4 illustrates the effects of various multiple kernel convolutions.

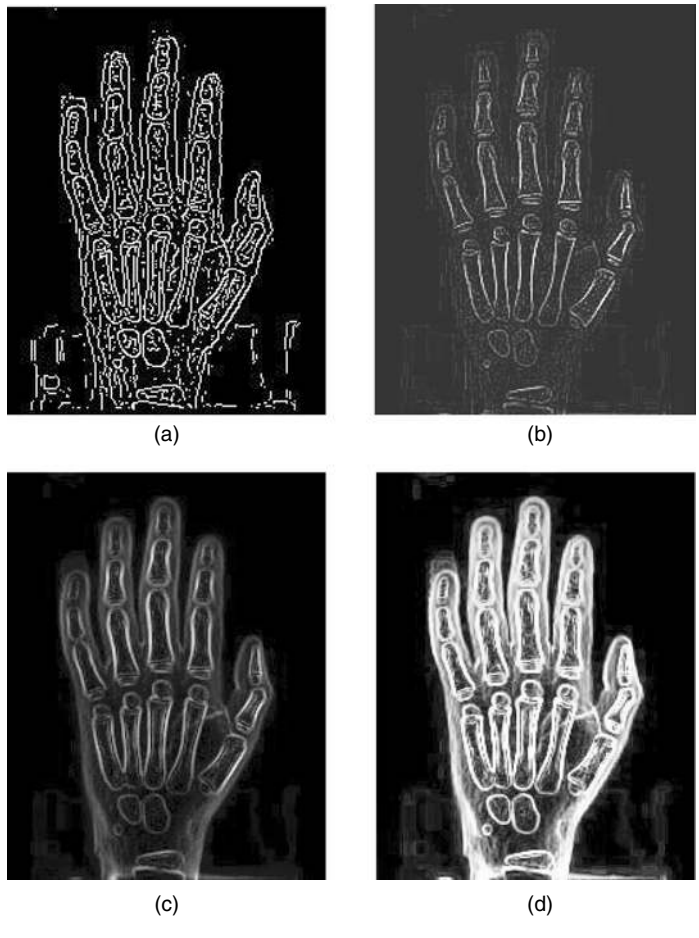



**Figure 15.4** Results of other multiconvolution kernels. (a) Canny with sigma = 0.8; (b) Laplacian with brightness enhancement; (c) Prewitt; (d) Kirsch.

## 3. Thresholding Primitive Segmentation

Thresholding makes color changes across a programmer-determined 'boundary', or threshold, more obvious. This technique uses a specified threshold value, minimum value and maximum value to control the color component values for each pixel of an image. Color values below the threshold are assigned the minimum value. Values above the threshold are assigned the maximum value. The threshold process is performed for each color component of each pixel. When the operation is complete, the color components of the destination image pixels will contain either the minimum value or the maximum value. For example, consider what happens to an image when a threshold operation is performed with a minimum of 0 and a maximum of 255. After the image is processed, the red, green and blue values of the pixels will be either 0 or 255. In the following examples, we write the threshold value in (min, threshold, max) format. We can see that the white area (indicating a higher brightness area) gets smaller as the threshold value goes up. Thresholding is a very important morphological technique for detecting tumor regions because a tumor represents a dense tissue area of, for example, the breast structure. However, it is very tricky to find the right threshold value that can work for a variety of mammograms. For this purpose, many researchers tried to arrive at a dynamic threshold which can learn its optimal value from the type of image under analysis. Figure 15.5 illustrates the effects of using a variety of static thresholds.

## 4. Morphological Primitive Segmentation

Morphological operators often take a binary image and a structuring element as input and combine them using a set operator (intersection, union, inclusion, complement). For the basic morphological operators, the structuring element contains only foreground pixels (i.e. ones) and 'don't cares'. These operators, which are all a combination of erosion and dilation, are often used to select or suppress features of a certain shape, e.g. removing noise from images, skeletonization-thinning or selecting objects with a particular direction. Morphological operators can also be applied to gray-level images, e.g. to reduce noise or to brighten the image. The method is to treat the image as a sequence of binary images by operating on each gray level as if it were the 1 value and assuming everything else to be 0. The resulting images can then be combined by laying them on top of each other and 'promoting' each pixel to the highest gray-level value coincident with that location. Mathematical morphology provides a number of important image processing operations, including erosion, dilation, opening and closing.

### *4.1 Erosion*

The basic effect of the operator on a binary image is to erode away the boundaries of regions of foreground pixels (i.e. white pixels, typically). Thus, areas of foreground pixels shrink in size and holes within those areas become larger. To compute the erosion of a binary input image by this structuring

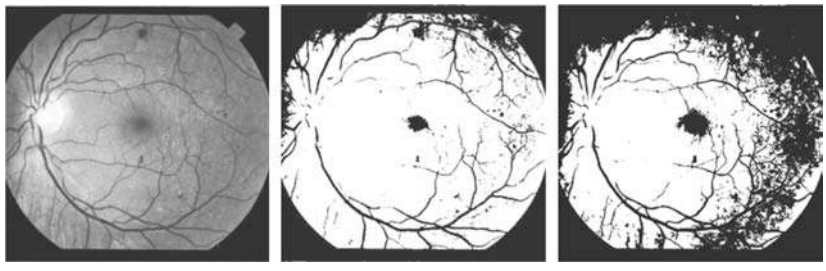



**Figure 15.5** Effects of some static thresholds. (a) The original image; (b) threshold 80; (c) threshold 100.

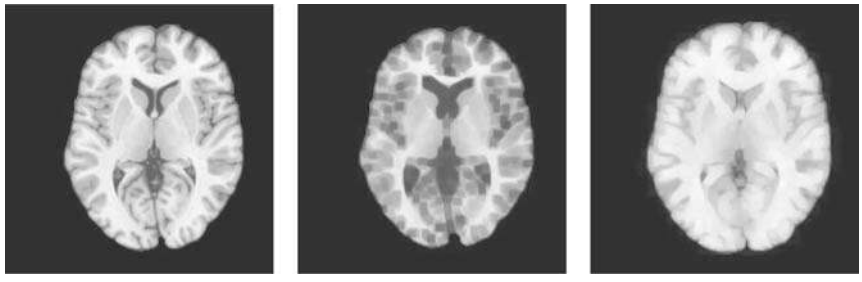



**Figure 15.6** The effects of the two morphological operations on an X-ray. (a) The original image; (b) erosion two times; (c) dilation two times.

element, we consider each of the foreground pixels in the input image in turn. For each foreground pixel (which we will call the input pixel), we superimpose the structuring element on top of the input image so that the origin of the structuring element coincides with the input pixel coordinates. If, for every pixel in the structuring element, the corresponding pixel in the image underneath is a foreground pixel, then the input pixel is left as it is. If any of the corresponding pixels in the image are background however, the input pixel is also set to the background value.

### *4.2 Dilation*

The basic effect of a dilation operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels. Thus, areas of foreground pixels grow in size while holes within those regions become smaller.

### *4.3 Opening and Closing*

These are both derived from the fundamental operations of erosion and dilation. The basic effect of an opening is like erosion, in that it tends to remove some of the foreground (bright) pixels from the edges of regions of foreground pixels. However, it is less destructive than erosion in general. As with other morphological operators, the exact operation is determined by a structuring element. The effect of the operator is to preserve foreground regions that have a similar shape to this structuring element, or that can completely contain the structuring element, while eliminating all other regions of foreground pixels. Closing is similar in some ways to dilation, in that it tends to enlarge the boundaries of foreground (bright) regions in an image (and shrink background color holes in such regions), but it is less destructive of the original boundary shape. Figure 15.6 illustrates the effects of applying repetitive erosion and dilation operators on mammograms.

## 5. Hybridizing the Primitive Segmentation Operators

Usually, hybrids of the basic segmentation primitives are used for static image rendering or what is known as volume visualization [8]. These techniques are used in medicine, geoscience, astrophysics, chemistry, microscopy, mechanical engineering and other areas. Visualization is usually achieved using either volume or surface rendering techniques. In this section we are proposing certain pipelines for volume visualization. Such visualization pipelines are based on the proper choice of various primitive segmentation operators in order to map the original image data into meaningful optical properties of the volume to be visualized. The visualization operators may use other operations that affect rendering

(e.g. add, subtract, invert and overlay). The visualization pipelines can be simple when the rendered image is composed of using only binary operations, such as erosion and inversion (see Figure 15.7).

More complex visualization pipelines can be produced when edge detection operators are involved. Figure 15.8 illustrates our first experiment when we used a thresholding value of 210 to segment the image into high-brightness and low-brightness areas.

We can use more than one threshold value to highlight the various volume elements in the image, as Figure 15.9 illustrates.

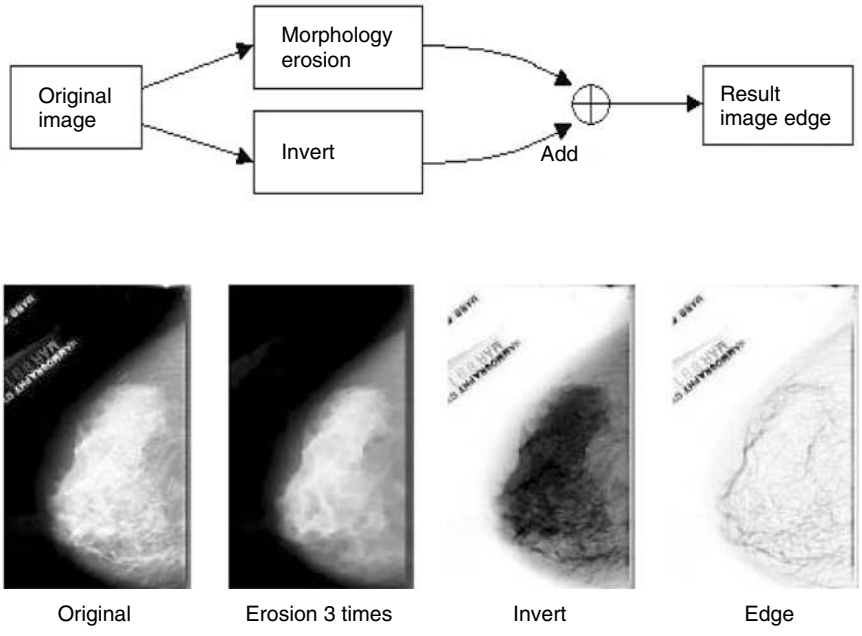



**Figure 15.7** A simple visualization pipeline.

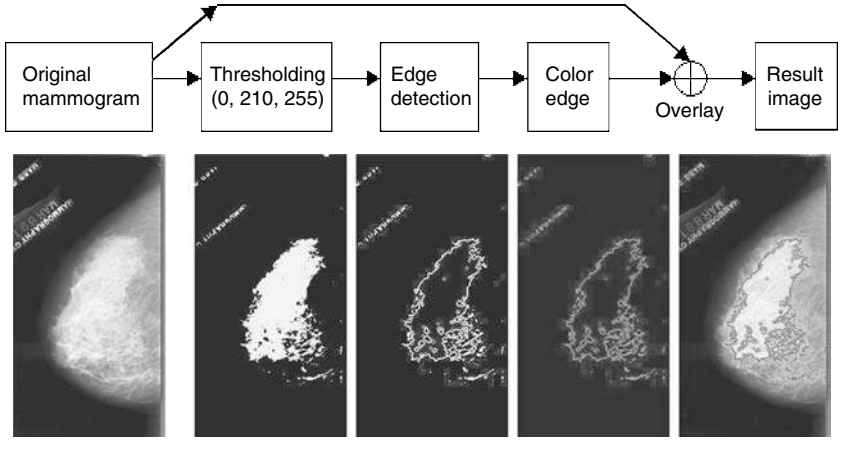
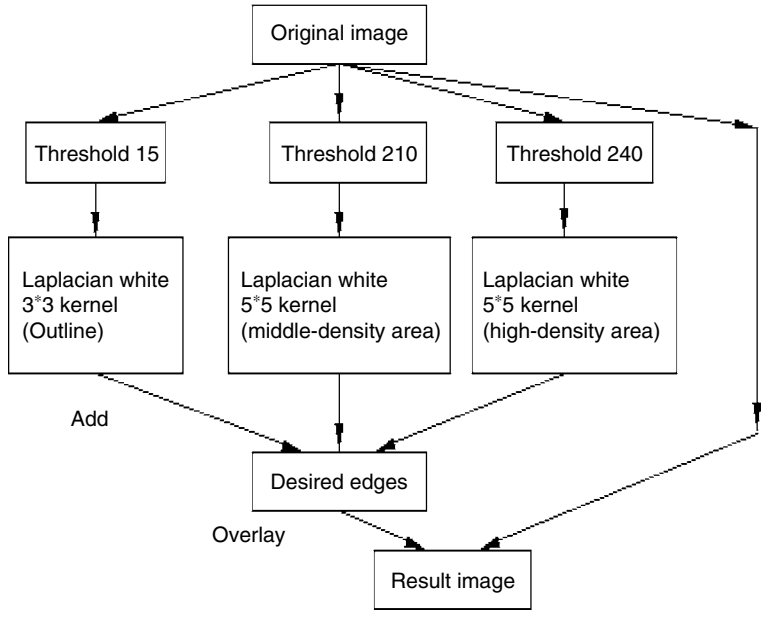



**Figure 15.8** Visualization pipeline aimed at separating the image into high- and low-brightness areas.



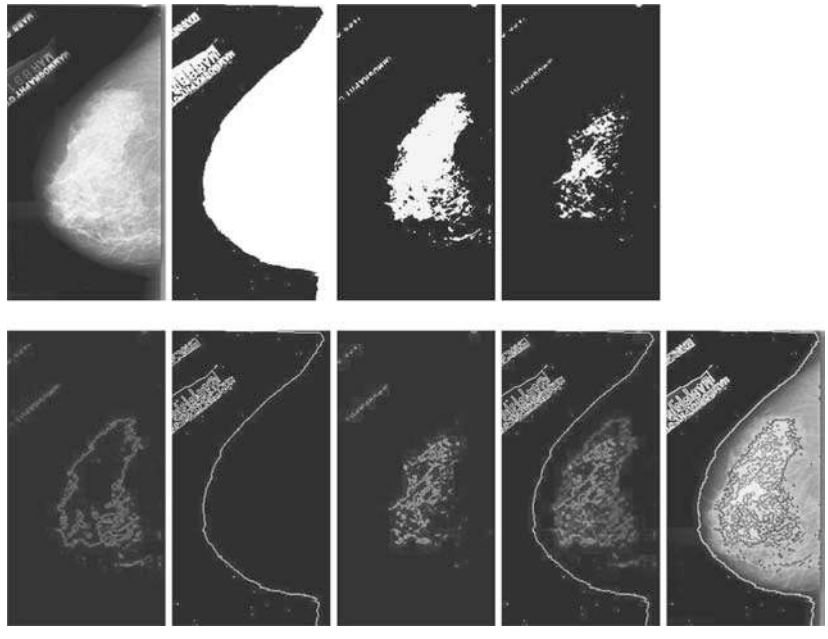



**Figure 15.9** Visualization pipelines for separating different lightness areas.

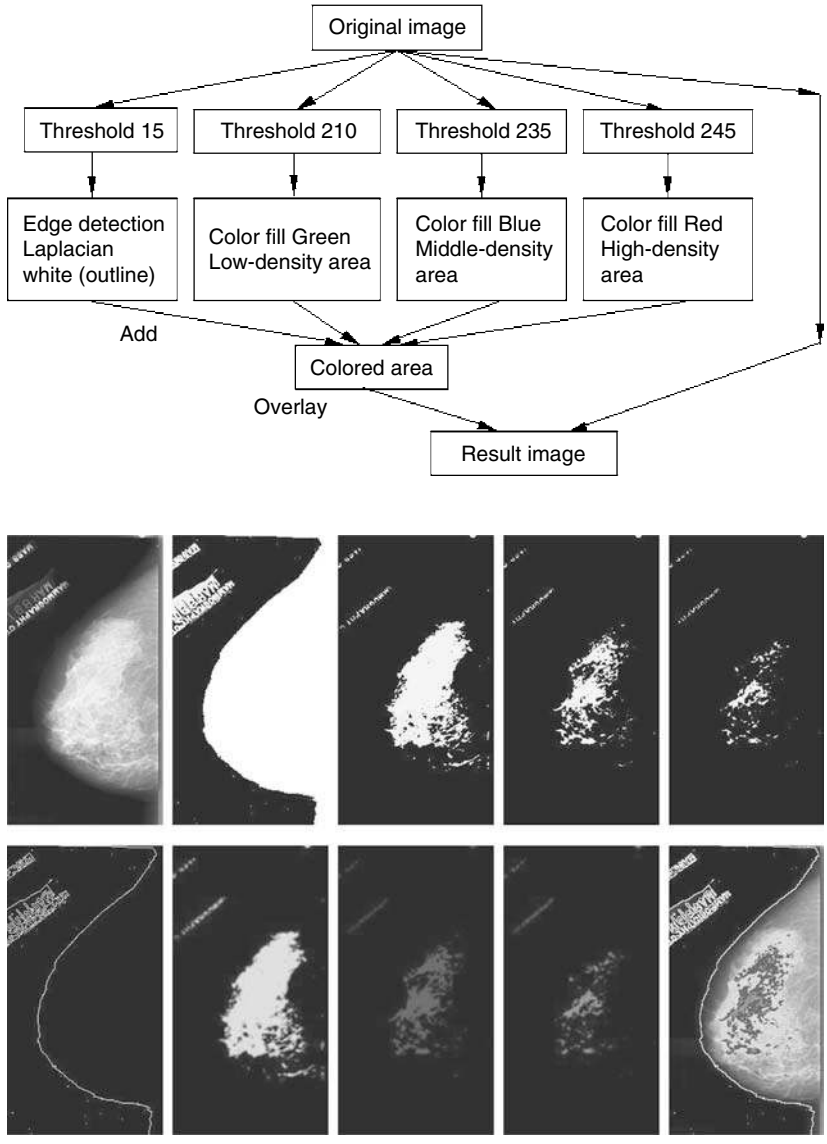



**Figure 15.10** Visualization pipeline with area filling rendering effects.

We can enhance the rendered areas by filling the segmented areas with different colors. Figure 15.10 illustrates the new visualization pipelines.

The visualization pipeline operation is a good tool for image processing experts to statically render given images, but it is difficult for health workers to use, especially when image style keeps changing. For this purpose, we dedicate the next section to developing a single effective technique that can cope with region identification, even if the image style keeps changing, and can be easily used by health workers. This new approach dynamically follows the direction of the region edges using some fuzzy membership rules. We call this new approach the Dynamic Fuzzy Classifier (DFC) [9].

## 6. Region Identification Based on Fuzzy Logic

Analysis of medical images usually involves nonlinear, noisy and imprecise data which requires methods that are not directly generated from the three primitive approaches described earlier in this chapter. In particular, the use of fuzzy techniques has proved successful in several medical applications such as mammography, since they take a more flexible approach to pattern recognition and edge detection [10]. Basically, there are two different possibilities for the development of fuzzy edge detectors or region identification:

1. Using fuzzy membership functions [9].
2. Using fuzzy identification rules (e.g. as in Figure 15.11) [11].

However, most membership functions appearing in the literature are determined heuristically, without any additional rules used to modify the membership values. Indeed, the use of rules can smooth while sharpening edges, but requires a rather large rule set compared to the simpler fuzzy membership classifier methods [12–16]. For this purpose, neural networks have been used to train the process of selecting the set of rules for detecting edges [17,18] and radial basis functional link nets [19] have been found to be especially powerful in this direction. Training such networks is not a straightforward process and many complications are associated with it [20,21]. Moreover, these techniques proved to have high success rates in detecting some types of cancer abnormality with systematic shape structures and in particular with calicification and circumscribed breast cancer types. However, stellate breast cancer lesions express vast architectural distortions and present with an appearance of fuzzy radiating lines. Attempts to automatically detect these abnormalities have generally concentrated on collecting some statistical features of known importance, such as textures which can be discriminated with low variances, radiating linear structure concurrency, spread of focus and radial distance. Hence, the authors believe that by developing a dynamically fuzzy membership classifier we can arrive at a simple pseudoadaptive edge detection algorithm which does not require training. This section describes this new approach in the context of a Dynamic Fuzzy Classifier (DFC); our experiments were conducted on mammograms (i.e. typical medical X-ray images).

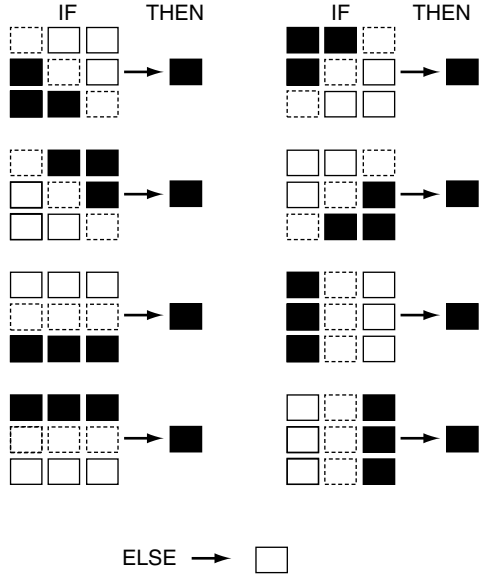



**Figure 15.11** Fuzzy rules that are used for region identification.

The DFC is intended to be a part of a soft tissue abnormality detection scheme. Design of the algorithm is based on two main properties of mammographic lesions and patterns. First, because of the unclear boundaries of the parenchyma and malignant masses in a mammogram, employing the principles of fuzzy sets in assigning the image pixels to different regions is appropriate. Secondly, since abnormal tissue lesions and masses are usually larger than a certain size, in determining which segmented region a pixel belongs to, the effects of its neighboring pixels, as well as its own intensity value, must be considered. The DFC method starts by moving a polygon mask on every pixel of the given image. Each pixel will be placed at the center of the polygon (Figure 15.12).

Based on the values of the brightness of the 13 pixels of the polygon, eight brightness direction features are extracted. These features are the magnitudes of the differences between the thirteen neighboring pixels. Using the values within the polygon, we can calculate the direction features by taking the different $2 \times 3$ orientations within the neighboring pixels within the polygon. For example, to calculate the East direction feature value, we need find the value of $P2 - P1 + P5 - P4 + P8 - P7$ (see Figure 15.13). For the Northeast direction feature, we need to find the value of $P9 - P0 + P2 - P4 + P11 - P8$ (see Figure 15.14). The eight feature direction values can be calculated in this way.

These eight direction features are the input to a fuzzy membership classifier that connects to two fuzzy set membership functions that represent the set of pixels which belong to the '0' (dark) region,

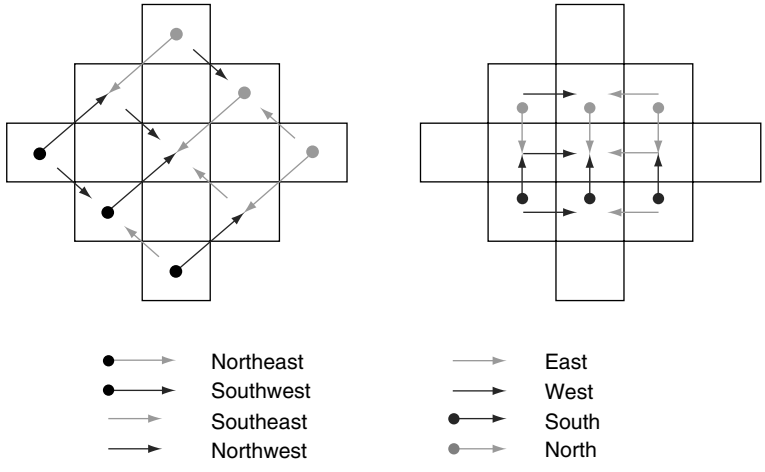



**Figure 15.12** Pixel direction polygon mask.

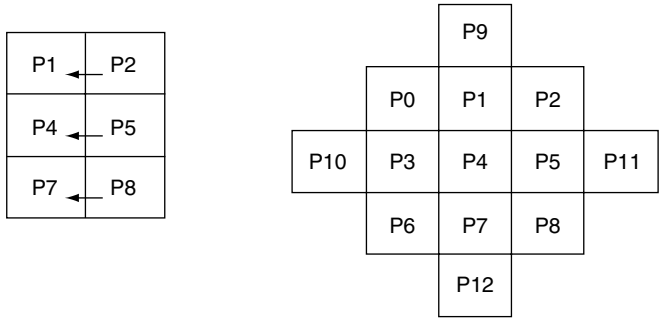



**Figure 15.13** Polygon reference map and the East direction feature computation.

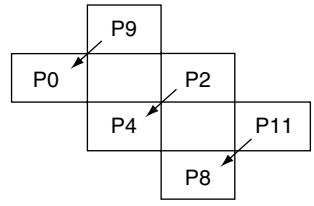



**Figure 15.14** Northeast direction feature computation.

and the set of pixels which belong to the '1' (bright) region. The membership value determines how close a pixel is to becoming a member of either set, as well as the brightness slope direction. From the eight direction features we choose the highest value which will indicate the slope direction.

At each pixel, the fuzziness measure of each pixel is dynamically updated to reduce the error function value. Inspired by the gradient method and the back-propagation algorithm for neural network training, three update rules are designed to both reduce the error in each pixel iteration and to consider the effects of brightness for all the neighboring pixels:

**Rule 1**: If the direction value is bigger than a brightness constant (e.g. 250), then set this pixel brightness (P4) to white, otherwise apply Rule 2.

**Rule 2**: If the direction value is greater than a mammogram threshold value, then apply Rule 3; otherwise set this pixel brightness (P4) to black.

**Rule 3**: To apply this rule, compute the following:

1. Choose $n$; an odd number of pixels along the line of the slope direction of this pixel (P4). This pixel should be in the center. The value of $n$ should not be greater than min(image width, image height).
2. Put the pixel values of this line into an array $\mathbf{A}[]$. We call this array a transition zone.
3. Calculate the average value of array $\mathbf{A}[]$ elements (i.e $A[0]+A[1]+\cdots+A[n])/n$).
4. Calculate the array accumulation value (i.e. $A[1]-A[0]+A[2]-A[0]+\cdots A[n]-A[0]$).

This rule states that if the average value equals the pixel value, and the accumulation value is greater than $n+$ (contrast constant), then set this pixel value (P4) to white; otherwise set this pixel value (P4) to black.

The proposed DFC method introduces the concept of fuzzy contrast that depends on how far the membership functions are stretched by an operator with respect to the middle point from data points within the transition zone being considered. The pixel orientation and slope of the transition zone provide the most important information, which are used for determining edge membership. Indeed the ideal case of an edge pixel is when this pixel is right at the middle of the slope of the transition zone (see Figure 15.15(a)). However, we cannot always find a pixel that is exactly in the middle (see Figure 15.15(b)). In order to get a better connectivity of detected edges, a thresholding value $b$ is used: $0.8 < b < 1.25$, where $b =$ midvalue/P4 brightness value, midvalue $= (A[0]+A[n-1])/2$. Rule 3 is responsible for identifying this case. Also, in many cases the brightness changes sharply and the middle pixel cannot be directly specified. We call such cases absolute steps. Rule 1 is designed to find edge pixels for absolute steps (see Figure 15.15(c)). An absolute pixel case is an edge if brightness increases sharply along the pixel direction.

The DFC method tries to intensify the contrast of those pixels which are almost at, or near to the actual edge. The fuzzy nature of the pixel brightness can be depicted from a 3D projection to the pixel brightness of an image segment (see Figure 15.16).

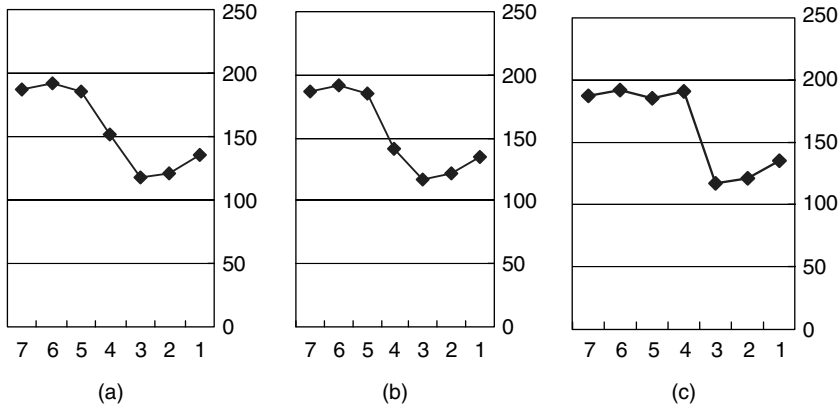



**Figure 15.15** Ideal and absolute edge membership. (a) Ideal; (b) not exactly in the middle; (c) absolute step.

| | | | | | | |
|---|---|---|---|---|---|---|
| 125 | 175 | 185 | 174 | 183 | 170 | 187 |
| 123 | 179 | 179 | 159 | 178 | 192 | 186 |
| 113 | 128 | 176 | 182 | 185 | 186 | 180 |
| 127 | 127 | 122 | 171 | 176 | 181 | 181 |
| 112 | 127 | 117 | 152 | 173 | 177 | 186 |
| 125 | 121 | 111 | 122 | 139 | 169 | 175 |
| 135 | 132 | 114 | 113 | 123 | 119 | 183 |

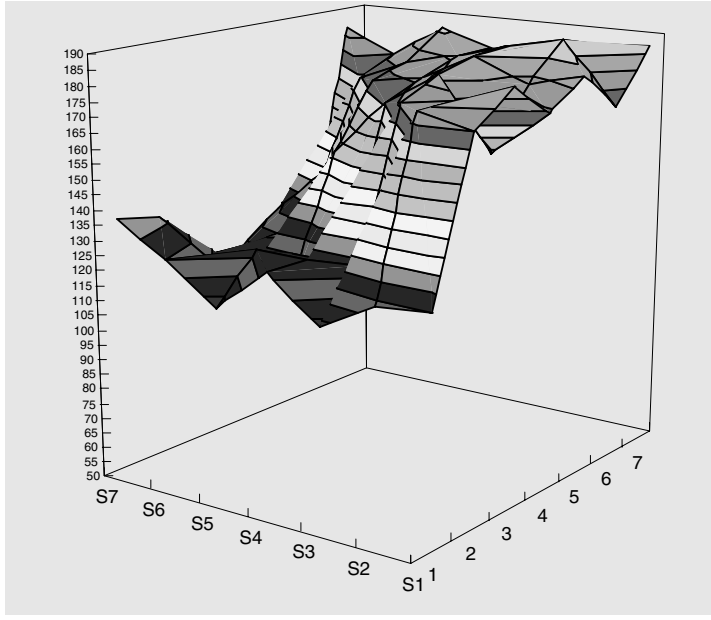



**Figure 15.16** 3D projection of pixel brightness.

## 6.1 Experimental Results

The data collection that was used in our experiments was taken from the Mammographic Image Analysis Society [22]. This same collection has been used in other studies of automatic mammography classification. Its corpus consists of 322 images, which belong to three big categories: normal, benign and malign. There are 208 normal images, 63 benign and 51 malign, which are considered abnormal. In addition, the abnormal cases are further divided into six categories: microcalcification, circumscribed masses, spiculated masses, ill-defined masses, architectural distortion and asymmetry. All the images also include the locations of any abnormalities that may be present. The existing data in the collection consists of the location of the abnormality (like the center of a circle surrounding the tumor), its radius, breast position (left or right), type of breast tissue (fatty, fatty glandular and dense) and tumor type if it exists (benign or malign). All the mammograms are medio-lateral oblique views. Figure 15.17 shows experiments conducted on abnormal breast cases from that database with a threshold value fixed at 12 and the number of transition elements equal to 25. Actually, we noted that with higher transition elements the edges become sharper and more obvious. However, selecting $n = 25$ resulted in rather more moderate results than among the other odd values that we may have chosen for $n$. The DFC results are produced without using any preprocessing to increase the contrast or to reduce the image noise. Figure 15.17 shows two other notable edge detection methods applied on the same original image.

We can furthermore reduce noise, as well as nonsignificant details, by applying a smooth filter before the DFC operation, see Figure 15.18.

Moreover, it is a well-known medical fact that dense breast tissues on mammograms indicate a higher cancer risk [31]. A dense region in any mammogram always has high brightness values. For this reason, we visualized low-brightness edges with different colors than those edges of high brightness. In this direction, we used *blue* to identify normal region edges (i.e. no possibility of having cancerous tissues, or 0 % density type in radiologist terminology). *Green* has been used to identify those edges with more dense regions indicating a higher possibility of having cancerous tissues (50–74 % density type). Finally, *White* is used to identify a very high possibility of having cancerous tissues (75–100 %

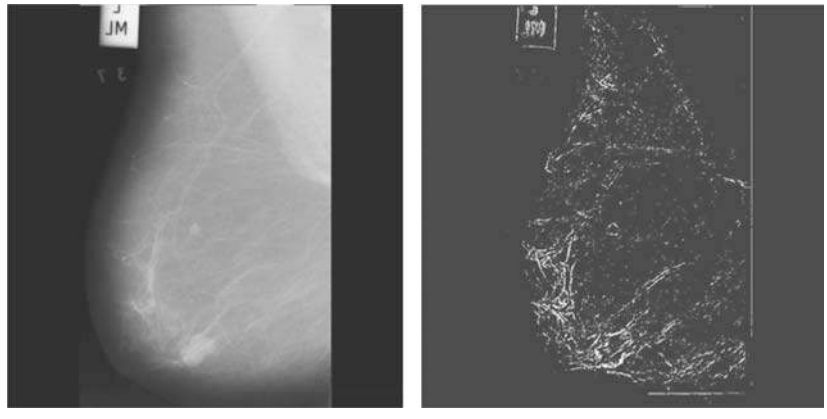



**Figure 15.17** DFC method applied to two mammograms. (a) The original image of Case #1, circumscribed cancer; (b) DFC threshold $= 12, n = 25$; (c) DFC with threshold $= 12, n = 25$ with coloring; (d) the original image of Case #2, stellate cancer; (e) DFC with threshold $= 12, n = 25$; (f) DFC with threshold $= 12, n = 25$ with coloring.

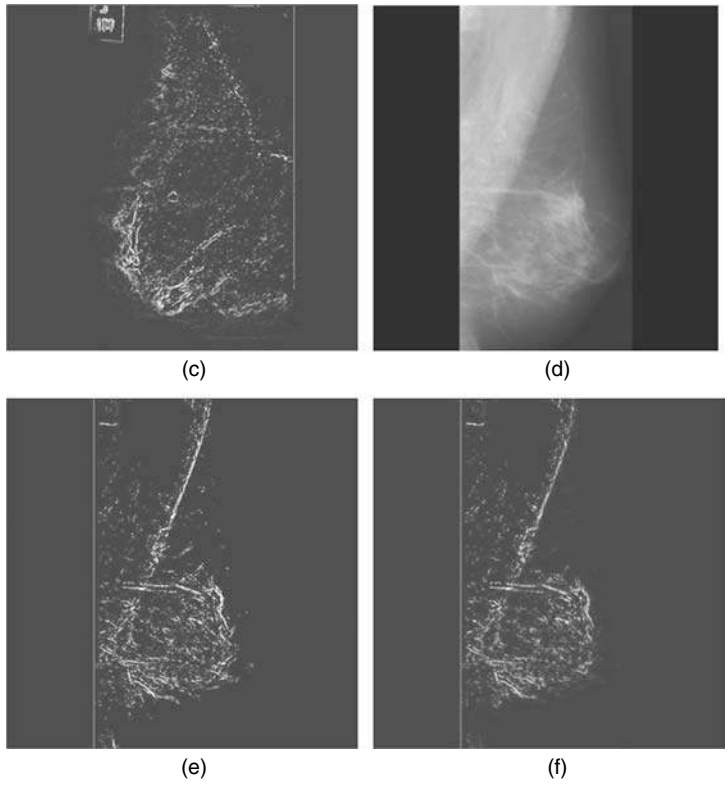



**Figure 15.17** (*continued*)

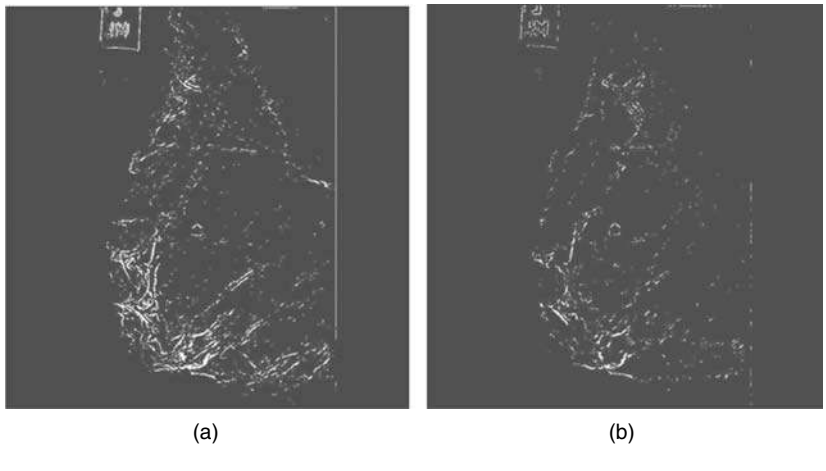
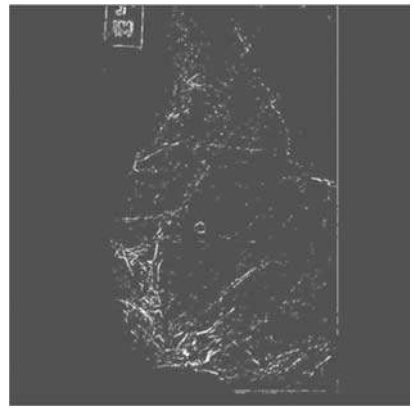



**Figure 15.18** Effect of the DFC method with smooth filters. (a) Applying median blur before DFC; (b) applying Gaussian blur before DFC; (c) applying a morphology erosion before DFC.



(c)

**Figure 15.18** (*continued*)

density type). We can consider this colorization process an additional rule which is based directly on the detected edge pixel brightness. Indeed, we tested DFC on all the normal cases of the database, none of them revealed specious structures with white or green colors. We are not claiming that the colorization scheme can be used to classify mammograms, but we can use it as one associative rule for mining abnormal mammograms. This issue is our current research program, in which we are trying to construct a data mining technique based on association rules extracted from our DFC method for categorizing mammograms. We are also intending, for the purpose of mammogram categorization, to use other measures besides our DFC association rules, like the brightness, mean, variance, skewness and kurtosis for the DFC segmented image. These measures have been reported to have some success in identifying abnormal mammograms [32]. Moreover, we experimented with all the edge detection techniques used in Table 15.1 and proved that no single method can be as effective as our current DFC method [33].

Figure 15.19 illustrates some comparisons for the stellate cancer image of this article.

**Table 15.1** Traditional mammography techniques.

| Mammography method | Reference |
|---|---|
| Gray-level thresholding | [23] |
| Comparing left and right breasts | [24] |
| Compass filters | [25] |
| Laplacian transform | [11] |
| Gaussian filters | [12] |
| Texture and fractal texture model | [17,19] |
| A space scale approach | [26] |
| Wavelet techniques | [27] |
| Mathematical morphology | [28] |
| Median filtering | [29] |
| Box-rim method | [30] |

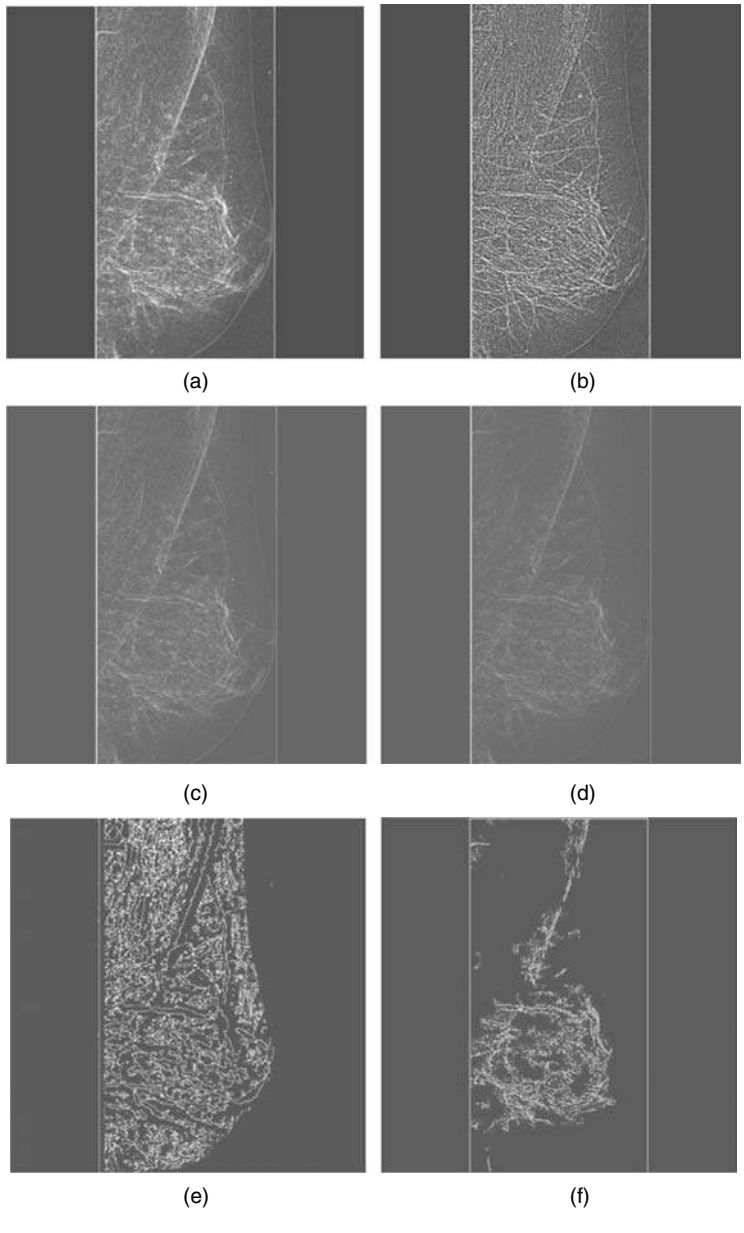
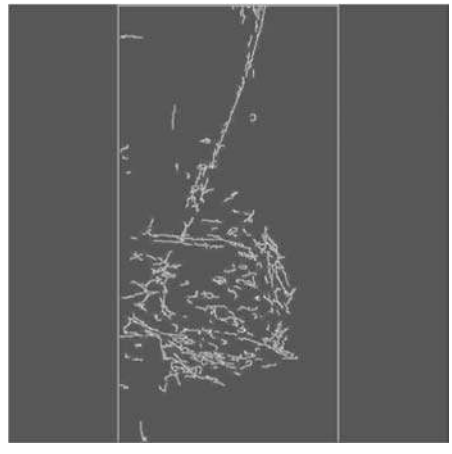



**Figure 15.19** Results of other traditional edge detection techniques. (a) The kirsch edge detection technique; (b) the Laplacian edge detection technique; (c) Sobel edge detection with contrast enhancement; (d) Prewitt edge detection with contrast enhancement; (e) Cafforio edge detection with inner window $= 3$ and outer window $= 21$; (f) Canny edge detection, $\delta = 0.045$; (g) Canny edge detection, $\delta = 0.065$.



(g)

**Figure 15.19** (*continued*)

## 7. Conclusions

In this chapter we identified three primitive routes for region identification based on convolution, thresholding and morphology. Experiments on medical images show that none of these routes is capable of clearly identifying the region edges. A better recognition can be achieved by hybridizing the primitive techniques through a pipelining sequence. Although many region identification pipelines can be found that enable us to clearly identify regions of interest, such a hybridizing technique remains valid when the characteristic images remain static. The problem with most of the medical images is that their characteristics vary so much, even for one type of imaging device. With this in mind, we are proposing a new fuzzy membership function that transforms the intensity of a pixel into the fuzzy domain, according to the direction of the brightness slope in its neighboring transition zone. A new intensification operator based on a polygon is introduced for determining the corrected intensity value for any pixel. The membership fuzzification classifier dynamically evaluates every pixel's brightness by optimizing its contrast according to the neighboring pixels. The method needs no preprocessing or training and does not change the brightness nature of the segmented image compared to the original image. The DFC method has been tested on a medical mammography database and has been shown to be effective for detecting abnormal breast regions. In comparisons with the traditional edge detection techniques, our current DFC method shows significant abnormality details, where many other methods (e.g. Kirsch, Laplacian) revealed irrelevant edges as well as extra noise. For Sobel and Prewitt, the original image becomes completely black. With contrast enhancements, Sobel and Prewitt still show extra edges and noise. However, with simpler edge detection techniques like the Caffario method [10], the result is completely filled with noise. Moreover, we believe that our DFC technique can be used to generate association rules for mining abnormal mammograms. This will be left to our future research work. Finally, we are currently involved in developing global measures for measuring the coherence of our DFC method in comparison with the other techniques such as Canny or Gabor GEF filters. This will enable us quantitatively to determine the quality of the developed technique. We aim, in this area, to benefit from the experience of other researchers such as Mike Brady (http://www.robots.ox.ac.uk/~mvl/).

## References


[1] Shiffman, S. Rubin, G. and Napel, S. "Medical Image Segmentation using Analysis of Isolable-Contour Maps," *IEEE Transactions on Medical Imaging*, **19**(11), pp. 1064–1074, 2000.

[2] Horn, B. K. P. *Robot Vision*, MIT Press, Cambridge, MA, USA, 1986.

[3] Pal, N. and Pal, S. "A Review on Image Segmentation Techniques," *Pattern Recognition*, **26**, pp. 1277–1294, 1993.

[4] Batchelor, B. and Waltz, F. *Interactive Image Processing for Machine Vision*, Springer Verlag, New York, 1993.

[5] Gonzalez, R. and Woods, R. *Digital Image Processing*, 2nd Edition, Addison-Wesley, 2002.

[6] Parker, J. R. *Algorithms for Image Processing and Computer Vision*, Wiley Computer Publishing, 1997.

[7] Canny, J. "A Computational Approach to Edge Detection," *IEEE Transactions on PAMI*, **8**(6), pp. 679–698, 1986.

[8] Elvins, T. T. "Survey of Algorithms for Volume Visualization," *Computer Graphics*, **26**(3), pp. 194–201, 1992.

[9] Mohammed, S., Yang, L. and Fiaidhi, J. "A Dynamic Fuzzy Classifier for Detecting Abnormalities in Mammograms," *The 1st Canadian Conference on Computer and Robot Vision CRV2004*, May 17–19, 2004, University of Western Ontario, Ontario, Canada, 2004.

[10] Cafforio, C., di Sciascio, E., Guaragnella, C. and Piscitelli, G. "A Simple and Effective Edge Detector". *Proceedings of ICIAP'97*, in Del Bimbo, A. (Ed.), *Lecture Notes on Computer Science*, **1310**, pp. 134–141. 1997.

[11] Hingham, R. P., Brady, J. M. *et al.* "A quantitative feature to aid diagnosis in mammography" *Third International Workshop on Digital Mammography*, Chicago, June 1996.

[12] Costa, L. F. and Cesar, R. M. Junior, *Shape Analysis And Classification: Theory And Practice*, CRC Press, 2000.

[13] Liang, L. R. and Looney, C. G. "Competitive Fuzzy Edge Detection", *International Journal of Applied Soft Computing*, **3**(2), pp. 123–137, 2003.

[14] Looney, C. G. "Nonlinear rule-based convolution for refocusing," *Real Time Imaging*, **6**, pp. 29–37, 2000.

[15] Looney, C. G. *Pattern Recognition Using Neural Networks*, Oxford University Press, New York, 1997.

[16] Looney, C. G. "Radial basis functional link nets and fuzzy reasoning," *Neurocomputing*, **48**(1–4), pp. 489–509, 2002.

[17] Guillemet, H., Benali, H., *et al.* "Detection and characterization of micro calcifications in digital mammography", *Third International Workshop on Digital Mammography*, Chicago, June 1996.

[18] Russo, F. and Ramponi, G. "Fuzzy operator for sharpening of noisy images," *IEE Electronics Letters*, **28** pp. 1715–1717, 1992.

[19] Undrill, P., Gupta, R. *et al.* "The use of texture analysis and boundary refinement to delineate suspicious masses in mammography" *SPIE Image Processing*, **2710**, pp. 301–310, 1996.

[20] Tizhoosh, H. R. *Fuzzy Image Processing*, Springer Verlag, 1997.

[21] van der Zwaag, B. J., Slump, K. and Spaanenburg, L. "On the analysis of neural networks for image processing," in Palade, V., Howlett, R. J. and Jain, L. C. (Eds), *Proceedings of the Seventh International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES'2003)*, Part II, volume 2774 of *Springer LNCS/LNAI*, pp. 950–958, Springer Verlag, 2003.

[22] Mammographic Image Analysis Society (MIAS) http://www.wiau.man.ac.uk/services/MIAS/MIASweb.html.

[23] Davies, D. H. and Dance, D. R. "Automatic computer detection of subtle calcifications in radiographically dense breasts", *Physics in Medicine and Biology*, **37**(6), pp. 1385–1390, 1992.

[24] Giger, M. L. "Computer-aided diagnosis", *Syllabus: 79th Scientific Assembly of the Radiological Society of North America*, pp. 283–298, 1993.

[25] Maxwell, B. A. and Brubaker, S. J. "Texture Edge Detection Using the Compass Operator," *British Machine Vision Conference*, 2003.

[26] Netsch, T. "Detection of micro calcification clusters in digital mammograms: A space scale approach", *Third International Workshop on Digital Mammography*, Chicago, June 1996.

[27] McLeod, G., Parkin, G., *et al.* "Automatic detection of clustered microcalcifications using wavelets", *Third International Workshop on Digital Mammography*, Chicago, June 1996.

[28] Neto, M. B., Siqueira, U, W. N. *et al.* "Mammographic calcification detection by mathematical morphology methods", *Third International Workshop on Digital Mammography*, Chicago, June 1996.

[29] Bovik, A. C. *et al.* "The effect of median filtering on edge estimation and detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-9**, pp. 181–194, 1987.

[30] Bazzani, A. *et al.*, "System For Automatic Detection of Clustered Microcalcifications in Digital Mammograms," *International Journal of Modern Physics C*, **11**(5) pp. 1–12, 2000.

[31] Halls, S. B. MD http://www.halls.md/breast/density.htm, November 10, 2003.
[32] Antonie, M.-L., Zäiane, O. R. and Coman, A. "Application of Data Mining Techniques for Medical Image Classification," *International Workshop on Multimedia Data Mining MDM/KDD2001*, San Francisco, August 26, 2001.
[33] Mohammed, S., Fiaidhi, J. and Yang, L. "Morphological Analysis of Mammograms using Visualization Pipelines," *Pakistan Journal of Information & Technology*, **2**(2), pp. 178–190, 2003.

# 16

# Feature Extraction and Compression with Discriminative and Nonlinear Classifiers and Applications in Speech Recognition

**Xuechuan Wang**
INRS-EMT, University of Quebec, 800 de la Gauchetière West, Montreal, Quebec H5A 1K6, Canada

*Feature extraction is an important component of a pattern classification system. It performs two tasks: transforming an input parameter vector into a feature vector and/or reducing its dimensionality. A well-defined feature extraction algorithm makes the classification process more effective and efficient. Two popular feature extraction methods are Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA). The Minimum Classification Error (MCE) training algorithm, which was originally proposed as a discriminative classifier, provides an integrated framework for feature extraction and classification. The Support Vector Machine (SVM) is a recently developed pattern classification algorithm, which uses nonlinear kernel functions to achieve nonlinear decision boundaries in the parametric space. In this chapter, the frameworks of LDA, PCA, MCE and SVM are first introduced. An integrated feature extraction and classification algorithm, the Generalized MCE (GMCE) training algorithm is discussed. Improvements on the performance of MCE and SVM classifiers using feature extraction are given on both Deterding vowels and the TIMIT continue speech database.*

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

# 1. Introduction

Pattern recognition deals with mathematical and technical aspects of classifying different objects through their observable information, such as gray levels of pixels for an image, energy levels in the frequency domain for a waveform and the percentage of certain contents in a product. The objective of pattern recognition is achieved in a three-step procedure, as shown in Figure 16.1. The observable information of an unknown object is first transduced into signals that can be analyzed by computer systems. Parameters and/or features suitable for classification are then extracted from the collected signals. The extracted parameters or features are classified in the final step based on certain types of measure, such as the distance, likelihood or Bayesian, over class models.

Conventional pattern recognition systems have two components: feature analysis and pattern classification, as shown in Figure 16.2. Feature analysis is achieved in two steps: the parameter extraction step and the feature extraction step. In the parameter extraction step, information relevant for pattern classification is extracted from the input data $x(t)$ in the form of a $p$-dimensional parameter vector $\boldsymbol{x}$. In the feature extraction step, the parameter vector $\boldsymbol{x}$ is transformed to a feature vector $\boldsymbol{y}$,

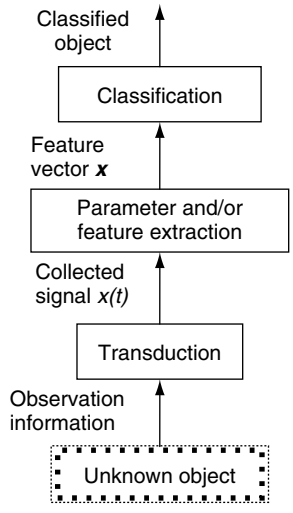



**Figure 16.1** A typical pattern recognition procedure.

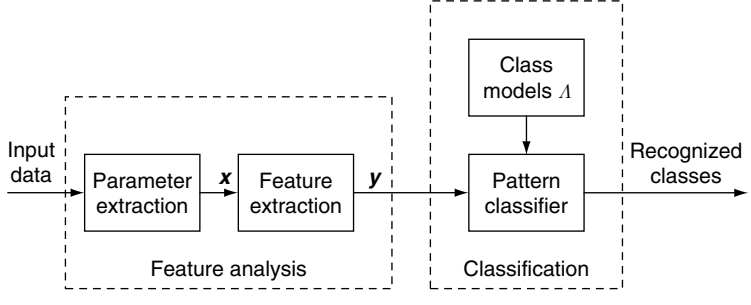



**Figure 16.2** A conventional pattern recognition system.

which has a dimensionality $m(m \leq p)$. If the parameter extractor is properly designed so that the parameter vector $\boldsymbol{x}$ is matched to the pattern classifier and its dimensionality is low, then there is no necessity for the feature extraction step. However in practice, parameter vectors are not suitable for pattern classifiers. For example, speech signals, which are time-varying signals, have time-invariant components and may be mixed up with noise. The time-invariant components and noise will increase the correlation between parameter vectors and degrade the performance of pattern classification systems. The corresponding parameter vectors thus have to be decorrelated before being applied to a classifier based on Gaussian mixture models (with diagonal variance matrices). Furthermore, the dimensionality of parameter vectors is normally very high and needs to be reduced for the sake of less computational cost and system complexity. For these reasons, feature extraction has been an important problem in pattern recognition tasks.

Feature extraction can be conducted independently or jointly with either parameter extraction or classification. LDA and PCA are the two popular independent feature extraction methods. Both of them extract features by projecting the original parameter vectors into a new feature space through a linear transformation matrix. But they optimize the transformation matrix with different intentions. PCA optimizes the transformation matrix by finding the largest variations in the original feature space [1–3]. LDA pursues the largest ratio of *between*-class variation and *within*-class variation when projecting the original feature to a subspace [4–6].

The drawback of independent feature extraction algorithms is that their optimization criteria are different from the classifier's minimum classification error criterion, which may cause inconsistency between feature extraction and the classification stages of a pattern recognizer, and consequently degrade the performance of classifiers [7]. A direct way to overcome this problem is to conduct feature extraction and classification jointly with a consistent criterion. The MCE training algorithm [7–9] provides such an integrated framework, as shown in Figure 16.3. It is a type of discriminant analysis but achieves a minimum classification error directly when extracting features. This direct relationship has made the MCE training algorithm widely popular in a number of pattern recognition applications, such as dynamic time-warping based speech recognition [10,11] and Hidden Markov Model (HMM) based speech and speaker recognition [12–14].

The MCE training algorithm is a linear classification algorithm, of which the decision boundaries generated are straight lines. The advantage of linear classification algorithms is their simplicity and computational efficiency. However, linear decision boundaries have little computational flexibility and are unable to handle data sets with concave distributions. SVM is a recently developed pattern classification algorithm with nonlinear formulation. It is based on the idea that the classification that affords dot-products can be computed efficiently in higher dimensional feature spaces [15–17]. The classes which are not linearly separable in the original parametric space can be linearly separated in

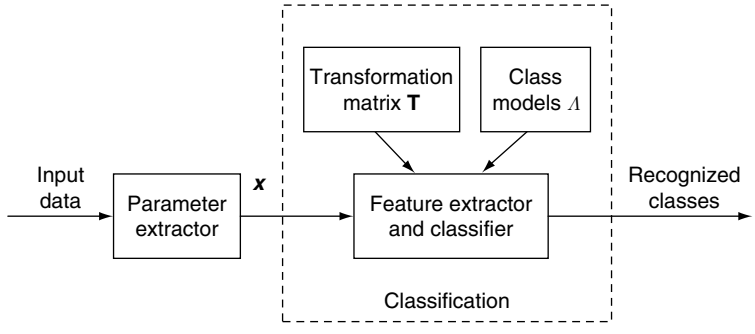



**Figure 16.3** An integrated feature extraction and classification system.

the higher dimensional feature space. Because of this, SVM has the advantage that it can handle the classes with complex nonlinear decision boundaries. SVM has now evolved into an active area of research [18–21].

This chapter will first introduce the major feature extraction methods – LDA and PCA. The MCE algorithm for integrated feature extraction and classification and the nonlinear formulation of SVM are then introduced. Feature extraction and compression with MCE and SVM are discussed subsequently. The performances of these feature extraction and classification algorithms are compared and discussed based on the experimental results on Deterding vowels and TIMIT continuous speech databases.

## 2. Standard Feature Extraction Methods

### *2.1 Linear Discriminant Analysis*

The goal of linear discriminant analysis is to separate the classes by projecting class samples from $p$-dimensional space onto a finely orientated line. For a $K$-class problem, $m = \min(K-1, p)$ different lines will be involved. Thus, the projection is from a $p$-dimensional space to a $c$-dimensional space [22].

Suppose we have $K$ classes, $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_K$. Let the $i$th observation vector from the $\mathrm{X}_j$ be $\mathbf{x}_{ji}$, where $j = 1, \ldots, J$ and $i = 1, \ldots, N_j$. $J$ is the number of classes and $N_j$ is the number of observations from class $j$. The *within*-class covariance matrix $\mathbf{S}_\mathrm{w}$ and *between*-class covariance matrix $\mathbf{S}_\mathrm{b}$ are defined as:

$$\mathbf{S}_\mathrm{w} = \sum_{j=1}^{K} S_j = \sum_{j=1}^{K} \frac{1}{N_j} \sum_{i=1}^{N_j} (x_{ji} - \mu_j)(x_{ji} - \mu_j)^\mathrm{T} \tag{16.1}$$

$$\mathbf{S}_\mathrm{b} = \sum_{j=1}^{K} N_j (\mu_j - \mu)(\mu_j - \mu)^\mathrm{T}$$

where $\mu_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_{ji}$ is the mean of class $j$ and $\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the global mean.

The projection from observation space to feature space is accomplished by a linear transformation matrix $\mathbf{T}$:

$$y = \mathbf{T}^\mathrm{T} x. \tag{16.2}$$

The corresponding *within*-class and *between*-class covariance matrices in the feature space are:

$$\tilde{\mathbf{S}}_\mathrm{w} = \sum_{j=1}^{K} \frac{1}{N_j} \sum_{i=1}^{N_j} (y_{ji} - \tilde{\mu}_j)(y_{ji} - \tilde{\mu}_j)^\mathrm{T} \tag{16.3}$$

$$\tilde{\mathbf{S}}_\mathrm{b} = \sum_{j=1}^{K} N_j (\tilde{\mu}_j - \tilde{\mu})(\tilde{\mu}_j - \tilde{\mu})^\mathrm{T}$$

where $\tilde{\mu}_j = \frac{1}{N_j} \sum_{i=1}^{N_j} y_{ji}$ and $\tilde{\mu} = \frac{1}{N} \sum_{i=1}^{N} y_i$. It is straightforward to show that:

$$\tilde{\mathbf{S}}_\mathrm{w} = \mathbf{T}^\mathrm{T} \mathbf{S}_\mathrm{w} \mathbf{T} \tag{16.4}$$

$$\tilde{\mathbf{S}}_\mathrm{b} = \mathbf{T}^\mathrm{T} \mathbf{S}_\mathrm{b} \mathbf{T}$$

A *linear discriminant* is then defined as the linear functions for which the objective function

$$J(T) = \frac{\left|\tilde{\mathbf{S}}_\mathrm{b}\right|}{\left|\tilde{\mathbf{S}}_\mathrm{w}\right|} = \frac{\left|\mathbf{T}^\mathrm{T} \mathbf{S}_\mathrm{B} \mathbf{T}\right|}{\left|\mathbf{T}^\mathrm{T} \mathbf{S}_\mathrm{W} \mathbf{T}\right|} \tag{16.5}$$

is maximal. It can be shown that the solution of Equation (16.5) is that the $i$th column of an optimal $\mathbf{T}$ is the generalized eigenvector corresponding to the $i$th largest eigenvalue of matrix $\mathbf{S}_\mathrm{w}^{-1} \mathbf{S}_\mathrm{b}$ [6].

### *2.2 Principal Component Analysis*

PCA is a well-established technique for feature extraction and dimensionality reduction [2,23]. It is based on the assumption that most information about classes is contained in the directions along which the variations are the largest. The most common derivation of PCA is in terms of a standardized linear projection which maximizes the variance in the projected space [1]. For a given $p$-dimensional data set X, the $m$ principal axes $T_1, T_2, \ldots, T_m$, where $l \leq m \leq p$, are orthonormal axes onto which the retained variance is maximum in the projected space. Generally, $T_1, T_2, \ldots, T_m$, can be given by the $m$ leading eigenvectors of the sample covariance matrix $\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^{\mathrm{T}}$, where $x_i \in \mathrm{X}$, $\mu$ is the sample mean and $N$ is the number of samples, so that:

$$\mathbf{S}T_i = \lambda_i T_i, \quad i \in 1, \cdots, m \tag{16.6}$$

where $\lambda_i$ is the $i$th largest eigenvalue of $\mathbf{S}$. The $m$ principal components of a given observation vector $\mathbf{x} \in \mathrm{X}$ are given by:

$$y = [y_1, \cdots, y_m] = \left[T_1^{\mathrm{T}}\mathbf{x}, \cdots, T_m^{\mathrm{T}}\mathbf{x}\right] = \mathbf{T}^{\mathrm{T}}\mathbf{x}. \tag{16.7}$$

The $m$ principal components of $\boldsymbol{x}$ are decorrelated in the projected space [2]. In multiclass problems, the variations of data are determined on a global basis, that is, the principal axes are derived from a global covariance matrix:

$$\hat{\mathbf{S}} = \frac{1}{N}\sum_{j=1}^{K}\sum_{i=1}^{N_j}(x_{ji} - \hat{\mu})(x_{ji} - \hat{\mu})^{\mathrm{T}} \tag{16.8}$$

where $\hat{\mu}$ is the global mean of all the samples, $K$ is the number of classes, $N_j$ is the number of samples in class $j$, $N = \sum_{j=1}^{K} N_j$ and $x_{ji}$ represents the $i$th observation from class $j$. The principal axes $T_1, T_2, \ldots, T_m$ are therefore the $m$ leading eigenvectors of $\hat{\mathbf{S}}$:

$$\hat{\mathbf{S}}T_i = \hat{\lambda}_i T_i, \quad i \in 1, \cdots, m \tag{16.9}$$

where $\hat{\lambda}_i$ is the $i$th largest eigenvalue of $\hat{\mathbf{S}}$. An assumption made for feature extraction and dimensionality reduction by PCA is that most information of the observation vectors is contained in the subspace spanned by the first $m$ principal axes, where $m < p$. Therefore, each original data vector can be represented by its principal component vector with dimensionality $m$.

## 3. The Minimum Classification Error Training Algorithm

### *3.1 Derivation of the MCE Criterion*

Consider an input vector $\mathbf{x}$, the classifier makes its decision by the following decision rule:

$$\mathbf{x} \in \text{Class } k \quad \text{if} \quad g_k(\mathbf{x}, \Lambda) = \max_{\text{for all } i \in K} g_i(\mathbf{x}, \Lambda) \tag{16.10}$$

where $g_i(\mathbf{x}, \Lambda)$ is a discriminant function of $\mathbf{x}$ to class $i$, $\Lambda$ is the parameter set and $K$ is the number of classes. The negative of $g_k(\mathbf{x}, \Lambda) - \max_{\text{for all } i \neq k} g_i(\mathbf{x}, \Lambda)$ can be used as a measure of misclassification of $\mathbf{x}$. This form, however, is not differentiable and needs further modification. In [7], a modified version is introduced as a misclassification measure. For the $k$th class, it is given by:

$$d_k(\boldsymbol{x}, \Lambda) = -g_k(\boldsymbol{x}, \Lambda) + \left[\frac{1}{N-1}\sum_{\text{for all } i \neq k}(g_i(\boldsymbol{x}, \Lambda))^{\eta}\right]^{1/\eta} \tag{16.11}$$

where $\eta$ is a positive number and $g_k(\boldsymbol{x}, \Lambda)$ is the discriminant of observation $\boldsymbol{x}$ to its known class $k$. When $\eta$ approaches $\infty$, it reduces to:

$$d_k(\boldsymbol{x}, \Lambda) = -g_k(\boldsymbol{x}, \Lambda) + g_j(\boldsymbol{x}, \Lambda) \tag{16.12}$$

where class $j$ has the largest discriminant value among all the classes other than class $k$. Obviously, $d_k(\boldsymbol{x}, \Lambda) > 0$ implies misclassification, $d_k(\boldsymbol{x}, \Lambda) < 0$ means correct classification and $d_k(\boldsymbol{x}, \Lambda) = 0$ suggests that $\boldsymbol{x}$ sits on the boundary. The loss function is then defined as a monotonic function of misclassification measure. The sigmoid function is often chosen since it is a smoothed zero–one function suitable for the gradient descent algorithm. The loss function is thus given as:

$$l_k(\boldsymbol{x}, \Lambda) = f(d_k(\boldsymbol{x}, \Lambda)) = \frac{1}{1 + e^{-\zeta d_k(\boldsymbol{x}, \Lambda)}} \tag{16.13}$$

where $\zeta > 0$. For a training set X, the empirical loss is defined as:

$$L(\Lambda) = E\{l_k(\boldsymbol{x}, \Lambda)\} = \sum_{k=1}^{K} \sum_{i=1}^{N_k} l_k(\boldsymbol{x}^{(i)}, \Lambda) \tag{16.14}$$

where $N_k$ is the number of samples in class $k$. Clearly, minimizing the above empirical loss function will lead to the minimization of the classification error. As a result, Equation (16.14) is called the MCE criterion [7,8]. The class parameter set $\Lambda$ is therefore obtained by minimizing the loss function through the steepest gradient descent algorithm. This is an iterative algorithm and the iteration rules are:

$$\begin{aligned} \Lambda_{t+1} &= \Lambda_t - \varepsilon \nabla L(\Lambda) \big|_{\Lambda = \Lambda_t} \\ \nabla L(\Lambda) &= \begin{bmatrix} \partial L / \partial \lambda_1 \\ \vdots \\ \partial L / \partial \lambda_d \end{bmatrix} \end{aligned} \tag{16.15}$$

where $t$ denotes the $t$th iteration, $\partial\lambda_1, \ldots, \partial\lambda_d \in \Lambda$ are class parameters and $\varepsilon > 0$ is the adaptation constant. For $s = 1, 2, \ldots, d$, the gradient $\nabla L(\Lambda)$ can be computed as follows:

$$\begin{aligned} \frac{\partial L}{\partial \lambda_s} &= \zeta \sum_{i=1}^{N_k} L^{(i)}(1 - L^{(i)}) \frac{\partial g_k(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \lambda_s} \quad \text{if } \lambda_s \in \text{ class k} \\ \frac{\partial L}{\partial \lambda_s} &= -\zeta \sum_{i=1}^{N_j} L^{(i)}(1 - L^{(i)}) \frac{\partial g_j(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \lambda_s} \quad \text{if } \lambda_s \in \text{ class j} \end{aligned} \tag{16.16}$$

In the case of Mahalanobis distance measure-based discriminant functions, $\Lambda = \{\mu, \boldsymbol{\Sigma}\}$, where $\mu$ is the class mean and $\boldsymbol{\Sigma}$ is the covariance matrix. The differentiation of discriminant functions with respect to $\Lambda$ is:

$$\begin{aligned} \frac{\partial g_m(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \mu} &= -(\boldsymbol{x} - \mu)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \mu), \ m = 1, \cdots, K \\ \frac{\partial g_m(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \boldsymbol{\Sigma}} &= -(\boldsymbol{x} - \mu)^{\mathrm{T}} (\boldsymbol{\Sigma}^{-1})^2 (\boldsymbol{x} - \mu), \ m = 1, \cdots, K \end{aligned} \tag{16.17}$$

An alternative definition of the misclassification measure can be used to enhance the control of the joint behavior of discriminant functions $g_k(\boldsymbol{x}, \Lambda)$ and $g_j(\boldsymbol{x}, \Lambda)$. The alternative misclassification is defined as follows:

$$d_k(\boldsymbol{x}, \Lambda) = \frac{\left[ \frac{1}{N-1} \sum_{\text{for all } i \neq k} (g_i(\boldsymbol{x}, \Lambda))^{\eta} \right]^{1/\eta}}{g_k(\boldsymbol{x}, \Lambda)} \tag{16.18}$$

In the extreme case, i.e. $\eta \to \infty$, Equation (16.18) becomes:

$$d_k(\boldsymbol{x}, \Lambda) = \frac{g_j(\boldsymbol{x}, \Lambda)}{g_k(\boldsymbol{x}, \Lambda)} \tag{16.19}$$

The class parameters and transformation matrix are optimized using the same adaptation rules as shown in Equation (16.15). The gradients with respect to $\Lambda$ are computed as

$$\begin{aligned}
\frac{\partial L}{\partial \lambda_s} &= -\zeta \sum_{i=1}^{N_k} L^{(i)}(1-L^{(i)}) \frac{g_j(\boldsymbol{x}^{(i)}, \Lambda)}{[g_k(\boldsymbol{x}^{(i)}, \Lambda)]^2} \frac{\partial g_k(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \lambda_s} \quad \text{if } \lambda_s \in \text{class } k \\
\frac{\partial L}{\partial \lambda_s} &= \zeta \sum_{i=1}^{N_j} L^{(i)}(1-L^{(i)}) \frac{1}{g_k(\boldsymbol{x}^{(i)}, \Lambda)} \frac{\partial g_j(\boldsymbol{x}^{(i)}, \Lambda)}{\partial \lambda_s} \quad \text{if } \lambda_s \in \text{class } j
\end{aligned} \tag{16.20}$$

where $\lambda_s \in \Lambda, s = 1, \cdots, d$. The differentiation of discriminant functions can be computed by Equation (16.17).

## *3.2 Using MCE Training Algorithms for Dimensionality Reduction*

As with other feature extraction methods, MCE reduces feature dimensionality by projecting the input vector into a lower dimensional feature space through a linear transformation $\mathbf{T}_{m\times p}$, where $m < p$. Let the class parameter set in the feature space be $\tilde{\Lambda}$. Accordingly, the loss function becomes:

$$l_k(\boldsymbol{x}, \tilde{\Lambda}, \mathbf{T}) = f(d_k(\mathbf{T}\boldsymbol{x}, \tilde{\Lambda})) = \frac{1}{1 + e^{-\zeta d_k(T\boldsymbol{x}, \tilde{\Lambda})}} \tag{16.21}$$

The empirical loss over the whole data set is given by:

$$L(\tilde{\Lambda}, \mathbf{T}) = E\left\{ l_k(\boldsymbol{x}, \tilde{\Lambda}, \mathbf{T}) \right\} = \sum_{k=1}^{K} \sum_{i=1}^{N_k} l_k(\boldsymbol{x}^{(i)}, \tilde{\Lambda}, \mathbf{T}) \tag{16.22}$$

Since Equation (16.22) is a function of $\mathbf{T}$, the elements in $\mathbf{T}$ can be optimized together with the parameter set $\tilde{\Lambda}$ in the same gradient descent procedure. The adaptation rule for $\mathbf{T}$ is:

$$\mathbf{T}_{sq}(t+1) = \mathbf{T}_{sq}(t) - \varepsilon \frac{\partial L}{\partial \mathbf{T}_{sq}} \Big|_{\mathbf{T}_{sq}=\mathbf{T}_{sq}(t)} \tag{16.23}$$

where $t$ denotes the $t$th iteration, $\varepsilon$ is the adaptation constant or learning rate and $s$ and $q$ are the row and column indicators of transformation matrix $\mathbf{T}$. The gradient with respect to $\mathbf{T}$ can be computed by

*Conventional MCE*:

$$\frac{\partial \mathrm{L}}{\partial \mathbf{T}_{sq}} = \zeta \sum_{k=1}^{K} \sum_{i=1}^{N_k} L^{(i)}(1-L^{(i)}) \times \left( \frac{\partial g_k(\mathbf{T}x^{(i)}, \tilde{\Lambda})}{\partial \mathbf{T}_{sq}} - \frac{\partial g_j(\mathbf{T}x^{(i)}, \tilde{\Lambda})}{\partial \mathbf{T}_{sq}} \right) \tag{16.24}$$

*Alternative MCE*:

$$\begin{aligned}
\frac{\partial \mathrm{L}}{\partial \mathbf{T}_{sq}} = \zeta \sum_{k=1}^{K} \sum_{i=1}^{N_k} & L^{(i)}(1-L^{(i)}) \\
& \times \frac{(\partial g_j(\mathbf{T}x^{(i)}, \tilde{\Lambda})/\partial \mathbf{T}_{sq}) g_k(\mathbf{T}x^{(i)}, \tilde{\Lambda}) - (\partial g_k(\mathbf{T}x^{(i)}, \tilde{\Lambda})/\partial \mathbf{T}_{sq}) g_j(\mathbf{T}x^{(i)}, \tilde{\Lambda})}{\left[ g_k(\mathbf{T}x^{(i)}, \tilde{\Lambda}) \right]^2}.
\end{aligned}$$

where, in Mahalanobis distance-based discriminant functions:

$$\frac{\partial g_m(\mathbf{T}x^{(i)}, \tilde{\Lambda})}{\partial \mathbf{T}} = (\mathbf{T}\boldsymbol{x} - \tilde{\mu})^{\mathrm{T}} \Sigma^{-1} \boldsymbol{x} + \boldsymbol{x}^{\mathrm{T}} \Sigma^{-1} (\mathbf{T}\boldsymbol{x} - \tilde{\mu}), \ m = 1, \cdots, K. \tag{16.25}$$

# 4. Support Vector Machines

## 4.1 Constructing an SVM

Considering a two-class case, suppose the two classes are $\Omega_1$ and $\Omega_2$ and we have a set of training data $X = \{x_1, x_2, \ldots, x_N\} \subset \mathrm{R}^p$. The training data are labeled by the following rule:

$$y_i = \begin{cases} +1, & \boldsymbol{x} \in \Omega_1 \\ -1, & \boldsymbol{x} \in \Omega_2 \end{cases} \tag{16.26}$$

The basic idea of SVM estimation is to project the input observation vectors nonlinearly into a high-dimensional feature space $\mathtt{F}$ and then compute a linear function in $\mathtt{F}$. The functions take the form:

$$f(x) = (w \cdot \Phi(x)) + b, \tag{16.27}$$

with $\Phi : \mathrm{R}^{\mathrm{p}} \to \mathtt{F}$ and $w \in \mathtt{F}$, where $(\cdot)$ denotes the dot product. Ideally, all the data in these two classes satisfy the following constraint:

$$y_i(w \cdot \Phi(x_i)) + b - 1 \geq 0 \qquad \forall i, \tag{16.28}$$

Considering the points $\Phi(x_i)$ in $\mathtt{F}$ for which the equality in Equation (16.28) holds, these points lie on two hyperplanes $H_1 : (w \cdot \Phi(x_i)) + b = +1$ and $H_2 : (w \cdot \Phi(x_i)) + b = -1$. These two hyperplanes are parallel and no training points fall between them. The margin between them is $2/\|w\|$. Therefore, we can find a pair of hyperplanes with maximum margin by minimizing $\|w\|^2$ subject to Equation (16.28) [24]. This problem can be written as a convex optimization problem:

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2}\|w\|^2 \\ &\text{Subject to} \quad y_i(w \cdot \Phi(x_i)) + b - 1 \quad \forall i \end{aligned} \tag{16.29}$$

where the first function is the *primal* objective function and the second function is the corresponding constraints. Equation (16.29) can be solved by constructing a Lagrange function from both the primal function and the corresponding constraints. Hence, we introduce positive Lagrange multipliers $\alpha_i$, $i = 1, \ldots, N$, one for each constraint in Equation (16.29). The Lagrange function is given by:

$$L_{\mathrm{P}} = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (w \cdot \Phi(x_i) + b) + \sum_{i=1}^{N} \alpha_i \tag{16.30}$$

$L_{\mathrm{P}}$ must be minimized with respect to $w$ and $b$, which requires the gradient of $L_P$ to vanish with respect to $w$ and $b$. The gradients are given by:

$$\begin{aligned} \frac{\partial L_{\mathrm{P}}}{\partial w_s} &= w_s - \sum_{i=1}^{N} \alpha_i y_i \Phi(x_{is}) = 0, \quad s = 1, \cdots p \\ \frac{\partial L_{\mathrm{P}}}{\partial b} &= -\sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{16.31}$$

where $p$ is the dimension of space F. Combining these conditions and other constraints on primal functions and Lagrange multipliers, we obtain the *Karush–Kuhn–Tucker* (KKT) conditions:

$$\begin{aligned}
&\frac{\partial L_{\mathrm{P}}}{\partial w_s} = w_s - \sum_{i=1}^{N} \alpha_i y_i \Phi(x_{is}) = 0, \quad s = 1, \cdots p \\
&\frac{\partial L_{\mathrm{P}}}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \\
&y_i(w \cdot \Phi(x_i)) + b - 1 \geq 0 \quad \forall i \\
&\alpha_i \geq 0 \quad \forall i, \\
&\alpha_i(y_i(w \cdot \Phi(x_i)) + b - 1) = 0 \quad \forall i
\end{aligned} \tag{16.32}$$

where $w, b$ and $\alpha$ are the variables to be solved. From KKT condition Equation (16.32) we obtain:

$$\begin{aligned}
w &= \sum_{i=1}^{N} \alpha_i y_i \Phi(x_i) \\
\sum_{i=1}^{N} \alpha_i y_i &= 0
\end{aligned} \tag{16.33}$$

Therefore,

$$\begin{aligned}
f(x) &= \sum_{i=1}^{N} \alpha_i y_i (\Phi(x_i) \cdot \Phi(x_i)) \\
&= \sum_{i=1}^{N} \alpha_i y_i k(x_i, \boldsymbol{x}) + b
\end{aligned} \tag{16.34}$$

where $k(x_i, \boldsymbol{x}) = (\Phi(x_i) \cdot \Phi(x_j))$ is a kernel function that uses the dot product in the feature space. Substitute Equation (16.33) into Equation (16.30). This leads to maximization of the dual function $L_{\mathrm{D}}$:

$$L_{\mathrm{D}} = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k_{ij} + \sum_{i=1}^{N} \alpha_i \tag{16.35}$$

Writing the dual function incorporating the constraints, we obtain the dual optimization problem:

$$\begin{aligned}
&\text{Maximize} \ -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k_{ij} + \sum_{i=1}^{N} \alpha_i \\
&\text{Subject to} \ \sum_{i=1}^{N} \alpha_i y_i = 0 \\
&\qquad\qquad \alpha_i \geq 0 \quad \forall i
\end{aligned} \tag{16.36}$$

Both the primal problem $L_{\mathrm{P}}$ (Equation (16.30)) and the dual problem $L_{\mathrm{D}}$ (Equation (16.35)) are constructed from the same objective function but with different constraints. Optimization of this primal–dual problem is a type of convex optimization problem and can be solved by the interior point

algorithm [21]. However, a discussion of the interior point algorithm is beyond the scope of this chapter. A detailed discussion on this algorithm is given by Vanderbei in [25].

## 4.2 Multiclass SVM Classifiers

SVM is a two-class-based pattern classification algorithm. Therefore, a multiclass-based SVM classifier has to be constructed. So far, the best method of constructing a multiclass SVM classifier is not clear [26]. Scholkopf *et al.*[27] proposed a 'one vs. all' type classifier. Clarkson and Moreno [26] proposed a 'one vs. one' type classifier. Their structures are shown in Figure 16.4.

Both types of classifier are in fact combinations of two-class-based SVM subclassifiers. When an input data vector $\boldsymbol{x}$ enters the classifier, a $K$-dimensional value vector $f^{(i)}(\boldsymbol{x}), i = 1, \ldots, K$ (one dimension for each class) is generated. The classifier then classifies $\boldsymbol{x}$ by the following classification criteria:

$$\boldsymbol{x} \in \text{Class}i \quad \text{if} \quad f^{(i)}(\boldsymbol{x}) = \max_{\text{for all } j \in K} f^{(j)}(\boldsymbol{x}) \tag{16.37}$$

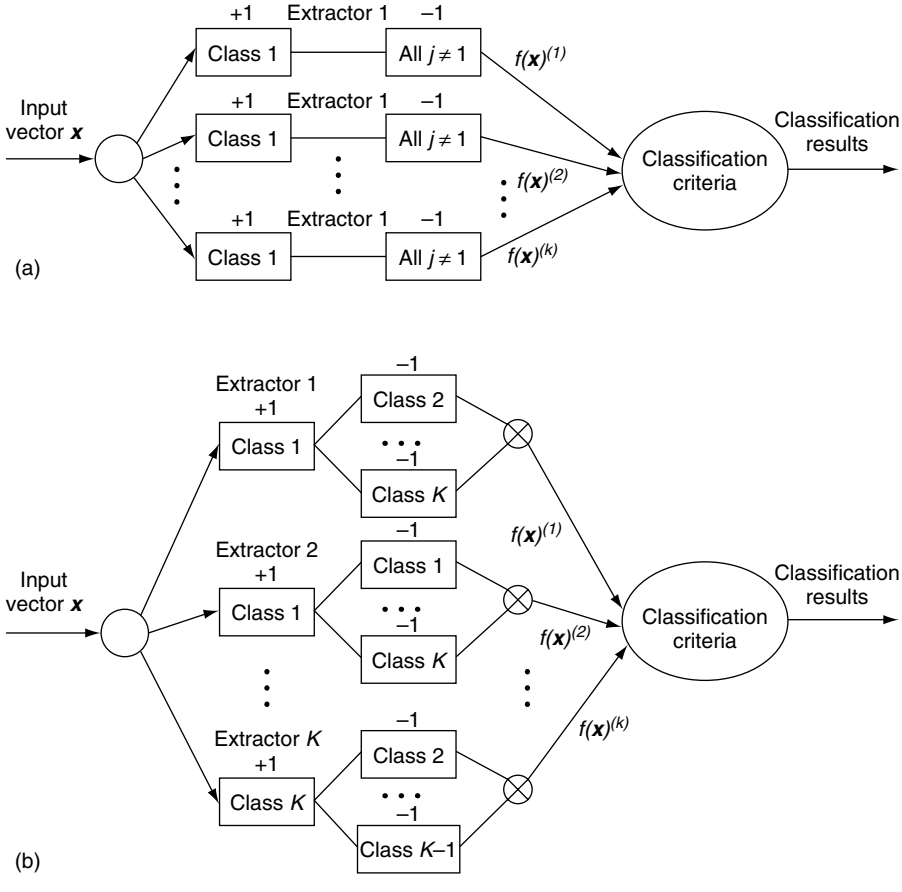



**Figure 16.4** Two types of multiclass SVM classifier. (a) One vs. all; (b) one vs. one.

# 5. Feature Extraction and Compression with MCE and SVM

## 5.1 The Generalized MCE Training Algorithm

One of the major concerns about MCE training for dimensionality reduction is the initialization of the parameters. This is because the gradient descent method used in the MCE training algorithm does not guarantee the global minimum value. The optimality of the MCE training process is largely dependent on the initialization of $\mathbf{T}$ and the class parameter set $\Lambda$.

Among these parameters, transformation matrix $\mathbf{T}$ is crucial to the success of MCE training since it filters the class information to be brought into the decision space. Paliwal *et al.* [9] give an initialization of the MCE training algorithm, in which $\mathbf{T}$ is taken to be a unity matrix. However, in many cases, this is a convenient way of initialization rather than an effective way, because the classification criterion has not been considered in the initialization. In order to increase the generalization of the MCE training algorithm, it is necessary to embed the classification criteria into the initialization process. From a searching point of view, we can regard MCE training as two sequential search procedures: one is a general but rough search for the initialization of parameters, and the other a local but thorough search for the optimization of parameters. The former search procedure will provide a global optimized initialization of class parameters and the latter will make a thorough search to find the relevant local minimum. Figure 16.5 compares the normal MCE training process to the generalized MCE training process. So far, no criterion for a general searching process has been proposed. However, we can employ current feature extraction methods to this process. In our practice, we employ LDA and PCA for the general searching process for the initialization of class parameters.

## 5.2 Reduced-dimensional SVM

The basic idea of Reduced-Dimensional SVM (RDSVM) to reduce the computational burden is that the total number of computations of SVM can be reduced by reducing the number of computations in kernel functions, since the number of observation vectors $N$ cannot be reduced to a very low level in many cases. An effective way of reducing the number of computations in kernel functions is to reduce the dimensionality of observation vectors.

RDSVM is in fact a combination of feature extraction and SVM algorithms. It has a two-layer structure. The first layer conducts feature extraction and compression, of which the objective is to reduce the dimensionality of the feature space and obtain the largest discriminants between classes. The second layer conducts SVM training in the reduced-dimensional feature space, which is provided by the first layer. Thus, the kernel functions will be calculated as follows:

$$k(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) = \Phi(\hat{\boldsymbol{x}}) \cdot \Phi(\hat{\boldsymbol{y}}) = \Phi(\mathbf{T}^{\mathrm{T}}\boldsymbol{x}) \cdot \Phi(\mathbf{T}^{\mathrm{T}}\boldsymbol{y}) = k(\mathbf{T}^{\mathrm{T}}\boldsymbol{x}, \mathbf{T}^{\mathrm{T}}\boldsymbol{y}) \tag{16.38}$$

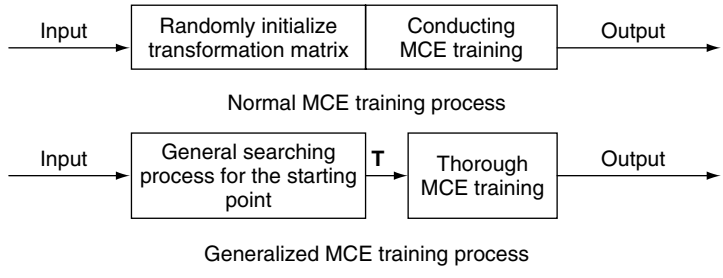



**Figure 16.5** A comparison between the normal and the generalized MCE training processes.

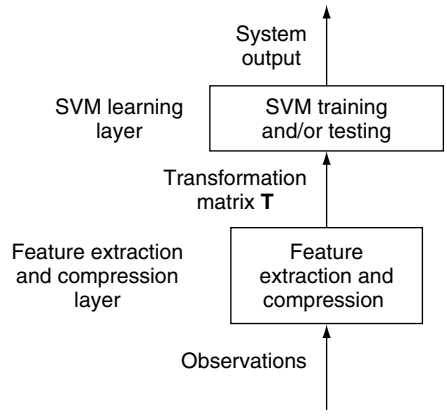



**Figure 16.6** Structure of RDSVM.

where $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{y}}$ are feature vectors in the reduced-dimensional feature space, $\boldsymbol{x}$ and $\boldsymbol{y}$ are observation vectors and $\mathbf{T}$ is the transformation optimized by the first layer. Figure 16.6 shows the structure of RDSVM.

## 6. Classification Experiments

Our experiments focused on vowel recognition tasks. Two databases were used. We started with the Deterding vowels database [28]. The advantage of starting with this is that the computational burden is small. The Deterding database was used to evaluate different types of GMCE training algorithms and SVM classifiers. Then, feature extraction and classification algorithms were tested with the TIMIT database [29]. The feature extraction and classification algorithms involved in the experiments are listed in Table 16.1.

In order to evaluate the performance of the linear feature extraction algorithms (PCA, LDA, MCE and GMCE), we used a minimum distance classifier. Here, a feature vector $\boldsymbol{y}$ is classified into the $j$th class if the distance $d_j(\boldsymbol{y})$ is less than the other distances $d_i(\boldsymbol{y})$, $i = 1, \ldots, K$. We use the Mahalanobis

**Table 16.1** Feature extraction and classification algorithms used in our experiments.

| Parameter used | Dimension | Feature extractor | Classifier |
|---|---|---|---|
| LAR(Deterding) | 10 | PCA | Minimum distance (Mahalanobis) |
| LAR(Deterding) | 10 | LDA | Minimum distance (Mahalanobis) |
| LAR(Deterding) | 10 | MCE | Minimum distance (Mahalanobis) |
| LAR(Deterding) | 10 | GMCE | Minimum distance (Mahalanobis) |
| MFCC(TIMIT) | 21 | PCA | Minimum distance (Mahalanobis) |
| MFCC(TIMIT) | 21 | LDA | Minimum distance (Mahalanobis) |
| MFCC(TIMIT) | 21 | MCE | Minimum distance (Mahalanobis) |
| MFCC(TIMIT) | 21 | GMCE | Minimum distance (Mahalanobis) |
| LAR(Deterding) | 10 | NONE | SVM one vs. one |
| LAR(Deterding) | 10 | NONE | SVM one vs. all |
| MFCC(TIMIT) | 21 | NONE | SVM one vs. one |

distance measure to compute the distance of a feature vector from a given class. Thus, the distance $d_i(\boldsymbol{y})$ is computed as follows:

$$d_i(\boldsymbol{y}) = (\boldsymbol{y} - \mu_i)^{\mathrm{T}} \Sigma_i^{-1} (\boldsymbol{y} - \mu_i) \tag{16.39}$$

where $\mu_i$ is the mean vector of class $i$ and $\sum_i$ is the covariance matrix. In our experiments, we use the full covariance matrix.

Three types of SVM kernel function are evaluated on the Deterding database. The formulation of kernel functions is as follows:

$$\begin{aligned} \textit{Linear kernel}: \; & k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{y} \\ \textit{Polynomial kernel}: \; & k(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} \cdot \boldsymbol{y} + 1)^p \\ \textit{RBF kernel}: \; & k(\boldsymbol{x}, \boldsymbol{y}) = e^{|x-y|^2/2\delta^2} \end{aligned} \tag{16.40}$$

## 6.1 Deterding Database Experiments

The Deterding vowels database has 11 vowel classes, as shown in Table 16.2. This database has been used in the past by a number of researchers for pattern recognition applications [26,28,30,31]. Each of these 11 vowels is uttered six times by 15 different speakers. This gives a total of 990 vowel tokens. A central frame of speech signal is excised from each of these vowel tokens. A tenth order linear prediction analysis is performed on each frame and the resulting Linear Prediction Coefficients (LPCs) are converted to ten Log-Area (LAR) parameters. 528 frames from eight speakers are used to train the models and 462 frames from the remaining seven speakers are used to test the models.

Table 16.3 compares the results for LDA, PCA, the conventional form and the alternative form of the MCE training algorithm. The results show that the alternative MCE training algorithm has the best performance. Thus, we used the alternative MCE in the following experiments.

Two types of GMCE training algorithm were investigated in our Deterding database experiments. One used LDA for the general search and the other used PCA. Figures 16.7 and 16.8 show the experiment results. Since the alternative MCE training algorithm was chosen for MCE training, we denote these two types of GMCE training algorithm as GMCE + LDA and GMCE + PCA, respectively.

**Table 16.2** Vowels and words used in the Deterding database.

| Vowel | Word | Vowel | Word | Vowel | Word | Vowel | Word |
|---|---|---|---|---|---|---|---|
| i | heed | O | hod | I | hid | C: | hoard |
| E | head | U | hood | A | had | u: | who'd |
| a: | hard | 3: | heard | Y | hud | | |

**Table 16.3** Comparison of various feature extractors.

| Database | Conventional MCE (%) | Alternative MCE (%) | LDA (%) | PCA (%) |
|---|---|---|---|---|
| Vowels (Train) | 85.6 | 99.1 | 97.7 | 97.7 |
| Vowels (Test) | 53.7 | 55.8 | 51.3 | 49.1 |

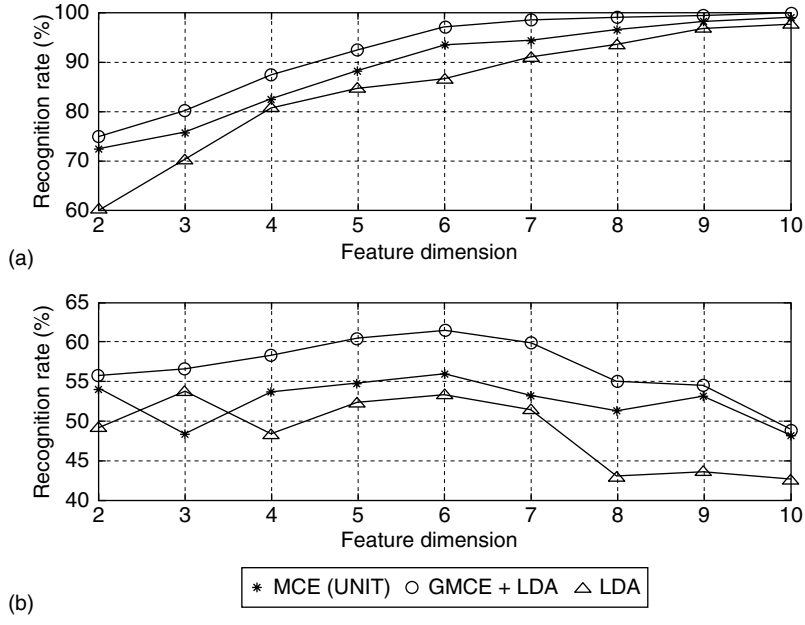



**Figure 16.7** Results of MCE(UNIT), GMCE + LDA and LDA on the Deterding database. (a) Training data; (b) testing data.

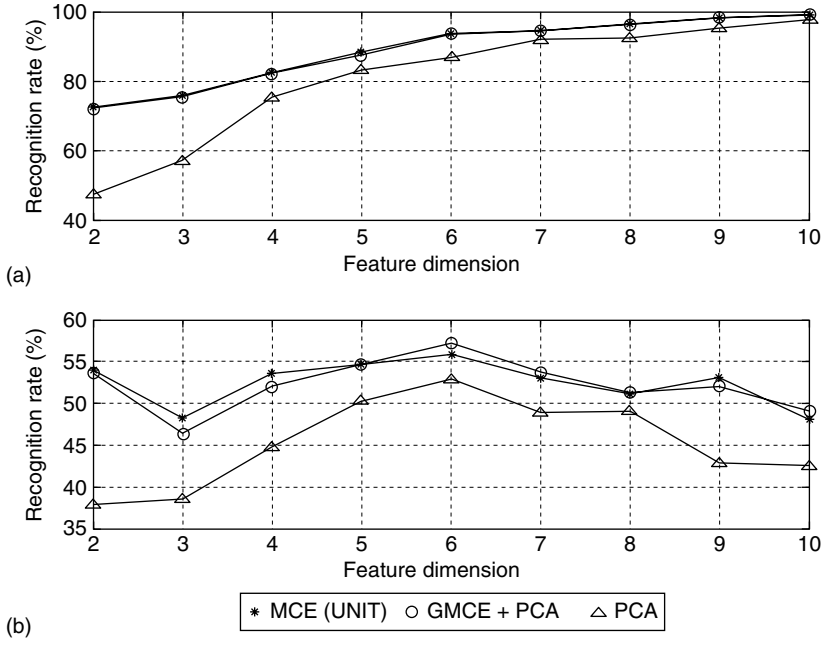



**Figure 16.8** Results of MCE(UNIT), GMCE + PCA and PCA on the Deterding database. (a) Training data; (b) testing data.

The normal alternative MCE training algorithm is denoted as MCE(UNIT). Observations from these results can be summarized as follows:

- The GMCE training algorithm has an improved performance when LDA is used for the general search for the initial transformation matrix and GMCE + LDA demonstrates the best performance among MCE(UNIT), GMCE + PCA, LDA and PCA.
- The performance of the GMCE training algorithm is not improved when PCA is employed for the general searching process.
- The performances of GMCE + LDA and GMCE + PCA on testing data show that the best classification results are usually obtained when the dimensionality is reduced to $50 \sim 70\,\%$.

Table 16.4 shows the classification results of different SVM classifiers. The order of the polynomial kernel function is 3. The classification results show that the performance of the RBF kernel function is the best among the three types of kernel. The overall performance of the 'one vs. one' multiclass classifier is much better than the 'one vs. all' multiclass classifier. Among all the six types of SVM classifier, the 'one vs. one' multiclass classifier with RBF kernel function has the best overall performance and was thus selected for further experiments.

Figure 16.9 gives a comparison of the results of the GMCE training algorithm, LDA, SVM and RDSVM. Since SVM can only be operated in the observation space, i.e. dimension 10, its results are presented as dots on dimension 10. Observations from the performance of RDSVM can be drawn as follows:

- The performance of RDSVM is better than that of SVM on training data on dimension 10, while on testing data it remains the same.
- Both SVM and RDSVM have better performances on dimensions 2 and 10 than the GMCE training algorithm and LDA. The performance of RDSVM is comparable to that of the GMCE training algorithm on training data and is better than that of LDA.
- On testing data, RDSVM performs slightly poorer than the GMCE training algorithm in low-dimensional feature spaces (dimensions 3–5), while on high-dimensional feature spaces (dimensions 6–9), RDSVM has a slightly better performance than the GMCE training algorithm. On dimensions 2 and 10, RDSVM performs much better than the GMCE training algorithm.
- The highest recognition rate on testing data does not appear on the full dimension (10) but on dimension 6.

## 6.2 TIMIT Database Experiments

In order to provide results on a bigger database, we used the TIMIT database for vowel recognition. This database contains a total of 6300 sentences, ten sentences spoken by each of 630 speakers. The

**Table 16.4** Deterding vowel data set classification results using SVM classifiers.

| Kernel | SVM classifier | Training set (%) | Testing set (%) |
|---|---|---|---|
| Linear | one vs. all | 49.43 | 40.91 |
| Linear | one vs. one | 79.73 | 53.03 |
| Polynomial | one vs. all | 59.85 | 42.42 |
| Polynomial | one vs. one | 90.53 | 55.63 |
| RBF | one vs. all | 78.98 | 51.95 |
| RBF | one vs. one | 90.34 | 58.01 |

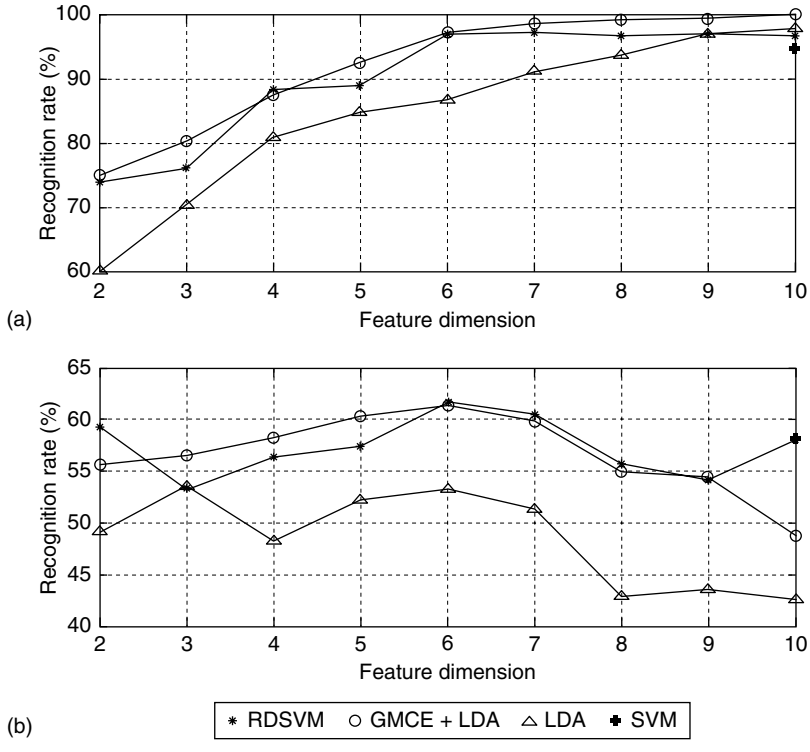



**Figure 16.9** Results of GMCE + LDA, LDA, SVM and RDSVM on the Deterding database. (a) Training data; (b) testing data.

training part of this database was used for training the vowel recognizers and the test part for testing. The vowels used in the classification tasks were selected from the vowels, semi-vowels and nasals given in the TIMIT database. Altogether, 17 vowels, one semi-vowel and one nasal were selected for the vowel classification experiments. The TIMIT database comes with phonemic transcription and associated acoustic phonemic boundaries. The center 20 ms segments of selected vowels were excised from each sentence. Spectral analysis was performed on these segments and each segment was represented by a 21-dimension Mel-Frequency Cepstral Coefficient (MFCC) feature vector. Each vector contained one energy coefficient and 20 MFCCs. The Mahalanobis distance-based minimum distance classifier was used as pattern classifier. Table 16.5 shows the number of segments of each vowel used in the experiment.

### 6.2.1 Comparison of Separate and Integrated Pattern Recognition Systems

Figure 16.10 shows the results of separate pattern recognition systems, PCA and LDA, plus classifier and integrated systems, MCE and SVM, in feature extraction and classification tasks. The dimensionalities used in the experiments were from 3 to 21 – full dimension. The horizontal axis of the figure is the dimension axis. The vertical axis represents the recognition rates. Since SVM is not suitable for dimensionality reduction, it is applied to classification tasks only and the results of

**Table 16.5** Number of observations of selected phonemes in training and testing sets.

| Phoneme | aa | ae | ah | ao | aw | ax | ay | eh | oy | uh |
|---|---|---|---|---|---|---|---|---|---|---|
| Training | 541 | 665 | 313 | 445 | 126 | 207 | 395 | 591 | 118 | 57 |
| Testing | 176 | 214 | 136 | 168 | 40 | 89 | 131 | 225 | 49 | 21 |
| Phoneme | el | en | er | ey | ih | ix | iy | ow | uw | Total |
| Training | 145 | 97 | 384 | 346 | 697 | 583 | 1089 | 336 | 106 | 7241 |
| Testing | 42 | 34 | 135 | 116 | 239 | 201 | 381 | 116 | 37 | 2550 |

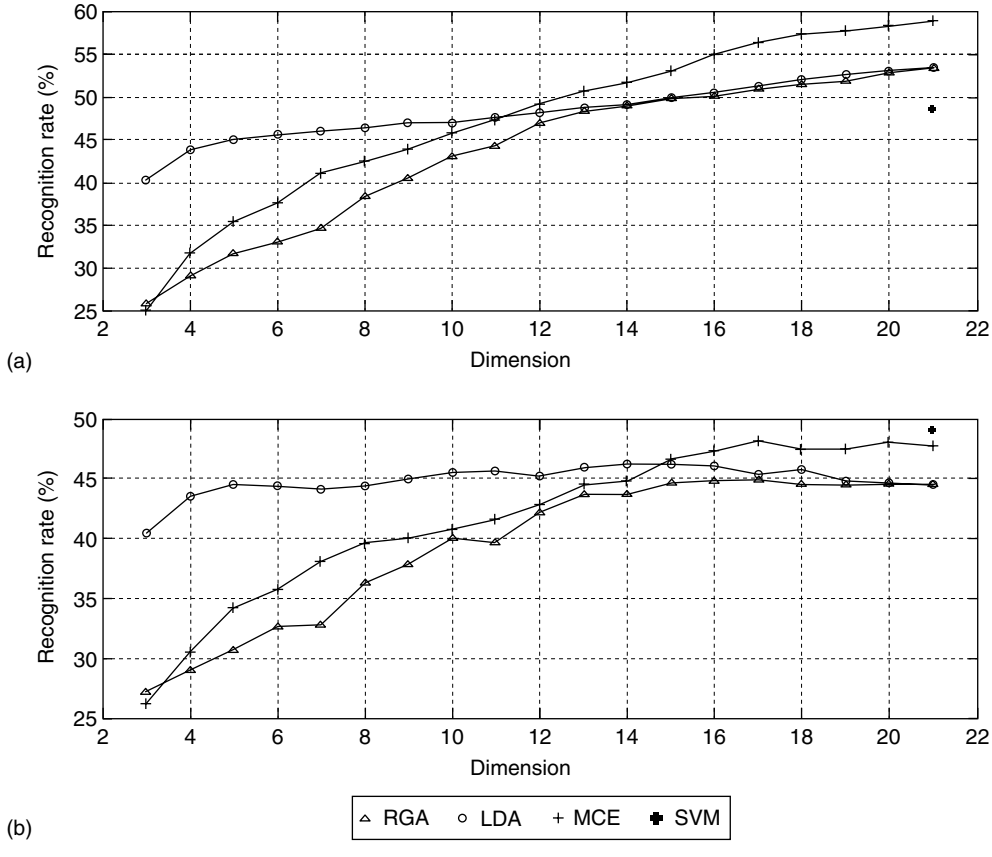



**Figure 16.10** Results of LDA, PCA, MCE and SVM on the TIMIT database. (a) Training data; (b) testing data.

SVM appear in the figure as single points. Observations from Figure 16.10 can be summarized as follows:

- LDA has a fairly flat performance curve. It performs the best in the low-dimensional feature spaces (dimensions 3–12) among LDA, PCA and the MCE training algorithm on training data. On testing data, LDA performs better than PCA and MCE in low-dimensional spaces (dimensions 3–15) too.

- The MCE training algorithm performs better than LDA and PCA in the high-dimensional feature spaces (dimensions 13–21) on training data. On the testing data, the MCE training algorithm performs better than PCA and LDA in the high-dimensional spaces from dimension 16–21.
- The performances of SVM on training data are not as good as those of LDA, PCA and the MCE training algorithm. However, SVM performs much better than LDA, PCA and the MCE training algorithm on testing data.

### 6.2.2 Analysis of the GMCE Training Algorithm

Two types of GMCE were used in this experiment. One employed LDA for the general search, which we denote as GMCE + LDA. The other employed PCA for the general search and we denote this as GMCE + PCA. Results of the experiments are shown in Figures 16.11 and 16.12. Observations from the two figures can be summarized as follows:

- When GMCE uses LDA as the general search tool, the performances of GMCE are better than both LDA and MCE in all dimensions. When GMCE uses PCA in the general search process, the general performances of GMCE are not significantly improved.

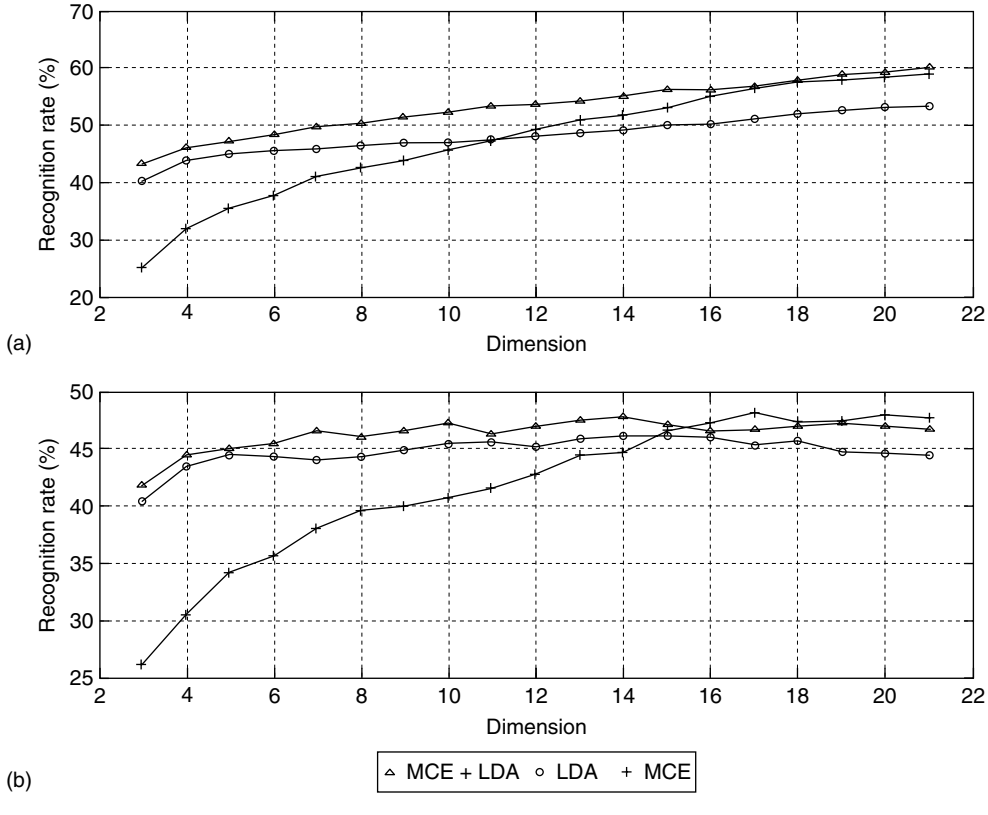



**Figure 16.11** Results of MCE(UNIT), GMCE + LDA and LDA on the TIMIT database. (a) Training data; (b) testing data.

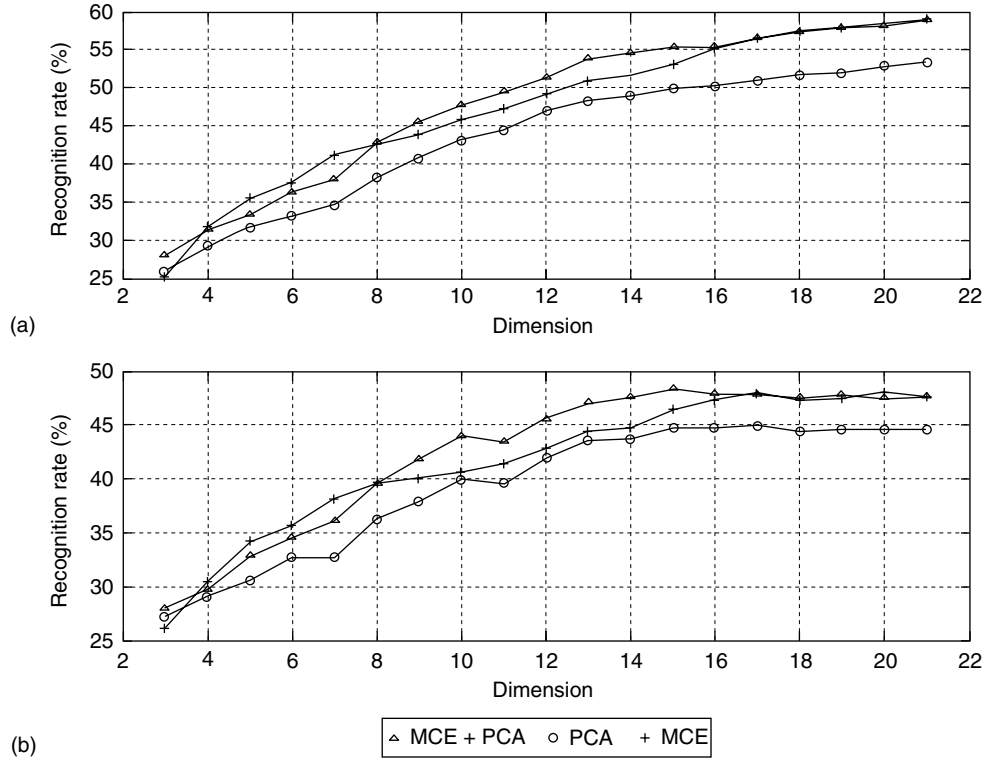



**Figure 16.12** Results of MCE(UNIT), GMCE + PCA and PCA on the TIMIT database. (a) Training data; (b) testing data.

- In high-dimensional feature spaces (dimensions 15–21), the performances of GMCE + LDA are close to those of the MCE training algorithm, which are better than LDA.
- In medium-dimensional (dimensions 7–15) and low-dimensional (dimensions 3–7) feature spaces, GMCE + LDA has significantly better performances than both LDA and MCE.

### 6.2.3 Analysis of RDSVM

Figure 16.13 compares the results of RDSVM to those of GMCE + LDA, LDA and SVM. Observations from the results can be summarized as follows:

- Compared to SVM, the performance of RDSVM on the full-dimensional feature space is improved on training data. RDSVM's performance on testing data (on the full-dimensional feature space) is also improved on some subdirectories and remains the same on the rest.
- The performance of RDSVM on training data is poorer than that of GMCE + LDA and LDA in both medium- and high-dimensional feature spaces (dimensions 12–21). In very low-dimensional feature spaces (dimensions 3 and 4), RDSVM performs better on training data than LDA and GMCE + LDA. On the other dimensions, the performance of RDSVM is between that of GMCE + LDA and LDA.
- The general performance of RDSVM on testing data is much better than that of GMCE + LDA and LDA on all dimensions. In some subdirectories, the recognition rates of RDSVM are over 5 % ahead of those of GMCE + LDA on average on all dimensions.

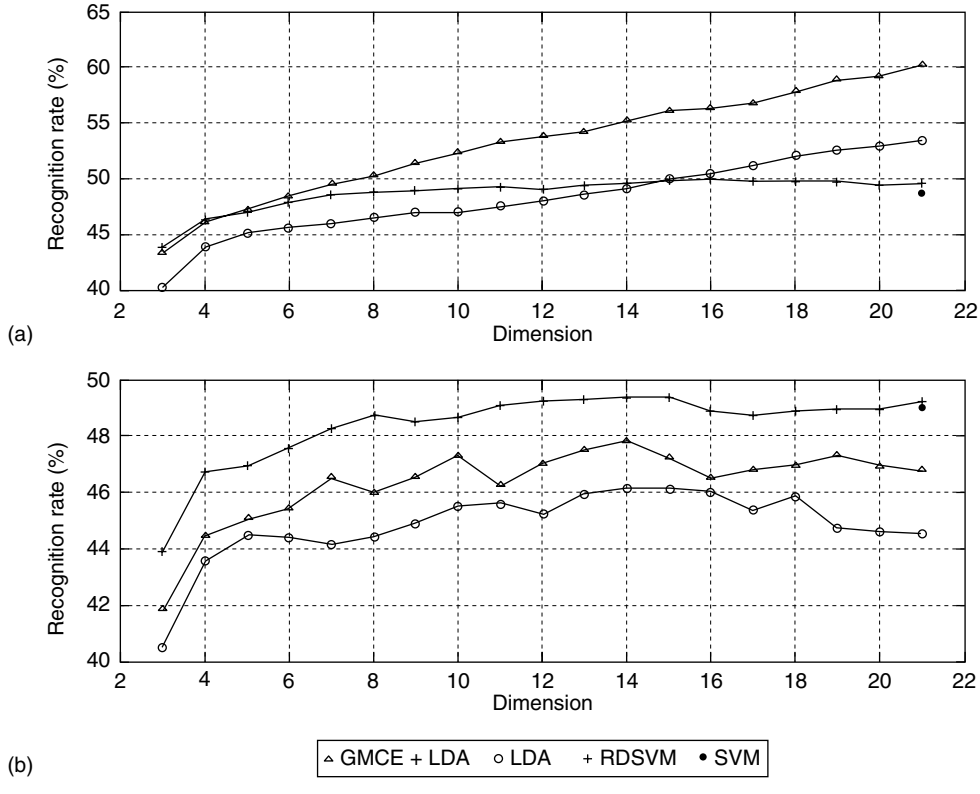



**Figure 16.13** Results of GMCE + LDA, LDA, SVM and RDSVM on the TIMIT database. (a) Training data in DR2; (b) testing data in DR2.

- The performance of RDSVM is very stable throughout all the dimensions on both training and testing data. The performance of RDSVM usually starts degrading only when the feature dimension is less than five.
- The performance curves of RDSVM on the training data of all the eight subdirectories are fairly smooth, as are those of GMCE + LDA and LDA. But the performance curves of RDSVM on the testing data are not as smooth as those on the training data.

## 7. Conclusions

In this chapter, we investigated major feature extraction and classification algorithms. Six algorithms were involved in the investigation; they are LDA, PCA, the MCE training algorithm, SVM, the GMCE training algorithm and RDSVM. From the observation of the experimental results, the following conclusions can be drawn:

1. *Feature extraction*. The two independent feature extraction algorithms, LDA and PCA, have close performances. But LDA is more stable in low-dimensional feature spaces. Compared to LDA and PCA, the integrated feature extraction and classification algorithms, MCE and GMCE training algorithms, have generally better performances. The MCE training algorithm performs better than

LDA and PCA in high-dimensional feature spaces. But its performance degrades rapidly with the decrease of feature dimensionality. The GMCE training algorithm integrates the advantages of both LDA and the MCE training algorithm and has superiority over other algorithms on all dimensions. The experiment results show that in the GMCE training algorithm, the LDA initialization criterion is better than the PCA initialization criterion.

2. *Classification*. The classification results show that the discriminative classifiers, MCE and GMCE training algorithms, have better fitness to the training data, while the nonlinear classifiers, SVM and RDSVM, have better generalization properties than discriminative classification algorithms.

## References


[1] Hotelling, M. "Analysis of a Complex of Statistical Variables into Principal Components," *Journal of Educational Psychology*, **24** pp. 498–520, 1933.

[2] Jolliffe, I. T. *Principal Component Analysis*, Springer Verlag, New York, 1986.

[3] Rao, C. R. "The use and interpretation of principal component analysis in applied research," *Sankhya A*, **26**, pp. 329–358, 1964.

[4] Bocchieri, E. L. and Wilpon, J. G. "Discriminative analysis for feature reduction in automatic speech recognition," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, **1**, pp. 501–504, 1992.

[5] Poston, W. L. and Marchette, D. J. "Recursive Dimensionality Reduction Using Fisher"s Linear Discriminant," *Pattern Recognition*, **31**(7), pp. 881–888, 1998.

[6] Sun, D. X. "Feature Dimension Reduction Using Reduced-Rank Maximum Likelihood Estimation For Hidden Markov Model," *Proceedings of International Conference on Spoken Language Processing*, Philadelphia, USA, pp. 244–247, 1996.

[7] Juang, B. H. and Katagiri, S. "Discriminative Learning for Minimum Error Classification," *IEEE Transactions on Signal Processing*, **40**(12), pp. 3043–3054, 1992.

[8] Katagiri, S., Lee, C. H. and Juang, B. H. "*A Generalized Probabilistic Descent Method*," *Proceedings of the Acoustic Society of Japan*, Fall Meeting, pp. 141–142, 1990.

[9] Paliwal, K. K., Bacchiani, M. and Sagisaka, Y. "Simultaneous Design of Feature Extractor and Pattern Classifier Using the Minimum Classification Error Training Algorithm," *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*, Boston, USA, pp. 67–76, 1995.

[10] Chang, P. C., Chen, S. H. and Juang, B. H. "Discriminative Analysis of Distortion Sequences in Speech Recognition," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, **1**, pp. 549–552, 1991.

[11] Komori, I. and Katagiri, S. "GPD Training of Dynamic Programming-based Speech Recognizer," *Journal of Acoustical Society of Japan(E)*, **13**(6), pp. 341–349, 1992.

[12] Chou, W., Juang, B. H. and Lee, C. H. "Segmental GPD Training of HMM-based Speech Recognizer," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, **1**, pp. 473–476, 1992.

[13] Liu, C. S., Lee, C. H., Chou, W. and Juang, B. H. "A Study on Minimum Error Discriminative Training for Speaker Recognition," *Journal of Acoustical Society of America*, **97**(1), pp. 637–648, 1995.

[14] Rainton, D. and Sagayama, S. "Minimun Error Classification Training of HMMs–Implementation Details and Experimental Results," *Journal of Acoustical Society of Japan(E)*, **13**(6), pp. 379–387, 1992.

[15] Boser, B. E., Guyon, I. M. and Vapnik, V. "A training algorithm for optimal margin classifiers," in Haussler, D. (Ed) *5th Annual ACM Workshop on COLT*, Pittsburgh, PA, pp. 144–152, 1992.

[16] Roth, V. and Steinhage, V. *Nonlinear discriminant analysis using kernel functions*, Technical report Nr IAI-TR-99-7, University of Bonn, 1999.

[17] Vapnik, V. *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[18] Joachims, T. "Making large-scale SVM learning practical," in Scholkopf, B. Burges, C. J. C. and Smola, A. J. (Eds) *Advancess in Kernel Methods – Support Vector Learning*, MIT Press, Cambridge, USA, pp. 169–184, 1998.

[19] Scholkopf, B., Burges, C. and Vapnik, V. "Incorporating invariances in support vector learning machines," *International Conference on Artificial Neural Networks – ICANN'96*, Berlin, pp. 47–52, 1996.

[20] Scholkopf, B., Bartlett, P., Smola, A. and Williamson, R. "Support vector regression with automatic accuracy control," *Proceedings of the 8th International Conference on Artificial Neural Netwoks, Perspectives in Neural Computing*, Berlin, pp. 111–116, 1998.

[21] Smola, A. J. and Scholkopf, B. *A tutorial on support vector regression*, NeuroCOLT2 Technical Report Series NC2-TR-1998-030, ESPRIT working group on Neural and Computational Learning Theory 'NeuroCOLT 2', 1998.

[22] Duda, R. O. and Hart, P. E. *Pattern Classification and Scene Analysis*, John Wiley & Sons, Inc., New York, 1973.

[23] Cottrell, G. W. "Principal Components Analysis of Images via Back Propagation," *SPIE Proceedings in Visual Communication and Image Processing*, **1001**, pp. 1070–1077, 1988.

[24] Burges, C. J. C. "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, **2**(2), pp. 955–974, 1998.

[25] Vanderbei, R. J. "LOQO: An interior point code for quadratic programming," *Optimization Methods and Software*, **11**, pp. 451–484, 1999.

[26] Clarkson, P. and Moreno, P. J. "On the use of Support Vector Machines for Phonetic Classification," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing'99*, pp 585–588, 1999.

[27] Scholkopf, B., Burges, C. and Vapnik, V. "Extracting support data for a given task," *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, Menlo Park, pp. 252–257, 1995.

[28] Robinson, T. *Dynamic Error Propogation Networks*, PhD Thesis, Cambridge University Engineering Department, February 1989.

[29] Fisher, W., Doddington, G. and Goudie-Mashall, K. "The DARPA speech recognition research database: specifications and status," *Proceedings DARPA Speech Recognition Workshop*, pp. 93–99, 1986.

[30] Sankar, A. and Mammone, R. "Neural tree networks," in Mammone, R. and Zeevi, Y. (Eds) *Neural Networks: Theory and Applications* Academic Press, pp. 281–302, 1991.

[31] Tsoi, A. C. and Pearson, T. "Comparison of the three classification techniques, CART, C4.5 and multi-layer perceptrons," in Lippman, R. L. Moody, J. E. and Touretzky, D. S. (Eds) *Advances in Neural Information Processing Systems 3*, Morgan Kaufmann, San Mateo, CA pp. 963–969, 1990.

# 17

# Improving Mine Recognition through Processing and Dempster–Shafer Fusion of Multisensor Data

**Nada Milisavljević**
Signal and Image Centre, Royal Military Academy, Brussels, Belgium

**Isabelle Bloch**
Department TSI – CNRS UMR 5141 LTCI,
Ecole Nationale Supérieure des Télécommunications, Paris, France

*We propose in this chapter a fusion approach for improving anti-personnel mine recognition. Based on three promising and complementary mine detection sensors, we first address the issue of extraction of meaningful measures, which are then modeled as belief functions and combined within a Dempster–Shafer framework. A starting point in the analysis is a preprocessed multisensor data set containing some mines and false alarms. A method for detecting suspected areas is developed for each of the sensors. A detailed analysis of the chosen measures is performed for the selected regions. A way of including the influence of various factors on sensors in the model is presented, as well as the possibility that not all sensors refer to the same object. For every selected region, masses assigned by each of the measures and each of the sensors are combined, leading to a first guess on whether there is a mine or a non- dangerous object. An original decision rule adapted to this type of application is described too.*

## 1. Introduction

For decades, great efforts of research centers all over the world have been made in the field of humanitarian mine detection. Unfortunately, this problem is still unsolved, mainly due to the necessarily high detection rate that is required, as well as the large variety of types of mine and of scenarios

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

where mines can be found. Consequently, there is no single sensor that works well enough in all the possible scenarios. A conventional Metal Detector (MD) is the oldest mine detection sensor, and in reality, it is still the one that is most often used by the deminers. Put simply, an MD detects metal. Consequently, there are situations where it cannot be used, due to either soil type (ferrous soils), metallic debris, that often remain on old battlefields, or, nowadays, plastic or low-metal content mines, for which an MD is practically useless. In these cases, other sensors are preferable, and one most often reaches for Ground-Penetrating Radar (GPR). GPR detects any object below the soil surface if it differs from the surrounding medium [1,2] in: the conductivity (metallic targets), the permittivity or the dielectric constant (plastic and nonconducting targets), or the permeability (ferrous metals). However, conventional GPR has problems in detecting shallow-buried mines due to a strong air/ground interface reflection that hides the objects placed in that layer. This is one of the situations where the role of an Infrared (IR) camera becomes important. Therefore, one of the most promising sensor combinations consists of an MD, GPR and IR. We propose a method of combination that can easily be adapted for other sensors and their combinations.

In this chapter, methods for analyzing GPR, MD and IR data are described, as part of work done within the Belgian HUDEM project [3]. In Section 2, GPR, MD and IR data presentation is briefly described, and their preprocessing is presented. Taking into account some characteristics of the GPR, MD and IR data in general, as well as the way that the preprocessing is done, a method for selecting suspected regions for each of the sensors is proposed in Section 3. Once the possibly dangerous regions are selected, a detailed analysis of each of the regions is performed, in order to extract measures that can give information about the true nature of the alarm. Section 4 discusses measures extracted from the preprocessed MD, IR and GPR data. After that, as shown in Section 5, each extracted piece of information is modeled in terms of belief functions, and discounting factors are introduced. In Section 6, the possibility that not all the sensors speak about the same object is discussed and modeled, and discounted measures are then combined within the Dempster–Shafer (DS) framework [4,5]. Since the combination rule is associative and commutative, the combination of measures is done first per sensor, in order to obtain the decision of each of the sensors separately, and then all the sensors are combined. The nonprobabilistic interpretation [6] of the DS method is chosen, on the one hand, to compensate for the fact that the data are not numerous enough for a reliable statistical learning, and on the other hand, so as to be able to easily include and model existing partial knowledge about the mines and the sensors. A new decision rule is proposed, leading to a simple way of making decisions or guesses about the true identity of each region. Finally, results obtained by applying the proposed method on multisensor data acquired in a sand lane at the TNO test facilities [7] within the Dutch HOM-2000 project are given and discussed in Section 7.

To our knowledge, there have been just two attempts at applying DS theory to this problem [8,9]. Both works treat an alarm as a mine, and not as an object that could be a mine, as well as a false alarm. In addition, in [8], beliefs are obtained classically, as probabilities [10], so this method faces the same problem as statistical ones: the lack of a sufficient amount of training data. In [9], the efficiency of the proposed method is estimated using real data containing a few mines and no placed false alarms, meaning that the obtained results do not show how the method deals with false alarms.

## 2. Data Presentation and Preprocessing

### *2.1 IR Data*

Our concern is only a passive IR sensor, which observes and measures heat without being in contact with the source of it. Therefore, the principle of IR mine detection is very simple. Objects (in our case mines), having different materials and thus different thermal characteristics than their surroundings, can be detected by IR sensors under some conditions. Namely, mines become hotter and colder faster than their surroundings, meaning that the daily evolution of IR mine signatures can be observed.

Consequently, depending on the time of the day, a mine can appear in an IR image as lighter (warmer) than its surroundings, as darker (colder), or it cannot be noticed due to having reached thermal equilibrium with the surroundings. In addition, the IR contrast between an object and its surroundings decreases strongly with the depth, so an IR sensor is most often used for direct detection of surface-laid mines. Indirect detection of buried mines is either due to the surface effect (disturbed soil due to recent burial of mines) or due to the volume effect (the presence of a long-buried mine changes the heat and water flow conditions above the mine) [11].

Since the IR images result from simple and known operational principles and the images are almost purely representing radiated energy, some classical noise reduction tools are applied in the preprocessing step [11]. Note that most of the mines are thin cylinders and that in general, due to some burial angle, they appear as ellipses in the IR images.

## *2.2 GPR Data*

Earth materials are mostly nonmagnetic and the change in conductivity mainly affects absorption of the GPR signal by the medium. Thus, it is usually the permittivity contrast that leads to a reflection of the electromagnetic waves radiated by the transmit antenna of the GPR and the detection of backscattered echoes by its receiving antenna.

### 2.2.1 Types of GPR Data Presentation

A common presentation of the signals obtained by GPR is in the form of scans: A-, B- and C-scans (adopted from acoustic terminology), and data processing can be applied to any of them. A single amplitude–time waveform, with the GPR antennas at a given fixed position, is referred to as an A-scan. A B-scan presents an ensemble of A-scans gathered along one axis, or, in other words, it is a two-dimensional (2D) image representing a vertical slice in the ground. Note that reflections from a point scatterer located below the surface are present in a broad region of a B-scan, due to the poor directivity of the transmitting and the receiving antenna. Finally, a C-scan is a three-dimensional (3D) data set resulting from collecting multiple parallel B-scans, hence recording the data over a regular grid in the soil surface plane. Usually, a C-scan is represented as a horizontal slice of this 3D data set by plotting its amplitudes at a given time.

### 2.2.2 A-scan Preprocessing Methods

In order to use the information on energy contained in an A-scan, the weakening of the signal with the depth should be compensated. Another problem that should be overcome is the strong reflection at the air/ground interface, that, besides strongly biasing the energy contained in each A-scan, often hides reflections from objects buried just below or placed just above the surface.

While propagating from the transmitter towards a buried object and being scattered back to the receiver, the electromagnetic waves of the GPR are subject to some losses [12,13]. In particular, the deeper the object is buried, the higher the losses introduced by the soil are. In order to compensate for these attenuations in function of $R$ or, more directly, of time $t$, a Time-Varying Gain (TVG) is introduced, by which a fixed gain of $X$ dB per second (or per meter) is added to the raw signal $s$, so the amplified signal in the time domain is:

$$s_{\mathrm{TVG}}(t) = s(t) \cdot 10^{\frac{Xt}{20}} \tag{17.1}$$

The optimal value of TVG strongly depends on the type of GPR used, on the type and characteristics of the soil, on the level of moisture and on the depth range of interest, so it is usually chosen to meet the operational requirements. Its values can typically vary from 0.1 dB/ns up to 100 dB/ns.

Regarding the background removal method (also called clutter reduction [12]), it aims to reduce clutter and eliminate strong air/ground interface reflections. Within regions where the soil surface is not rough, where the electromagnetic properties of the soil are unchanged and where the antenna distance from the ground is kept constant, it can be expected that the position at which these strong reflections occur remains constant. In addition, in such regions it can be assumed that some background disturbances affect the neighboring A-scans approximately equally. This is often the case for smaller regions. A simple way for removing the background consists then of choosing a window within which, for each sample number, the mean of neighboring A-scans is found and subtracted from the value of a central A-scan at that sample number. Usually, this sliding window within which the mean is calculated is in the direction of scanning:

$$s_{\text{new}}(x, y, t) = s_{\text{TVG}}(x, y, t) - \frac{1}{2n+1} \cdot \sum_{i=-n}^{n} s_{\text{TVG}}(x, y-i, t) \tag{17.2}$$

where $s_{\text{TVG}}$ is a signal after the TVG and before background removal, $s_{\text{new}}$ is the signal after background removal, $x$ is the cross-track coordinate, $y$ is the scanning direction coordinate and $n$ is the half-width of the sliding window. The choice of $n$ should be a compromise between the expected size of the objects, the distance between two neighboring objects, and the possible change of the height of the antennas. If the range of expected sizes of objects and their distances is not known, if surfaces are rough or if the height of the antennas is not constant, this type of background removal is not really useful. An ideal solution would be to have a 'known-to-be-empty' region for the background estimation, for which the GPR distance from the soil surface is somehow (if possible) kept the same as for the measuring region, but this request is rarely met in reality. Finally, if neither of the above requirements for background estimation by finding the mean of several neighboring A-scans is met, more complicated ways for background removal would have to be investigated.

### 2.2.3 A C-scan Containing Energy Projections of Preprocessed A-scans

The advantage of GPR in providing 3D information can be a drawback too, in terms of data quantity. In order to reduce the amount of GPR data, we choose to project the information contained in A-scans in one plane. Another important reason for creating a unique C-scan from 3D data is in using it later both to select possibly dangerous regions and to fuse it with other sensors that give 2D images of the regions, such as an IR or an MD [11]. Several possibilities for creating such C-scans as projections of A-scans exist, of which the most useful is to sum square values of each A-scan, i.e. to represent it by its real energy. Namely, once TVG and background removal are applied to the data, the signal attenuation effects of the depth and the strength of the air/ground interface reflection are suppressed to some level. It allows us to project the energy contained in one A-scan in one point, as well as to put together all such points in one plane and by that indirectly induce comparison between different A-scans, belonging either to objects at various depths or simply to the background. In other words, each A-scan $s_{\text{new}}(x, y, j\Delta T)$, where $j$ is sample number, and $\Delta T$ is the sampling time, is represented by its energy $E_s(x, y)$:

$$E_s(x, y) = \sum_{j=1}^{N} s_{\text{new}}^2(x, y, j\Delta T) \tag{17.3}$$

Since the peak energy is proportional, amongst others, to the volume of the object seen from that point, the resulting C-scan is a good start for selection of regions possibly containing (dangerous) objects.

## 2.3 MD Data

The detection principle of an MD is based on inducing electric currents in a conducting object through exciting it by an electromagnetic field. If the frequency of the excitation is low (audio band), the induced currents are able to penetrate deeply into the object. These eddy currents then produce a secondary electromagnetic field, detected by a secondary coil of the MD. A traditional MD has just a sonar signal for indicating the presence of metallic objects. These days, it is possible to convert its signal to an image, and this type of MD is under our consideration.

The image provided by an imaging MD is a result of visualization of sonar signals gathered through a two-dimensional scan. Such an image is blurred due to the fact that both primary and secondary magnetic fields spread as bundles, as well as due to the fact that the footprint of the coil is large when compared to the size of metallic objects such as AP mines. That is why deconvolution is an important aspect in preprocessing of the imaging MD data.

### 2.3.1 Deconvolution

The MD response of an object, $f(x, y)$, is convolved with the Point-Spread Function (PSF) of the MD, $h(x, y)$ ($x$ and $y$ being, again, pixel coordinates in the image), resulting in a blurred image, $i(x, y)$, given by the following degradation model:

$$i(x, y) = f(x, y)^* h(x, y) + n(x, y) \tag{17.4}$$

Here, $n(x, y)$ represents noise, and * stands for the 2D discrete linear convolution operator. Various techniques of restoring the image exist [14], and one of the most widely used is the Wiener filter. It minimizes the mean square error between the estimate of an image and the true image, using a linear operator, under assumptions of signal-independent noise, linear degradation, stationarity of the image and degradation filter, and 2D-periodic approximation of the PSF. For our application, a main problem of Wiener filtering is that the PSF of the MD has to be known. It can be found either theoretically (by developing a mathematical model of such a system) or experimentally (by acquiring the response of the MD on a very small metallic ball that can be, in a first approximation, treated as a metallic point). Unfortunately, the impulse response of the MD depends on various factors, amongst which is the distance between the detector and the observed point. It limits the applicability of standard deconvolution techniques that need to have the PSF as an input to the cases where the burial depth of the metallic object under consideration is known. This is a great problem for real mine detection situations. Some advanced deconvolution techniques, such as (iterative) blind deconvolution [15,16,17], aim at overcoming such a problem. The idea of the method is to estimate both the true image and the PSF by first randomly initializing their estimates. Based on that, a first estimate of the PSF is found using input on the energy of the additive noise and then imposing blur constraints regarding the size of the PSF. In a next step, a first estimate of the true image is calculated imposing image constraints (that the image has a final support). The procedure is then iterated some preset number of times or until the two estimates start to converge. Our results have shown that final results depend on the initializing estimates, that both estimates converge slowly, that the PSF estimate converges even more slowly than the true image estimate, and that the method is unstable and lacks reliability (as also pointed out in [15]).

Thus, iterative blind deconvolution is not well-suited for our case due to its dependence on starting estimates in the first place. On the other hand, though Wiener filtering is more reliable and faster, it needs an adequate PSF as its input. A potential way to go could be to use measured PSFs of the MD as a function of the depth and try to deconvolve (using Wiener filtering) a recorded blurred image using each of the PSFs until the one is found for which the sharpest image is obtained. By that, both the true image and the depth (by knowing which depth such a PSF corresponds to) could be estimated.

Unfortunately, in most of the MD applications in mine detection, the data has too poor resolution in one of the directions, and is saturated as well. Therefore, in the following, we use the raw data and keep in mind its physical meaning described above.

## 3. Region Selection

In this section, for each of the sensors we propose possible ways of alarming, i.e. of finding regions possibly containing mines, that should be further analyzed.

### *3.1 IR Regions*

In the case of IR, alarming has to take into account the possibility of having an inversion of the contrast, i.e. the possibility that a mine can appear both darker and lighter than its surroundings, depending on the time of day. Furthermore, such a method has also to be able to deal with various levels, i.e. strengths of contrast between a mine and the background, again depending on the time of day, mine material, type of soil, etc.

One of the simplest ways for determining regions of interest is certainly by thresholding a preprocessed IR image, but this involves a visual assessment of the best threshold level for a particular scenario. Since we look for some automatic solutions, a way for estimating background should be involved and then pixels that exceed significantly the estimated background would be preserved. There are different ways for estimating the background. In the case of human intervention in the field, this would be to find a 'known-to-be-empty' region near to the analyzed area, in order to be sure that their backgrounds are similar. If such a region does not exist, the simplest and quite safe solution is to estimate the background for each image separately, from some preset number of the bordering pixels [18] around an IR image, small enough not to include pixels belonging to a mine, and still large enough to be a statistically reliable estimation. In the case of daylight measurements, the thermal clutter is quite strong, meaning that the background can significantly change from one part of the investigated region to another, i.e. from one IR image to another, which makes this idea even more attractive and realistic. This method is adopted here. Furthermore, a method should be determined for actually estimating the background and using that estimation for selecting pixels of the image that differ from the background (enough to suspect that they signal the presence of an object in the scene), and by that obtaining a binary image of the scene. Again, several possibilities exist, amongst which three seem both simple and realistic:

1. To find the mean of the background area, $m_{\text{back}}$, and to preserve only pixels that, by absolute value (in order to deal with contrast inversions), exceed the background mean for at least some factor $f_1$, i.e. to keep the pixels the values of which, $v$, satisfy the condition:

$$|v - m_{\text{back}}| > f_1 m_{\text{back}} \tag{17.5}$$

2. To go a step further by finding the standard deviation of the background as well, $s_{\text{back}}$, and keep only the pixels that fulfill this condition:

$$|v - m_{\text{back}}| > f_2 s_{\text{back}} \tag{17.6}$$

where $f_2$ is again some factor that can be varied, depending on the situation (whether we have information that the contrast is high or low, i.e. whether mines are mostly buried or surface-laid). This condition is based on the assumption that the noise is of a Gaussian type, and it is the one used in [18].

3. One more way is to determine the maximum value of the noise in the background, $n_{\max}$, as the maximum difference between the pixels in that region and their mean, and preserve just those pixels that satisfy the condition (with $f_3$ being yet another factor that can be varied as a function of the scenario):

$$|v - m_{\text{back}}| > f_3 n_{\max} \tag{17.7}$$

Our tests have shown that all three methods lead to similar results. In the following, we use the second one, and the value of $f_2$ is chosen to be 3 as a compromise between being able to preserve responses of shallowly buried targets without increasing significantly the clutter in the case of surface-laid objects.

## 3.2 GPR Regions

A simple means of region selection is to threshold projected A-scans. This gives good results, even without TVG [19]. Still, a potential general problem introduced by this method is that deeper buried objects have weaker signals and, accordingly, weaker energies. Therefore, it can happen that they are not detected if the threshold is not low enough, while a threshold that is too low leads to an increase in the number of false alarms. Even if TVG is applied, such a simple thresholding can rarely work well enough in case of humanitarian mine detection, which demands the highest possible detection rates. As shown in the previous section, the problem of correctly choosing the appropriate TVG is subtle, and strongly depends on a variety of factors. Consequently, it is practically impossible to be sure the chosen TVG is the right one.

In order to preserve weaker signals, possibly belonging to deeper buried objects, without causing a strong increase in the number of false alarms, we propose a simple method for the region selection. The idea is to find local maxima and analyze their neighborhoods by grouping together all the points within some window around each of them, the value of which is close to the value of that local maximum. As a result, we get a 'blob' within a window around every local maximum. Note that a blob belonging to one local maximum actually can be a group of blobs, and not just one connected blob. Namely, within a window around a local maximum there could be some regions with higher, and some with lower, energies than the chosen threshold (on percentage of the maximum decrease of a local maximum value) allows.

This means the local maxima method needs three pieces of information as input. The first one is the size of the window around the local maxima within which the points are analyzed. Again (see the previous section), it must be larger than the size of the largest expectable object convolved with the GPR antenna opening and smaller than the minimum expectable distance between two objects in a lane. If this interval does not exist, mistakes can hardly be avoided – either an object that is too large could be detected twice, or a few closely placed objects could be detected as a single object. A way to deal with this problem is mentioned in Section 7. A second input is the minimum value of local maxima that should still be detected ($thr_1$). Obviously, this value affects the number of detected regions. Finally, the minimum percentage of the local maximum value so that neighboring pixels are grouped with it ($thr_2$) has to be chosen too. The choice of this value should not be critical, since it should influence sizes of all obtained blobs similarly.

## 3.3 MD Regions

Depth estimation through deconvolution, discussed in the previous section, asks for good quality images, i.e. with a good resolution. Thus, this method is feasible in cases of small images of single targets, and that diminishes its usefulness for the depth estimation in cases of region selection on images of broader areas. To our knowledge, there is no other way for determining the depth of an object in case of imaging MD data. Since the strength of the response of a metallic object depends on its depth,

it further means that in the case of region selection, great care has to be taken not to discard weaker signals possibly corresponding to deeper buried metallic objects. In other words, if simple thresholding is applied for region selection, it should be low enough to detect such signals and objects. However, larger metallic objects leave strong and large, often saturated, signals in quite a broad area around them (due to the fact that deconvolution cannot be performed, so their strong response is convolved with quite a large footprint of the MD). A low threshold means that such objects would be presented on images by huge blobs covering large areas and sometimes even hiding responses of smaller metallic objects in their neighborhood. Finally, such strong and wide responses of neighboring objects are often joined. Therefore, standard thresholding is not well suited here.

### 3.3.1 MD Region Selection on Saturated Images by the Local Maxima Method with Windows

An important difference between the local maxima method introduced in the previous subsection for GPR images and the one for MD data is in the fact that in the latter case, a general danger of having saturated images exists, which should be taken into account. Namely, for GPR data, a first local maximum that is found is immediately selected as the center of the corresponding region, the neighborhood of which is to be analyzed and possibly grouped with it, but this approach is not well-suited for saturated MD images. In the case of saturation, a local maximum found in such a way would always be somewhere at the borders of saturated regions, i.e. uniformly-valued areas, meaning that a bordering pixel of the uniform region would be chosen as a center of a window.

Therefore, the method has to be adjusted for saturated images. The modification is as follows: once a local maximum is found, its neighborhood is checked. The checking stops either when the pixel value starts to decrease, or when the distance from the starting point reaches the half-size of the window, depending on what happens first. (Note that the criterion is the half-size and not the full size of the window, in order to minimize mistakes. Otherwise, such a window would contain only uniform maxima values, so there would later still be false alarms in regions next to it, with slightly smaller values than these maxima.) Then, the middle pixel of the found range with a uniform pixel value is chosen as the real local maximum and as the center of a region. After that, the procedure is the same as for the GPR. The influence of $thr_1$ on the number of blobs and the influence of $thr_2$ on their size, are similar to those explained in the previous subsection. Regarding the choice of the size of the window, the reasoning remains the same as for the GPR data.

The choice of the size of the window is very important in avoiding artifacts, but it should not be forgotten that this approach can be quite dangerous in reality if we do not have precise information about the expected sizes of objects and the distances between them. A too large size of the window can cause grouping of two objects together, as one alarm. Thus, in real applications, one should keep in mind the size of the window chosen, and once a region is selected, one should remember which area around the local maximum (which size of the window) is being considered while alarming, i.e. which area really corresponds to a selected region.

### 3.3.2 MD Region Selection on Saturated Images by the Local Maxima Method without Windows

When knowledge on the size of objects is available, the above method can be applied, and it will result in a good division of one blob belonging to several objects. In general, if such knowledge does not exist or is not completely reliable, we cannot be sure whether one large blob is a result of several objects or of only one large metallic object. Consequently, a safer way to go is to put no restrictions on the size of the window, i.e. not to have windows at all. This means that the no-window method searches for a local maximum. When it finds one, it follows its value and observes its neighborhood

as long as a decrease in the value does not occur. Then, the center of this uniform-valued region is the central local maximum and all the pixels around it are grouped with it as long as their value does not decrease below that defined by $thr_2$. The procedure continues as long as there are pixels with values above $thr_1$. Still, some small artifacts on the borders of a found region can remain, the values of which are too small to be grouped with that region but still larger than $thr_1$, so that afterwards, these artifacts can become centers of some false regions. Thus, we introduce the following: a local maximum region is accepted as a potential suspicious region only if around its local maximum (within some resolution circle, the radius of which, $r$, can be around or less than one half of the resolution, i.e. of an MD coil radius) there are no larger values than it in the starting image. Otherwise, this region is discarded since it can be either a remaining part of some already detected region or simply noise.

Note that the above modification can be introduced here since there are no restrictions on sizes of windows, so it can be assumed that these are really artifacts. In the case of the previous local maxima method for saturated images with windows, the underlying assumption is that there is some knowledge on the size of objects. Thus, such a modification could be dangerous and that is why it is not included there.

This method is good and safe for region selection in cases when no sure information exists on expectable sizes of objects and their arrangement (distance).

# 4. Choice of Measures and Their Extraction

## *4.1 IR Measures*

An IR provides 2D images and thus information about the shape and size of objects. Since most AP landmines are thin cylinders, they appear as ellipses on IR images, taking into account the burial angle, i.e. that mines are not buried ideally parallel to the ground surface. Based on the binary images obtained by selecting regions as described in the previous section, our idea is to apply classical edge detection on each of these images and perform shape analysis on binary, edge-detected images. Working on edges is chosen as a way to speed up the analysis. The continuity of these edges is destroyed by several factors [20], such as noise and natural phenomena in the scene itself, leading to the real problem investigated here: detection of partially occluded ellipses on binary images. In [11], three methods are analyzed: a nonparametric approach (see [21,22]), chain analysis (our contribution introduced in [23], which is a sort of border tracing [24] or chain coding [14]) and our modifications [23] of randomized Hough transforms for ellipses [25,26,27]. In the following, we describe only the third method since it is the one that we use here.

### 4.1.1 Randomized Hough Transform (RHT) and Ellipticity Measure

This method does not preserve the original shape of an edge. As with any Hough transform [28–30], it is necessary to decide which shape to detect. In some cases this can be a problem, but not here, as long as mines are circular (viewed as elliptical). In the case of ellipse detection, a triplet (based on the ideas given in [25,31]) of foreground pixels $(P_1, P_2, P_3)$ is randomly chosen, and, by analyzing their coordinates and their neighborhood, parameters of the ellipse that contains these three pixels are found as follows:

- first, by looking at a small neighborhood of each of the three pixels, the tangents of a potential ellipse in these three pixels are determined as lines through each of the pixels and their neighbors;
- the middle lines of the angles between two pairs of tangents are found; the intersection of the two lines represents the position of the ellipse center, $C(x_c, y_c)$;
- then, the remaining three parameters of the ellipse, $a$, $b$ and $c$, can be estimated by replacing coordinates of each of the three pixels of the triplet in the ellipse equation: $a(x-x_c)^2 + 2b(x-x_c)(y-y_c) + c(y-y_c)^2 = 1$, and solving the system of the three obtained equations.

This step is repeated some preset number of times and parameters of found ellipses are stored. Hough transforms, in general, are time consuming, but this randomized version overcomes that problem. We propose one more way to make it even faster. Namely, instead of choosing, as the final result, ellipses that are visited the highest number of times (which demands a high enough preset number of times of looking for ellipses to make results valid), the number of times the procedure is repeated can be decreased to some extent, taking into account that ellipses can be missed if this number is too low. After that, the image should be rechecked and foreground pixels lying on each of the ellipses counted. Ellipses that contain enough foreground pixels and that are within some dimension interval may be chosen as the best representatives of detected elliptical objects on an image. This way, calculation time can be spared and results remain representative. Another advantage of this method is that it does not need good preprocessing and it does not depend on continuity of edges.

The first chosen measure is, consequently, shape ellipticity (how well the shape of the selected region fits in an ellipse obtained using the randomized Hough transform method described here).

### 4.1.2 The Second Measure – Elongation

This measure tells how elongated the shape is. The method applies to the binary image of a selected region. We calculate the center of gravity of the image and then we find:

- the minimum and maximum distance of bordering pixels from the center of gravity, and the ratio between them (*ratio*1) (this works well only if the center of gravity lies inside the boundary, otherwise it does not make sense, so checking whether the center of gravity is inside is done before this step; if it is not inside the boundary, *ratio*1 is set to 0); note that only the outer boundaries (edges) of the shape are taken into account here;
- second moments, and from them, the ratio of the minor and major axes of the obtained quadratic form (*ratio*2); this can be applied on edges as well, but the influence of the noise on the position of the center of gravity is much larger in that case.

In the later section on mass assignment, the two measures will be used.

### 4.1.3 The Third Measure – Area/Size

The reasoning behind choosing this measure is the following. When preliminary information about the expected size of mines exists (e.g. on the basis of knowledge about the types of mine laid in some particular region), a range of areas of detected object that could be a mine, but something else as well, can always be predicted, taking into account again possible deformations because of some burial angle. Outside that range, it is much more likely that objects are not mines.

## *4.2 GPR Measures*

### 4.2.1 Choice of A-scan and C-scan Measures and Their Extraction

The selection of regions that possibly contain mines is usually the end of the detection process, resulting in a lot of false alarms in reality. Therefore, we are looking for some useful and reliable measures that can be relatively easily extracted from GPR data, and that should give more information regarding the true identity of objects within each of these regions.

The acquisition step in the cross-track direction is often around the size of a typical AP mine. Consequently, depending on its position in comparison with the gathered B-scans, the same mine can appear on one, two or, in a limit case (taking into account the finite antenna opening, i.e. the fact that antennas typically have a poor directivity, so objects appear larger than they really are), even

three successive B-scans. In such cases, both the size and shape of the 2D projection are useless. In order to overcome this problem and still extract some information from the preprocessed C-scan, two assumptions have to be made. The first one is that the objects are not prolonged in one direction, so that the dimension in the scanning direction can give an idea about the object size. Since GPR antennas are usually close to the ground during the acquisition, and since analyzed objects are not too deeply buried, it can also be assumed that the antenna opening does not change significantly with the actual object depth (the so-called near-field assumption), meaning that the dimension of a region along the scanning direction does not depend on burial depth. Under these two assumptions, a first measure is chosen: the width of each selected region in the scanning direction, $y_{\text{size}}$.

Since the analyzed C-scan is obtained from energy projected A-scans preprocessed by means of TVG and background removal, if the difference in material and shapes is ignored, it can be said that the energy of a local maximum is proportional to the volume of the corresponding object. This is the second chosen measure, $E$.

The reflection of a GPR signal on the interface causes peaks in A-scans, the strength of which depends mainly on the type of material of the object. Once the background is removed, the first peak should correspond to the position of the top surface of the object, i.e. to its depth. That is the third chosen measure, expressed in number of time samples and obtained as the sample number of the first maximum in $s_{\text{new}}$. We convert the depth $n_1$, expressed in number of samples, to $d_1$ in centimeters through proportionalities, by locating an object with a known depth in a calibration area, and assuming that electromagnetic properties of the sand remain constant within the lane, and that the soil is homogeneous and isotropic [24]. For such an object buried at depth $d_{\text{ref}}$, the first maximum in its A-scan appears at the sample number $n_{\text{ref}}$. Analyzing the raw A-scans, it can be seen that the air/ground interface appears around the sample number $n_{\text{surf}}$. This means that for buried objects ($n_1 > n_{\text{surf}}$), one can estimate the depth in centimeters by:

$$d_1 = \frac{d_{\text{ref}}(n_1 - n_{\text{surf}})}{n_{\text{ref}} - n_{\text{surf}}} \tag{17.8}$$

The soil surface is taken as the reference point for the depth and the negative sign indicates depths in the ground. In the case where an object is surface-laid ($n_1 < n_{\text{surf}}$), the height of its top above the surface could be estimated taking antennas as the reference object, and knowing its height. Unfortunately, it is not possible to determine which sample number the antenna position corresponds to, since in practice, sample number 0 is chosen in an arbitrary way. This makes distance estimations above the surface unreliable. Still, the real power of GPR is in its subsurface detection abilities, and for surface-laid objects various other sensors could be used in combination with this sensor. Therefore, we choose to assign all positive depths to zero.

### 4.2.2 B-scan Hyperbola Detection and Chosen Measures

Another way of getting useful measures from GPR data is to analyze B-scans obtained from A-scans preprocessed by the TVG and background removal methods. Namely, while A-scans can give local information at the position of a local maximum only, B-scans can provide more global information about its neighborhood. An interesting way to gain information about the size of an object, its 3D position and the propagation velocity of electromagnetic waves above the object, is by analyzing characteristic hyperbolic shapes on B-scans [32]. These shapes result from the poor directivity of GPR antennas, due to which reflections of a small object, approximated by a point scatterer, are smeared out [13] over a broad region in B-scans.

The geometry of GPR data acquisition can be presented as in Figure 17.1, for a given displacement of the antennas from the starting position of a B-scan line. $Y_{\text{t}}$ denotes the displacement for the transmitting (T) and $Y_{\text{r}}$ for the receiving (R) antenna, where their difference, $\Delta Y = Y_{\text{r}} - Y_{\text{t}}$ is a characteristic of the GPR. The lateral distance of the observed object from the same starting point is $Y_{\text{o}}$.

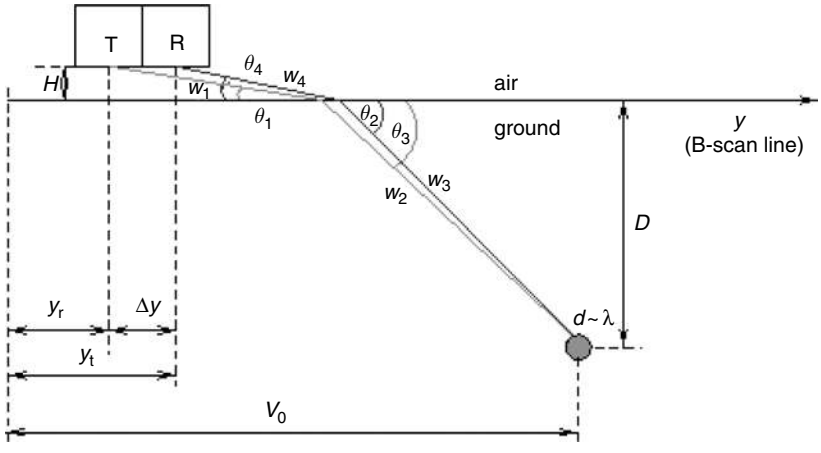



**Figure 17.1** The geometry of GPR data acquisition.

The height of the antennas above the soil surface is $H$, and it is assumed to be constant. It is assumed that the size of the object, $d$, is comparable to the wavelength of the GPR signal, $\lambda$, so that it can be approximated by a point scatterer. Note that this assumption is valid for AP mines and standard GPR frequencies. The burial depth is equal to $D$. The path of electromagnetic waves traveling from T to the object consists of two parts, one through the air ($w_1$) and another through the soil ($w_2$). At the interface between the two media, waves are refracted so that a change of path angle occurs, from $\theta_1$ to $\theta_2$, with respect to the soil surface. Similarly, the path of waves reflected from the object and traveling back to the GPR receiver consists of two parts, $w_3$ through the soil and $w_4$ through the air, with corresponding angles $\theta_3$ and $\theta_4$. A detailed analysis of this situation and corresponding complex calculations can be found in [11]. In the following, we introduce some simplifications of the geometry of GPR data acquisition, leading to simplified calculations that can be found in, e.g. [32]. There are several arguments in favor of these simplifications:

- the main interest in GPR is for subsurface imaging;
- it is easy to determine the air/ground interface on A- or B-scans, due to the strong reflection at such interfaces;
- GPR antennas often operate very close to the ground;
- the distance between T and R is often negligible in comparison with other distances.

Accordingly, at a first approximation, it can be said that $H = 0$ and $\Delta Y = 0$, and consequently, $Y_r = Y_t = Y_a$, where $Y_a$ is the distance of the central point between T and R from the starting position in a B-scan line. This further means that $w_1 = w_4 = 0$ and $w_2 = w_3 = w$, with $w$ being the (one-way) wavepath between the antennas and the object:

$$w = \sqrt{D^2 + (Y_a - Y_o)^2} \tag{17.9}$$

Finally, the round-trip travel time, or Time-Of-Flight (*TOF*), can be found as:

$$TOF = 2\frac{w}{v} = 2\frac{\sqrt{D^2 + (Y_a - Y_o)^2}}{v} \tag{17.10}$$

with $v$ being the propagation velocity of the electromagnetic waves through the soil (assuming that it is a constant value). *TOF* is usually expressed in discrete values, as a number of time samples $j$ taken every $\Delta T$:

$$TOF = j\Delta T \tag{17.11}$$

The data are stored in such a way that the $y$-axis is also discretized:

$$Y_a = i_a \Delta Y \tag{17.12}$$

$$Y_o = i_o \Delta Y \tag{17.13}$$

Here, $\Delta Y$ presents the acquisition step in the $y$-direction, while $i$ is the number of samples, $i_a$ being the sample number in the $y$-direction corresponding to one A-scan collecting position of antennas in a B-scan line, and $i_o$ being the discretized lateral position ($y$-coordinate) of the object. Substituting Equations (17.11), (17.12) and (17.13) into Equation (17.10), we come to the following:

$$j^2 = A + B(i_a - C)^2 \tag{17.14}$$

with:

$$A = \left(\frac{2D}{v\Delta T}\right)^2 \tag{17.15}$$

$$B = \left(\frac{2\Delta Y}{v\Delta T}\right)^2 \tag{17.16}$$

$$C = i_o \tag{17.17}$$

Equation (17.14) is the parametric equation of a hyperbola, showing that indeed, due to a poor directivity of T and R, objects appear as hyperbolae in B-scans: moving the antennas during the acquisition of a B-scan, means that their position $i_a$ changes, and one small object leaves a hyperbola in the acquired image.

As a next step, a way to detect or extract hyperbolae from B-scans has to be found. For that, the Randomized Hough Transform (RHT) [26,30] for hyperbola detection [32] can be of great help. In the following, we only briefly describe this idea, which is discussed in detail in [33], and which is based on modifying a previously developed method for ellipse detection by RHT [22,23]. First, we randomly choose three foreground pixels in an image containing edges of a thresholded B-scan image (after TVG and background removal). Then, we find the three parameters $(A, B, C)$ of the hyperbola that contains these points by substituting their coordinates into Equation (17.10). If the obtained parameters are realistic [33], they are stored as a potential final solution. This process is repeated a preset number of times, chosen as a compromise between speed and accuracy. Whenever a new hyperbola is found, it has to be checked to see whether it has already been found, and if this is the case, the number of times it was found increases. At the end, two possibilities exist for deciding which hyperbolae are the real solutions, by ranking them based either on the number of times each of them was found, or on the number of foreground pixels each of them contains. If the number of times the RHT is performed is high enough, the two ways should give the same results. If the number of times is not very high, the second way leads to better results.

If the detection is performed in an automated way, another question is how to determine how many highly ranked hyperbolae to preserve. It depends on how many objects can be expected in a scene. While estimating that number, one should not forget that depth is still a free dimension, so that there can be a few objects, placed one below the other. Therefore, in case there is no certain information

regarding the possible number of objects in the scene, the safest way is to choose several hyperbolae and eliminate some of them later, on the basis of how realistic the measures estimated from each of them are. If human interaction is envisaged, the problem can be simply solved by visually estimating how many hyperbolae indeed exist in the scene.

Equations (17.16), (17.15) and (17.17) can be rewritten, respectively, as:

$$v = \frac{2\Delta Y}{\Delta T\sqrt{B}} \tag{17.18}$$

$$D = \frac{v\Delta T\sqrt{A}}{2} \tag{17.19}$$

$$i_{\mathrm{o}} = C \tag{17.20}$$

Thus, once the typical hyperbolic shape is extracted from a B-scan, the propagation velocity in the medium above the object, the burial depth of the object and its cross-track coordinate *xy*-position (and so its 3D position) can be estimated. In addition, the hyperbola opening is proportional to the size of the object $d$ [32]:

$$d = k\frac{A}{B} \tag{17.21}$$

with $k$ being the characteristic of the scattering function that depends on object shape. Since we do not have information about the shapes of objects, we assume that their scattering functions are approximately the same.

There are two ways of finding the burial depth, not always leading to the same results, so we use two notations, $D^*$ and $D$. Taking into account that $\sqrt{A}$ is the position of the top of the hyperbola, $D^*$ is the depth obtained from it through proportionalities, as explained in the previous subsection for depths below the soil surface (see Equation (17.8)), i.e. for values $\sqrt{A} > n_{\mathrm{surf}}$. If this condition is not satisfied, meaning that the top of such an object is found to be above the surface, $D^*$ is set to 0 (see the previous subsection). $D$ is the depth that is found from the velocity using Equation (17.19), taking into account, once again, where the top of the obtained hyperbola ($\sqrt{A}$) is in comparison with the position of the air/ground interface. If $\sqrt{A} > n_{\mathrm{surf}}$ is satisfied, Equation (17.19) is modified to:

$$D = \frac{v\Delta T\left(n_{\mathrm{surf}} - \sqrt{A}\right)}{2} \tag{17.22}$$

In this case, $D$ presents the depth measured from the air/ground interface and it has a negative sign. If $\sqrt{A} \leq n_{\mathrm{surf}}$, Equation (17.19) is applied, giving the distance between the top of the hyperbola and the sample number 0. As mentioned in the previous subsection, this number does not say much, except that its positive sign means that such an object is laid above the surface. Consequently, $D^*$ and $D$ must not be compared directly above the surface. Below the soil surface, $D^*$ and $D$ ideally should be equal. In reality, their differences result from the way they are estimated, since $D^*$ assumes that $v$ remains the same everywhere and is equal to the one that is found for the reference object. If $v$ estimated from hyperbolae differs significantly from that reference value, the value of $D$ will be affected. The antennas of the GPR are generally not optimized for detection above the surface, which diminishes the need for a precise depth estimation there. The only important information in that case is its positive sign, indicating that such an object is above the surface.

The propagation velocity calculated from Equation (17.18) should have a value within a known range of values for that type of soil if the object is buried below the soil surface (if $D$ is negative). If the object is placed above the soil surface (positive $D$), ideally, $v$ should be equal to the propagation velocity in the air, $c = 3 \cdot 10^8\,\mathrm{m/s}$.

Based on the above, we select the following independent measures extracted by hyperbola detection for further modeling:

- depth information given by $D^*$;
- propagation velocity $v$ together with the sign of $D$;
- the ratio between the object size and its scattering function, $d/k$.

### *4.3 MD Measures*

If the object is metallic, the size and shape detected by an MD should ideally correspond to those found by GPR and IR. If it has a low metal content, the shape and size of the metal cannot be linked with the detections of the other two sensors. The same distinction appears in the terminology of humanitarian demining, where mines are classified, on the basis of their metal content, into three types: metallic, low-metal content (so-called 'minimum metal content' [34]) and nonmetallic. In this classification, a metallic mine is one made almost completely of metal, except e.g. handles, while a mine with low metal content has metal only in small parts, e.g. in the fuse. In addition, depth extracted from deconvolution of the MD image could be used in a similar way as the depth information extracted from GPR. [35].

However, in reality, the scanning resolution in one dimension is set to be around the expected size of AP mines, as a compromise between the necessity not to miss any mine, on the one hand, and on the other, the speed of the data acquisition and the field area to be analyzed. For such a resolution, which is around the size of an AP mine, there is no use of the area and shape measures. Furthermore, due to the fact that there is no knowledge on the PSF of the applied MD, and besides that the image is saturated and the resolution is poor, depth measure extraction through deconvolution is impossible.

From each of the regions selected by the local maxima analysis method for saturated images, with or without windows (see Section 3), three parameters can be extracted:

- number of pixels, $N_p$;
- the value of its local maximum, $V_{max}$;
- the width of the region in the $y$-direction, $w_y$.

## 5. Modeling of Measures in Terms of Belief Functions and Their Discounting

In real mine detection situations, the acquired data are far from numerous enough for reliable statistical learning. Besides, they are highly variable depending on the context and conditions [36]. Furthermore, not every possible object, neither mines nor objects that can be confused with them, can be modeled. On the other hand, some general knowledge exists regarding AP mines, their sizes, shapes, burial depths, etc., as well as regarding detection possibilities of each of the sensors. For these reasons, we decide to model and combine the measures of each of the sensors in terms of belief functions within the DS theory, since in this framework, ignorance, partial knowledge, uncertainty and ambiguity can be appropriately modeled [4,5].

Note that all three sensors give images, around 90 % of the mines have an elliptical (regular) top surface seen under some angle, and the major goal of our humanitarian demining efforts is to distinguish between a mine and a non-dangerous (friendly) object (stones, cans, etc.). Therefore, we define the frame of discernment $\Theta$ as: $\Theta = \{\text{MR, MI, FR, FI}\}$. Here, MR is a mine of regular shape, MI a mine of irregular shape, FR a friendly object of regular shape and FI a friendly object of irregular shape.

The modeling step aims to define a mass function for each measure expressing the information provided by this measure on the presence of a mine. Note that the mass function is the distribution of an initial unitary amount of belief among the subsets of $\Theta$ [6]. In the following, we present a model for each of the measures, based on a bibliographic survey, as well as on the single sensor trials

within the Belgian HUDEM (HUmanitarian DEMining) project [3]. The proposed equations for mass assignments are only examples illustrating the required tendencies. The numbers and parameters in these mass functions are not really important. Only the general shape of these functions matters, and the method is robust with respect to the choice of parameters. Usually, one of the difficulties when using belief functions is the definition of focal elements, and in particular of disjunctions. The approach we propose here relies on the specificities of the proposed measures. For example, a shape measure is not able to separate MR and FR, and their disjunction will therefore be considered for mass assignment. This approach is detailed for each sensor and each measure below.

## *5.1 IR Measures*

### 5.1.1 Ellipse Fitting Mass Assignment

We apply the ellipse fitting algorithm described in the previous section on the edges of the selected IR regions. The mass for the subset of regular shapes assigned by this measure indicates how well this shape fits an ellipse:

$$m_{\mathrm{f}}(\mathrm{MR} \cup \mathrm{FR}) = \max\left(0, \min\left\{\frac{A_{\mathrm{oe}} - 5}{A_{\mathrm{o}}}, \frac{A_{\mathrm{oe}} - 5}{A_{\mathrm{e}}}\right\}\right) \tag{17.23}$$

where $A_{\mathrm{oe}}$ is the part of the object area that belongs to the fitted ellipse as well, $A_{\mathrm{o}}$ is the object area and $A_{\mathrm{e}}$ is the ellipse area. The subtraction of five pixels is introduced to include the limit case of an ellipse with just five pixels, where we cannot judge about the shape at all, so ignorance should be maximal.

The mass of the irregular subset is the larger of two values, the percentage of ellipse area that does not belong to the object and the percentage of the object area that does not belong to the fitted ellipse:

$$m_{\mathrm{f}}(\mathrm{MI} \cup \mathrm{FI}) = \max\left\{\frac{A_{\mathrm{e}} - A_{\mathrm{oe}}}{A_{\mathrm{e}}}, \frac{A_{\mathrm{o}} - A_{\mathrm{oe}}}{A_{\mathrm{o}}}\right\} \tag{17.24}$$

The full set gets the remaining mass:

$$m_{\mathrm{f}}(\Theta) = 1 - m_{\mathrm{f}}(\mathrm{MR} \cup \mathrm{FR}) - m_{\mathrm{f}}(\mathrm{MI} \cup \mathrm{FI}) \tag{17.25}$$

### 5.1.2 Elongation Mass Assignment

Once *ratio*1 and *ratio*2 (see the previous section) are found, the mass value for the regular subset is the smaller of them:

$$m_{\mathrm{e}}(\mathrm{MR} \cup \mathrm{FR}) = \min(ratio1, ratio2) \tag{17.26}$$

while the mass of the irregular subset is the absolute value of their difference:

$$m_{\mathrm{e}}(\mathrm{MI} \cup \mathrm{FI}) = |ratio1 - ratio2| \tag{17.27}$$

The full set takes the rest:

$$m_{\mathrm{e}}(\Theta) = 1 - \max(ratio1, ratio2) \tag{17.28}$$

### 5.1.3 Area/Size Mass Assignment

If the approximate range of expectable mine areas is from $A_{\min}$ to $A_{\max}$, based on the reasoning given in the previous section, masses are modeled as:

$$m_{\mathrm{a}}(\Theta) = \alpha \cdot \exp \frac{[A_{\grave{\mathrm{a}}} - 0.5 \cdot (A_{\min} + A_{\max})]^2}{0.5 \cdot (A_{\max} - A_{\min})} \tag{17.29}$$

$$m_{\mathrm{a}}(\mathrm{FR} \cup \mathrm{FI}) = 1 - m_{\mathrm{a}}(\Theta) \tag{17.30}$$

The parameter $\alpha$ is usually set to a value close to 1. Taking it not exactly equal to 1 allows for having nonzero values for the complement function, which avoids completely excluding some solutions (since mass functions are combined using product operators, having zero somewhere is very strong).

## *5.2 GPR A-scan and Preprocessed C-scan Measures*

### 5.2.1 $y_{\mathrm{size}}$ Mass Assignment

This measure is extracted as the width of a region selected by the local maxima method. It cannot provide information about mines alone. Although we know the approximate range of sizes of AP mines, still, whenever an object has a size within that range, it could be something else as well. Therefore, in that range, masses should be mainly assigned to the full set, $\Theta$. If the object is too large or too small, it is far more likely that it is not a dangerous one, and a large part of the mass should be given to friendly objects. Therefore, it makes sense to model masses as shown in Figure 17.2(a). The position of the center and the width of the central interval, where masses go mainly to the full set, depend on the available information. If there is no information regarding expectable mine size, or if a wide range of sizes can be expected, these curves should be quite uninformative, so the central interval should be very wide. If it is known which types of mine can be expected in a minefield and their size is similar, the curves, and accordingly this measure, become very selective (narrow central interval).

### 5.2.2 *E* Mass Assignment

Similarly to $y_{\mathrm{size}}$, whenever the energy is as expected for mines, it could be any other object as well, assigning masses mainly to $\Theta$. Otherwise, it is likely that an object is friendly, so masses can be modeled as shown in Figure 17.2(b). The remaining reasoning is the same as for the previous measure.

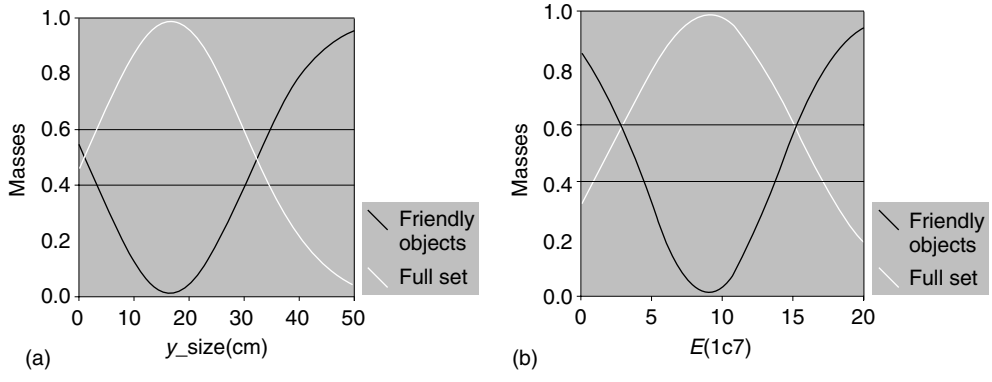



**Figure 17.2** Masses assigned by (a) the $y_{\mathrm{size}}$ measure; and (b) the E measure.

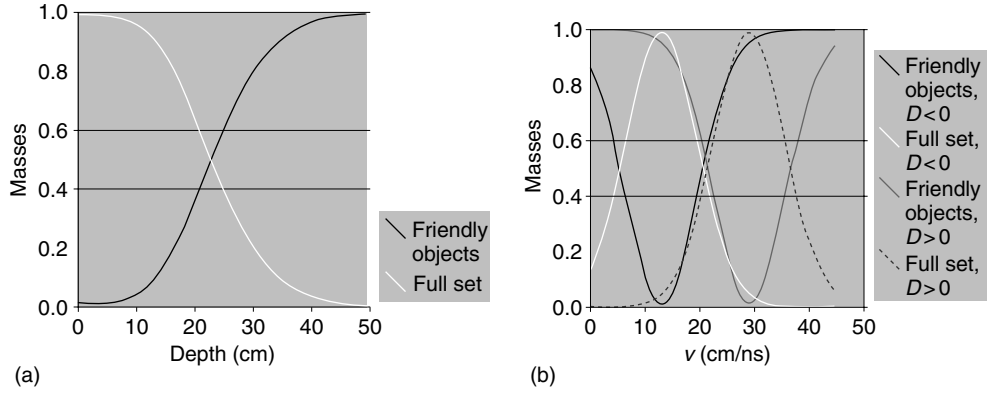



**Figure 17.3** Masses assigned by (a) the depth; and (b) the velocity measure.

### 5.2.3 $d_1$ Mass Assignment

If an object is buried too deeply, it is possibly a non-dangerous one. Otherwise, it could be anything. Following this logic, masses can be modeled as shown in Figure 17.3(a).

## *5.3 GPR B-scan (Hyperbola) Measures*

### 5.3.1 $D^*$ Mass Assignment

Since its meaning is the same as that of $d_1$, $D^*$ is modeled as given in Figure 17.3(a).

### 5.3.2 $v$ Mass Assignment

As mentioned earlier, the value of the propagation velocity depends on the medium, so, in the case of the soil, it should be around the values for this medium, and if it is the air, it should be close to $c$. In order to decide which model should be used, the sign of $D$ is used as an indicator. If the value of $v$ is expectable for a particular medium, an object that gives that estimation of $v$ could be anything. If $v$ differs significantly from the expected values for that medium, it can be expected that it is something friendly or simply background. This reasoning is illustrated in Figure 17.3(b).

### 5.3.3 $d/k$ Mass Assignment

If this measure is within a range of values that can be expected for mines, such an object could be anything from the full set. For very low or high values of this parameter, it is quite certain that the object is non-dangerous. Following this idea, masses are modeled as shown in Figure 17.4(a).

## *5.4 MD Measures*

### 5.4.1 Measures for MD Regions Selected by Local Maxima Analysis for Saturated Images With Windows

The number of pixels of a region corresponds to its area, which is a highly unreliable parameter, taking into account the poor $x$-direction resolution in real scenarios, so we discard this information. Since the value of a local maximum depends at least on the metal content of an object and its depth,

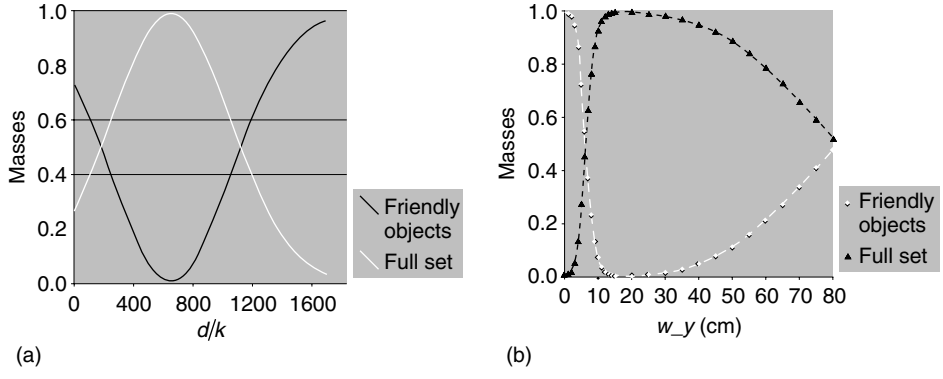



**Figure 17.4** Masses assigned by (a) the $d/k$ measure; and (b) the $w_y$ measure.

and since we cannot extract any of the two in any other way to our knowledge, $V_{\max}$ is not useful here either. Therefore, the only remaining information whose usefulness should be discussed here, is the width of the region in the $y$-direction, $w_y$. If we do not have information on the possible size of metal in the mines, we can only keep regions as they are. Consequently, if MD is used alone, we can treat all the regions as potential mines. For fusion with the other two sensors, we can also give the highest confidence of the MD in treating these regions as mines, plus providing the local maxima values in order to have a hint on its metallic content (by using the depth information of the GPR). On the other hand, if we have information on the expected sizes of metal in a minefield, we can include that information in the sense of assigning masses by this measure. For example, if it is expected that there are no mines for which the size of the metal is smaller than around 5 cm, nor larger than around 80 cm (which is quite loose in reality), we can model mass assignments by the $w_y$ measure, as given in Figure 17.4(b).

This reasoning can be refined if we have additional knowledge or information about the metal content of the objects. An approach dealing with such information is proposed in [35].

### 5.4.2 Measures for MD Regions Selected by Local Maxima Analysis for Saturated Images Without Windows

The discussion regarding the usefulness of the first two measures, $N_p$ and $V_{\max}$, remains the same as in the previous case. Regarding $w_y$ information, this time we cannot make use of it at all, since an underlying assumption for using this type of local maxima region selection is that we do not have information on the expected sizes of objects, nor their arrangements in a minefield. This means that this is our final output, or, taking the cautious approach, that all the MD selected regions should be treated as potential mines. Therefore, in the case of fusion of MD with the other two sensors, each of these regions will have the maximum value of the mass of the full set, meaning that the ignorance is the highest possible.

## *5.5 Discounting Factors*

The behavior of each of these sensors is strongly scenario-dependent, referring to:

- the quality of the acquired data;
- the reliability of each of the sensors under particular weather conditions, type of soil, etc.;
- the types of object under analysis.

These are the reasons for including discounting factors [4,37,38] in the model. Discounting factors consist of three types of parameter:

- $g_{ij}$ – the confidence level of sensor $j$ in its assessment when judging measure $i$ (0 – not confident at all, 1 – completely confident);
- $b_{ij}$ – the level of importance of measure $i$ of sensor $j$ (1 – low, $b_{\text{scale}}$ – high, where $b_{\text{scale}}$ is the scale for $b$ parameters);
- $s_j$ – the deminer's confidence in the opinion of sensor $j$ (1 – low, $s_{\text{scale}}$ – high, where $s_{\text{scale}}$ is the scale for $s$ parameters).

Discounting and combination do not commute, so it has to be pointed out when each of the discounting parameters is used. First, $g_{ij}$ and $b_{ij}/b_{\text{scale}}$ are used to discount masses assigned by measure $i$ of sensor $j$, and that is done for all measures of that sensor. Then we combine the measures per sensor. After that, the resulting masses are discounted using $s_{ij}/s_{\text{scale}}$ parameters, before combining the sensors. These factors modify masses assigned for each measure so that:

1. For any subset $A \neq \Theta$ and any measure $i$, new masses, $m_i(A)$, are computed from the initial ones, $m_{i\text{IN}}(A)$, as:

$$m_i(A) = \beta \cdot m_{i\text{IN}}(A). \tag{17.31}$$

2. For the full set:

$$m_i(\Theta) = 1 - \beta \cdot [1 - m_{i\text{IN}}(\Theta)] \tag{17.32}$$

where $\beta$ is replaced by either $g_{ij}$, $b_{ij}/b_{\text{scale}}$ or $s_{ij}/s_{\text{scale}}$.
The discounting factors are presented in detail in [35].

## 6. Region Association, Combination of Measures and Decision

### *6.1 Region Association*

At this point, one more problem arises, and that is how to associate the regions selected by IR, MD and GPR, since:

- typically, they all have different resolutions in the two directions; and
- IR provides separate images of the regions, while the other two provide one image containing all the regions.

We propose a very simple solution, and that is to analyze the distances between the centers of the selected regions, or, to be more precise:

- the positions of the local maxima of the regions in the cases of GPR and MD;
- for IR, the coordinates of detected ellipses, or the coordinates of the center of gravity in the cases of regions with only a few pixels.

The sensors are associated two by two, analyzing Euclidean distances between their centers and grouping the regions the distance of which is below some maximum allowed value. In the case where there is more than one region of one sensor close enough to some region of another sensor, the one with the smallest distance is chosen.

Regarding the maximum allowed distance, it must be determined on the basis of the typically coarse resolutions of MD and GPR, so it must be quite loose. Note that, instead of analyzing the Euclidean distance, it is possible to differentiate the distance in the $x$-direction, with a higher tolerance due to a

coarse resolution, and in the $y$-direction, where all three sensors have a good resolution. Still, our tests have shown that there is no difference in results.

Taking into account that sensors are grouped two by two, it may happen that the results of association are ambiguous, i.e. that, for example, Region 3 of IR is grouped with Region 20 of GPR, that Region 3 of IR is grouped with Region 7 of MD, and that Region 20 of GPR is grouped with Region 5 of MD (and no IR region is grouped with them). Of course, this situation sounds perfect for testing possibilities of clustering sensors [39]. Unfortunately, in cases where there is no possibility of applying the method due to severe limitations of the MD data, the only remaining possibility is to analyze all the obtained groups, and after combination, see how to treat the results. In other cases (the same states if the mass assignment of GPR is unchanged), since the masses are assigned only to the friendly subset and the full set, there is no possibility of having a conflict with IR.

## 6.2 Combination of Masses

The combination is performed using Dempster's rule in unnormalized form [4,6,38]:

$$m(A) = \sum_{\substack{k,l \\ A_k \cap B_l = A}} m_i(A_k) \cdot m_j(B_l) \tag{17.33}$$

where $m_i$ and $m_j$ are masses assigned by measures $i$ and $j$ after discounting, and their focal elements are $A_1, A_2, \ldots, A_m$ and $B_1, B_2, \ldots, B_n$, respectively. Since Dempster's rule is commutative and associative, it can be applied repeatedly regardless of order, until all measures are combined. The measures under combination have to fulfill the condition of being 'cognitively' independent [38], which is weaker than the notion of statistical independence required in the probabilistic case. Measures are cognitively independent if each of them assigns masses without any direct link to the others. This is the case in the proposed model.

First, the fusion of measures per sensor is done and internal conflicts (masses of the empty set) of IR and of GPR are analyzed (as mentioned earlier, it does not make sense to do that for MD). If the internal conflict of a sensor is high, its data are corrupted by noise, occlusion, etc., so they should be discounted. We cannot know for sure which information is more affected by this corruption than others. Still, discounting is performed on each measure, since it is modeled in such a way that its influence is proportional to the level of ambiguity and uncertainty of the information, as shown in the previous subsection.

Since both GPR sets of measures provide the depth information, this offers a possibility to discount IR in the function of depth. Taking into account knowledge about the IR camera and its reliability as a function of the depth, we have clustered IR separately as soon as the depth extracted by GPR is below 5 cm, and in between, the measures of IR are linearly discounted as a function of the GPR depth by the factor $r = 1 + (d/5)$. Here, $d$(cm) is the depth extracted by the GPR (and it is negative below the soil surface).

After discounting the sensors that have high internal conflicts, the sensors for each associated region are fused. Note that performing the combination in two steps (first for all information extracted from one sensor and then between sensors) is consistent because of the associativity of Dempster's rule: if there is no discounting, the result is the same as for direct global combination. What can still happen, however, is that the conflict after fusion (mass of the empty set) is high. In other words, although it seemed quite possible that they refer to the same object, they actually do not. For example, although IR could reach some depths of 5 cm, where GPR and MD detect something, it actually detects something nonmetallic on the surface. In that case, taking into account the safety and need not to miss mines, we declare that there is more than one object in the region and analyze the object of each of the sensors separately.

## 6.3 Decision

After clustering and combination of sensors, the resulting masses are found for each of the clusters. On the basis of these masses, a final conclusion about the true identity of each object under analysis has to be made for every cluster. Usual decision rules rely on beliefs, *Bel*, plausibilities, *Pl* [4], and pignistic probabilities, *P* [40], defined as follows:

- for any subset $B$:

$$Bel(B) = \sum_{A \subset B, A \neq 0} m(A) \tag{17.34}$$

$$Pl(B) = \sum_{B \cap A \neq 0} m(A) \tag{17.35}$$

- for any singleton $C$:

$$P(C) = \sum_{A \in 2^{\Theta}, C \in A} \frac{m(A)}{|A|\,[1 - m(0)]} \tag{17.36}$$

Unfortunately, a decision rule based on any of these three functions does not lead to fruitful conclusions, since there are no focal elements containing mines alone [41]. Namely, after combination of the measures on the second level, the resulting focal elements are: FR, FI, (FR∪FI), (MR∪FR), (MI∪FI) and Θ. Hence, as shown in [35], the following relations are always true:

$$Bel(M) \leq Bel(F) \tag{17.37}$$

$$Pl(M) \leq Pl(F) \tag{17.38}$$

$$P(M) \leq P(F) \tag{17.39}$$

so the decision would always be made in favor of $F$. Here, $M$ denotes mines ($M = \mathrm{MR} \cup \mathrm{MI}$) and $F$ denotes friendly objects ($F = \mathrm{FR} \cup \mathrm{FI}$).

Since, in the case of any ambiguity, far more importance has to be given to mines, we propose to define guesses, $G(A)$, where $A \in \{M, F, 0\}$, in the following way:

$$G(M) = \sum_{M \cap B \neq 0} m(B) \tag{17.40}$$

$$G(F) = \sum_{B \subseteq F, B \neq 0} m(B) \tag{17.41}$$

$$G(0) = m(0) \tag{17.42}$$

The guess value of a mine is the sum of masses of all the focal elements containing mines, no matter whether they are regular or irregular. The guess of a friend is the sum of masses of all the focal elements containing nothing else but friends (again, either regular or irregular), and the guess of something else is equal to the mass of the empty set. Note that the guess of a mine is equal to its plausibility, while the guess of a friend is equal to its belief. This reflects the fact that we have to be cautious in deciding $F$. In other words, the introduced guesses are but a cautious way for estimating confidence degrees. Once guesses for the three types of object are found, they are ordered and this list, together with confidence degrees, is given as the final result.

## 7. Results

In this section, we present results that have been obtained on real data, provided by the TNO Physics and Electronics laboratory, The Hague, The Netherlands, within the Dutch HOM-2000 project. These data include IR, GPR and MD images obtained on a sand lane containing 21 mines and 7 friendly objects. After the processing and region selection of each type of data, 42 regions were obtained, 28 corresponding to regions containing the actual objects, and 14 for which clutter caused alarms. This means that finally we had to recognize 21 mines and 21 false alarms. When we combined A-scan and C-scan measures of GPR with the other two sensors, the following results were obtained:

- 19 mines were detected and two mines were missed, due to the sensitivity of the used sensors and not due to the method proposed here;
- six placed false alarms were correctly recognized, and one was wrongly classified as a mine; eight clutter-caused alarms were classified as being friendly, and six were wrongly classified as mines.

If we compare these results to the ones obtained on each sensor separately, significant improvements can be observed:

- from IR alone, we got six more undetected mines and one more false alarm;
- from MD alone, we got three more false alarms and one more undetected mine;
- from GPR alone, we got four more false alarms.

This shows that the fusion using the proposed approach allows us to improve the mine detection rate, while decreasing the false alarm rate.

When we combined B-scan measures of GPR with the other two sensors, the results were exactly the same for the mines and for the false alarms. However, the number of false alarms due to placed objects increased, while those due to clutter decreased, keeping the global false alarm rate constant. Our tests also showed that the results are robust with respect to changes of the size of the window for MD region selection, the way MD alarms are treated (either using the region size measure or simply considering all alarms as mines), as well as the type of the chosen set of measures of GPR. This proves the power of the developed model, both for each of the sensors separately and for their fusion. Furthermore, the clustering of the sensors leads to superior results in ambiguous cases with high external conflicts, which justifies our choice of an unnormalized combination rule.

## 8. Conclusion

We proposed in this chapter a complete and original approach to analyzing and combining data from different sensors for anti-personnel mine recognition. The extraction of features from sensors makes use of the characteristics of each sensor, and is therefore specific to the sensors under study (infrared images, metal detectors or ground penetrating radar).

The next steps are more general. These features have been modeled in the belief function framework. The difficult step of defining the disjunctive focal elements and the corresponding masses has been solved based on the knowledge we have about the extracted features. This way of reasoning could be easily adapted to other sensors.

The fusion step is completely general. In particular, we have solved in an elegant way the problem occuring when several objects are actually seen by different sensors. Reliability of sources, as well as expert confidence values, are introduced as discounting factors.

As for the decision step, due to the fact that the errors do not have the same impact depending on the decided object, we designed a new decision rule that favors the decision for a mine in the case of ambiguities.

Experimental results were shown on real objects put in a sand lane. They are very good. The benefit of fusion is demonstrated on these data: it improves significantly the decisions. This approach could now be tested on some other real data sets, as well as on other sensors.

## Acknowledgments

We are thankful to the TNO Physics and Electronics Laboratory, The Hague, The Netherlands for giving us the permission and great opportunity to work on the data gathered on their test facilities within the Dutch HOM-2000 project.

## References

[1] Baum, C. E. *Detection and Identification of Visually Obscured Targets*, Taylor & Francis, Philadelphia, USA, 1999.

[2] Young, J. D. and Peters, L. Jr. "A Brief History of GPR Fundamentals and Applications," *Proceedings of 6th International Conference on Ground Penetrating Radar (GPR'96)*, Sendai, Japan, pp. 5–14, 1996.

[3] Milisavljević, N. and Acheroy, M. "Overview of Mine Detection Sensors and Ideas for their Estimation and Fusion in the Scope of HUDEM Project," *Proceedings of Australian–American Joint Conference on the Technologies of Mines and Mine Countermeasures*, Sydney, Australia, 1999.

[4] Shafer, G. *A Mathematical Theory of Evidence*, Princeton University Press, 1976.

[5] Smets, P. "What is Dempster–Shafer's Model?," in Yager, R. R., Fedrizzi, M. and Kacprzyk, J. (Eds) *Advances in the Dempster–Shafer Theory of Evidence*, pp. 5–34, John Wiley & Sons, 1994.

[6] Smets, P. and Kennes, R. "The Transferable Belief Model," *Artificial Intelligence*, **6**, pp. 191–234, 1994.

[7] de Jong, W., Lensen, H. A. and Janssen, Y. H. L. "Sophisticated Test Facility to Detect Landmines," *Proceedings of SPIE Conference on Detection Technologies for Mines and Minelike Targets*, **3710**, pp. 1409–1418, 1999.

[8] den Breejen, E., Schutte, K. and Cremer F. "Sensor fusion for anti-personnel landmine detection, a case study," *Proceedings of SPIE Conference on Detection Technologies for Mines and Minelike Targets*, **3710**, pp. 1235–1245, 1999.

[9] Perrin, S. *Contribution à l'algorithmique multicapteur pour la détection de mines antipersonnel*, PhD dissertation, Ecole Centrale de Lille, USTL, France, 2001.

[10] Denoeux, T. *Reasoning With Imprecise Belief Structures*, Technical report Heudiasys 97/44, Université de Technologie de Compiègne, France, 1997.

[11] Milisavljević, N. *Analysis and Fusion Using Belief Function Theory of Multisensor Data for Close-Range Humanitarian Mine Detection*, PhD dissertation, Ecole Nationale Supérieure des Télécommunications, Paris, France, 2001.

[12] Daniels, D. J. *Surface-Penetrating Radar*, The Institution of Electrical Engineers, London, United Kingdom, 1996.

[13] Scheers, B. *Ultra-Wideband Ground Penetrating Radar, with Application to the Detection of Anti-Personnel Landmines*, PhD dissertation, Université Catholique de Louvain, Louvain-La-Neuve, Belgium, 2001.

[14] Gonzalez, R. C. and Woods, R. E., (Eds) *Digital Image Processing*, Addison-Wesley Publishing Company, USA, 1992.

[15] Kundur, D. *Blind Deconvolution of Still Images Using Recursive Inverse Filtering*, Masters thesis, Department of Electrical and Computer Engineering, University of Toronto, Canada, 1995.

[16] Kundur, D. and Hatzinakos, D. "Blind Image Deconvolution," *IEEE Signal Processing Magazine*, **13**(3), pp. 43–64, 1996.

[17] Kundur, D. and Hatzinakos, D. "Blind Image Restoration via Recursive Filtering," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, **4**, pp. 2283–2286, 1996.

[18] Milisavljević, N. *et al.* "Comparison of Belief Functions and Voting Method for Fusion of Mine Detection Sensors," *Proceedings of SPIE Conference on Detection and Remediation Technologies for Mines and Minelike Targets VI*, **4394**, pp. 1011–1022, 2001.

[19] Breuers, M. G. J., Schwering, P. B. W. and van den Broek, S. P. "Sensor Fusion Algorithms for the Detection of Land Mines," *Proceedings of SPIE Conference on Detection Technologies for Mines and Minelike Targets*, **3710**, pp. 1160–1166, 1999.

[20] Dolan, J. and Riseman, E. "Computing Curvilinear Structure by Token-Based Grouping," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Champaign, IL, USA, pp. 264–270, 1992.

[21] Lim, S. G. and Alder, M. D. "A Nonparametric Approach for Detecting Lines and Curves," *Proceedings of International Conference on Image Processing*, Lausanne, Switzerland, pp. 837–840, 1996.

[22] McLaughlin, R. A. and Alder, M. D. *The Hough Transform versus the UpWrite*, TR97-02, CIIPS, The University of Western Australia, 1997.
[23] Milisavljević, N. "Comparison of Three Methods for Shape Recognition in the Case of Mine Detection," *Pattern Recognition Letters*, **20**(11–13), pp. 1079–1083, 1999.
[24] Haig, T., Attikiouzel, Y. and Alder, M. D. "Border Following: New Definition Gives Improved Border," *IEE Proceedings-I*, **139**(2), pp. 206–211, 1992.
[25] McLaughlin, R. A. *Randomized Hough Transform: Improved Ellipse Detection with Comparison*, TR97-01, CIIPS, The University of Western Australia, 1997.
[26] Xu, L. "Randomized Hough Transform (RHT): Basic Mechanisms, Algorithms and Computational Complexities," *CVGIP: Image Understanding*, **57**(2), pp. 131–154, 1993.
[27] Xu, L., Oja, E. and Kultanen, P. "A New Curve Detection Method: Randomized Hough Transform (RHT)," *Pattern Recognition Letters*, **11**, pp. 331–338, 1990.
[28] Duda, O. and Hart, P. E. "Use of the Hough Transform to Detect Lines and Curves in Pictures," *Communications of the Association for Computing Machinery*, **15**(1), pp. 11–15, 1972.
[29] Kälviäinen, H., Hirvonen, P., Xu, L. and Oja, E. "Comparisons of Probabilistic and Non-Probabilistic Hough Transforms," *Proceedings of 3rd European Conference on Computer Vision*, Stockholm, Sweden, pp. 351–360, 1994.
[30] Leavers, V. F. *Shape Detection in Computer Vision Using the Hough Transform*, Springer, London, 1992.
[31] Yuen, H. K., Illingworth, J. and Kittler, J. "Detecting Partially Occluded Ellipses using the Hough Transform," *Image and Vision Computing*, **7**(1), pp. 31–37, 1989.
[32] Capineri, L., Grande, P. and Temple, J. A. G. "Advanced Image-Processing Technique for Real-Time Interpretation of Ground Penetrating Radar Images," *International Journal on Imaging Systems and Technology*, **9**, pp. 51–59, 1998.
[33] Milisavljević, N., Bloch, I. and Acheroy, M. "Application of the Randomized Hough Transform to Humanitarian Mine Detection," *Proceedings of the 7th IASTED International Conference on Signal and Image Procesing (SIP2001)*, Honolulu, Hawaii, USA, pp. 149–154, 2001.
[34] Banks, E. *Antipersonnel Landmines – Recognising and Disarming*, Brassey's, London-Washington, 1997.
[35] Milisavljević, N. and Bloch, I. "Sensor Fusion in Anti-Personnel Mine Detection Using a Two-Level Belief Function Model," *IEEE Transactions On Systems, Man, and Cybernetics C*, **33**(2), pp. 269–283, 2003.
[36] Milisavljević, N., Bloch, I., van den Broek, S. P. and Acheroy, M. "Improving Mine Recognition through Processing and Dempster–Shafer Fusion of Ground-Penetrating Data," *Pattern Recognition*, **36**(5), pp. 1233–1250, 2003.
[37] Dubois, D., Grabisch, M., Prade, H. and Smets, P. "Assessing the Value of a Candidate," *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence (UAI'99)*, Stockholm, Sweden, pp. 170–177, 1999.
[38] Smets, P. "Belief Functions: the Disjunctive Rule of Combination and the Generalized Bayesian Theorem," *International Journal of Approximate Reasoning*, **9**, pp. 1–35, 1993.
[39] Schubert, J. "On Nonspecific Evidence," *International Journal of Intelligent Systems*, **8**, pp. 711–725, 1993.
[40] Smets, P. "Constructing the Pignistic Probability Function in a Context of Uncertainty," *Uncertainty in Artificial Intelligence*, **5**, pp. 29–39, 1990.
[41] Milisavljević, N., Bloch, I. and Acheroy, M. "Characterization of Mine Detection Sensors in Terms of Belief Functions and their Fusion, First Results," *Proceedings of 3rd International Conference on Information Fusion (FUSION 2000)*, **II**, pp. ThC3.15–ThC3.22, 2000.

# 18

# Fast Object Recognition Using Dynamic Programming from a Combination of Salient Line Groups

**Dong Joong Kang**
**Jong Eun Ha**
School of Information Technology, Tongmyong University of Information Technology, Busan 608-711, Korea

**In So Kweon**
Department of Electrical & Computer Science Engineering, Korea Advanced Institute of Science and Technology, Daejun, Korea

*This chapter presents a new method of grouping and matching line segments to recognize objects. We propose a dynamic programming-based formulation extracting salient line patterns by defining a robust and stable geometric representation that is based on perceptual organizations. As the end point proximity, we detect several junctions from image lines. We then search for junction groups by using the collinear constraint between the junctions. Junction groups similar to the model are searched in the scene, based on a local comparison. A DP-based search algorithm reduces the time complexity for the search of the model lines in the scene. The system is able to find reasonable line groups in a short time.*

## 1. Introduction

This chapter describes an algorithm that robustly locates collections of salient line segments in an image. In computer vision and related applications, we often wish to find objects based on stored models from an image containing objects of interest [1–6]. To achieve this, a model-based object recognition system

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

first extracts sets of features from the scene and the model, and then it looks for matches between members of the respective sets. The hypothesized matches are then verified and possibly extended to be useful in various applications. Verification can be accomplished by hypothesizing enough matches to constrain the geometrical transformation from a 3D model to a 2D image under perspective projection.

We first extract junctions formed by two lines in the input image, and then find an optimal relation between the extracted junctions, by comparing them with previously constructed model relations. The relation between the junctions is described by a collinear constraint and parallelism can also be imposed. Junction detection acts as a line filter to extract salient line groups in the input image and then the relations between the extracted groups are searched to form a more complex group in an energy minimization framework. The method is successfully applied to images with some deformation and broken lines. Because the system can define a topological relation that is invariant to viewpoint variations, it is possible to extract enough lines to guide 2D or 3D object recognition.

Conventionally, the DP-based algorithm as a search tool is an optimization technique for the problems where not all variables are interrelated simultaneously [7–9]. In the case of an inhomogeneous problem, such as object recognition, related contextual dependency for all the model features always exists [10]. Therefore, DP optimization would not give the true minimum.

On the other hand, the DP method has an advantage in greatly reducing the time complexity for a candidate search, based on the local similarity. Silhouette or boundary matching problems that satisfy the locality constraint can be solved by DP-based methods using local comparison of the shapes. In these approaches, both the model and matched scene have a sequentially connected form of lines, ordered pixels, or chained points [11–13]. In some cases, there also exist many vision problems, in which the ordering or local neighborhood cannot be easily defined. For example, definition of a meaningful line connection in noisy lines is not easy, because the object boundary extraction for an outdoor scene is itself a formidable job for object segmentation.

In this chapter, we do not assume known boundary lines or junctions, rather, we are open to any connection possibilities for arbitrary junction groups in the DP-based search. That is, the given problem is a local comparison between predefined and sequentially linked model junctions and all possible scene lines in an energy minimization framework.

Section 2 introduces previous research about feature grouping in object recognition. Section 3 explains a quality measure to detect two line junctions in an input image. Section 4 describes a combination model to form local line groups and how junctions are linked to each other. Section 5 explains how related junctions are searched to form the salient line groups in a DP-based search framework. Section 6 gives a criterion to test the collinearity between lines. Section 7 tests the robustness of the junction detection algorithm by counting the number of detected junctions as a function of the junction quality and whether a prominent junction from a single object is extracted under an experimentally decided quality threshold. Section 8 presents the results of experiments using synthetic and real images. Finally, Section 9 summarizes the results and draws conclusions.

## 2. Previous Research

Guiding object recognition by matching perceptual groups of features was suggested by Lowe [6]. In SCERPO, his approach is to match a few significant groupings from certain arrangements of lines found in images. Lowe has successfully incorporated grouping into an object recognition system. First, he groups together lines thought particularly likely to come from the same object. Then, SCERPO looks for groups of lines that have some property invariant with the camera viewpoint. For this purpose, he proposes three major line groups – proximity, parallelism and collinearity.

Recent results in the field of object recognition, including those of Jacobs, Grimson and Huttenlocher, demonstrate the necessity of some type of grouping, or feature selection, to make the combinatorics of object recognition manageable [9,14]. Grouping, as for the nonaccidental image features, overcomes the unfavorable combinatorics of recognition by removing the need to search the space for all matches

between image and model features. Grimson has shown that the combinatorics of the recognition process in cluttered environments using a constrained search reduces the time complexity from an exponential to a low-order polynomial if we use an intermediate grouping process [9]. Only those image features considered likely to come from a single object could be included together in hypothetical matches. And these groups need only be matched with compatible groups of model features. For example, in the case of a constrained tree search, grouping may tell us which parts of the search tree to explore first, or allow us to prune sections of the tree in advance.

This chapter is related to Lowe's work using perceptual groupings. However, the SCERPO grouping has a limitation: forming only small groups of lines limits the amount by which we may reduce the search. Our work extends the small grouping to bigger perceptual groups, including more complex shapes. Among Lowe's organization groups, the proximity consisting of two or more image lines is an important clue for starting object recognition. When projected to the image plane, most manmade objects may have a polyhedral plane in which two or several sides give line junctions. First, we introduce a quality measure to detect meaningful line junctions denoting the proximity. The quality measure must be carefully defined not to skip salient junctions in the input image. Then, extracted salient junctions are combined to form more complex and important local line groups. The combination between junctions is guided by the collinearity that is another of Lowe's perceptual groups. Henikoff and Shapiro [15] effectively use an ordered set of three lines representing a line segment with junctions at both ends. In their work, the line triples, or their relations as a local representative pattern, broadly perform the object recognition and shape indexing. However, their system cannot define the line triple when the common line sharing two junctions is broken by image noise or object occlusion. And the triple and bigger local groups are separately defined in low-level detection and discrete relaxation, respectively. The proposed system in this chapter is able to form the line triple and bigger line groups in a consistent framework. Although the common line is broken, the combination of the two junctions can be compensated by the collinearity of the broken lines. We introduce the following:

1. A robust and stable geometric representation that is based on the perceptual organizations (i.e. the representation as a primitive search node includes two or more perceptual grouping elements).
2. A consistent search framework combining the primitive geometric representations, based on the dynamic programming formulation.

## 3. Junction Extraction

A junction is defined as any pair of line segments that intersect, and whose intersection point either lies on one of the line segments, or does not lie on either of the line segments. An additional requirement is that the acute angle between the two lines must lie in a range $\theta_{\min}$ to $\theta_{\max}$. In order to avoid ambiguity with parallel or collinear pairs [6], $\theta_{\min}$ could be chosen to be a predefined threshold. Various junction types are well defined by Etemadi *et al.* [7].

Now a perfect junction (or two-line junction) is defined as one in which the intersection point **P** lies precisely at the end points of the line segments. Figure 18.1 shows the schematic diagram of a typical junction. Note that there are now two virtual lines that share the end point **P**. The points $P_1$ and $P_4$ locating the opposite sides of $P_2$ and $P_3$, denote the remaining end points of the virtual lines, respectively. Then, the junction quality factor is:

$$Q_{\mathrm{J}} = \frac{|L_1| - \sigma_1^{||} - \sigma_2^{\perp}}{|VL_1|} \cdot \frac{|L_2| - \sigma_2^{||} - \sigma_1^{\perp}}{|VL_2|} \tag{18.1}$$

where $|VL_i|(i = 1, 2)$ are the lengths of the virtual lines, as shown in Figure 18.1. The standard deviations $\sigma_i^{||}$ and $\sigma_i^{\perp}$, incorporating the uncertainties in the line extraction process for the position of the end points of the line segments along and perpendicular to its direction respectively, may be replaced by constants without affecting the basic grouping algorithms [7]. In this chapter, the

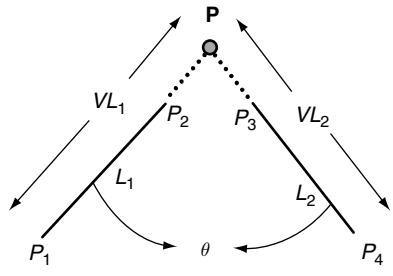



**Figure 18.1** The junction.

two variance factors $\sigma_i^{||}$ and $\sigma_i^{\perp}$ are ignored. The defined relation penalizes pairings in which either line is far away from the junction point. The quality factor also retains the symmetry property.

## 4. Energy Model for the Junction Groups

The relational representation, made from each contextual relation of the model and scene features, provides a reliable means to compute the correspondence information in the matching problem. Suppose that the model consists of $M$ feature nodes. Then, a linked node chain, given by the sequential connection of the nodes, can be constructed.

If the selected features are sequentially linked, then it is possible to calculate a potential energy from the enumerated feature nodes. For example, assume that any two-line features of the model correspond to two features $f_I$ and $f_{I+1}$ of the scene. If the relational configuration of each line node depends only on the connected *neighboring* nodes, then the energy potential obtained from the $M$ line nodes can be represented as:

$$E_{\text{total}}(f_1, f_2, \ldots, f_M) = E_1(f_1, f_2) + E_2(f_2, f_3) + \cdots + E_{M-1}(f_{M-1}, f_M) \tag{18.2}$$

where

$$E_I(f_I, f_{I+1}) = \sum_{k=1}^{K} |r_{2,k}(f_I, f_{I+1}) - R_{2,k}(I, I+1)| \tag{18.3}$$

Here, $r_{2,k}$ and $R_{2,k}$ denote the binary relations of any two connected line features of the scene and the model, respectively. The parameter $K$ is the number of binary relations.

For the relational representation of junctions, the model and scene node $I$ and $f_I$ in Equations (18.2) and (18.3) are replaced by the model junction and corresponding scene junction, respectively. Figure 18.2(a) presents a schematic of lines consisting of an object. Figure 18.2(b) shows the binary relations of sequentially connected junctions for line pattern matching. Equation (18.3) for junction chains can be rewritten accordingly as:

$$E_I(f_I, f_{I+1}) = \alpha \cdot |\theta(f_I) - \Theta(I)| + \beta \cdot |r(f_I, f_{I+1}) - R(I, I+1)| \tag{18.4}$$

Each junction has the *unary* angle relation from two lines constituting a single junction, as shown in the first term of Equation (18.4) and in Figure 18.1. $\theta(f_I)$ and $\theta(I)$ are corresponding junction angles in a scene and a model, respectively. We do not use a relation depending on line length, because lines in a noisy scene could be easily broken. The binary relation for the scene ($r$) and model ($R$) in the

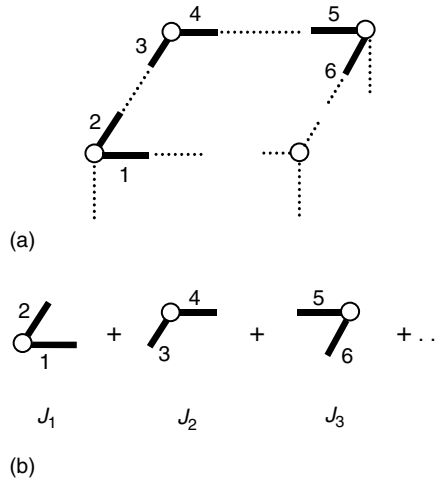



**Figure 18.2** Binary relations made from any two connected junction nodes: (a) line segments on a model; and (b) the combination of junctions by perceptual constraints, such as proximity, collinearity and parallelism.

second term is defined as a topological constraint or an angle relation between two junctions. For example, the following descriptions can represent the *binary* relations.

1. Two lines 1 and 4 should be approximately parallel (*parallelism*).
2. Scene lines corresponding to two lines 2 and 3 must be a *collinear* pair [6] or the same line. That is, two junctions are combined by the collinear constraint.
3. Line ordering for two junctions $J_1$, $J_2$ should be maintained, for example as *clockwise* or *counter-clockwise*, as the order of line 1, line 2, line 3 and line 4.

The relation defined by the connected two junctions includes all three perceptual organization groups that Lowe used in SCERPO. These local relations can be selectively imposed according to the type of the given problem. For example, a convex line triplet [15] is simply defined, by removing the above constraint 1 and letting line 2 and line 3 of constraint 2 be equal to each other. The weighting coefficients $\alpha$ and $\beta$ of the energy potential are experimentally given, by considering the variance factor of the line perturbation for image noise.

## 5. Energy Minimization

Dynamic Programming (DP) is an optimization technique good for problems where not all variables are interrelated simultaneously [8,16]. Suppose that the global energy can be decomposed into the following form:

$$E(f_1, \ldots, f_M) = E_1(f_1, f_2) + E_2(f_2, f_3) + \cdots + E_{M-1}(f_{M-1}, f_M) \tag{18.5}$$

in which $M$ is the number of the model nodes, such as lines or junctions, and $f_I$ is a scene label that can be assigned to the model node $I$.

Figure 18.3 shows a schematic DP diagram to find a trapezoidal model in the scene lines. Figure 18.3(a) presents a typical case in which we cannot define an ordering for the scene lines due

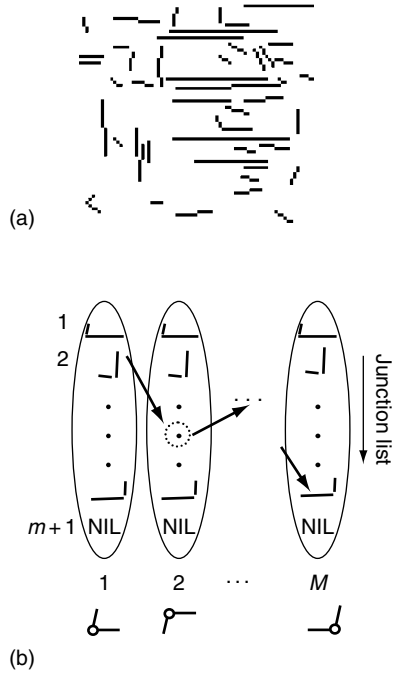



**Figure 18.3** The DP algorithm searches a scene node corresponding to each model node. A model feature can be matched to at least one node, among scene nodes, $1, \ldots, m+1$ of a column, including NULL node (NIL). (a) Line segments for the rear view of a vehicle; and (b) a DP-based search. $m$ is the number of junctions detected from (a) and $M$ is the number of predefined model junctions.

to the cluttered background. Therefore, it is difficult to extract a meaningful object boundary that corresponds to the given model. In this case, the DP-based search structure is formulated as the column in Figure 18.3(b), in which all detected scene features are simultaneously included in each column. Each junction of the model can get a corresponding junction in the scene as well as a null node, which indicates no correspondence. The potential matches are defined as the energy accumulation form of Equation (18.5). From binary relations of junctions (i.e. arrows in Figure 18.3(b)) defined between two neighboring columns, the local comparison-based method using the recursive energy accumulation table of Equation (18.5) can give a fast matching solution.

The DP algorithm generates a sequence that can be written in the recursive form. For $I = 1, \ldots, M-1$,

$$D_I(f_{I+1}) = \min_{f_I}[D_{I-1}(f_I) + E_I(f_I, f_{I+1})] \tag{18.6}$$

with $D_0(f_1) = 0$. The minimal energy solution is obtained by

$$\min_f E(f_1, \ldots, f_M) = \min_{f_M} D_{M-1}(f_M) \tag{18.7}$$

If each $f_I$ takes on $m$ discrete values, then to compute $D_{I-1}(f_I)$ for each $f_I$ value, one must evaluate the summation $[D_{I-1}(f_{I-1}) + E_{I-1}(f_{I-1}, f_I)]$ for the $m$ different $f_{I-1}$ values. Therefore, the overall minimization involves $(M-1)m^2$ evaluations of the summations. This is a large reduction from the exhaustive search for the total evaluation of $E(f_1, \ldots, f_M)$. $m$ is the number of junctions satisfying a threshold for the junction quality in the scene.

## 6. Collinear Criterion of Lines

Extraction of the image features such as points or lines is influenced by the conditions during image acquisition. Because the image noise distorts the object shape in the images, we need to handle the effect of the position perturbation in the features, and to decide a threshold or criterion to discard and reduce the excessive noise. In this section, the noise model and the error propagation for the collinearity test between lines are proposed. The Gaussian noise distribution for two end points of a line is an effective and general approach, as referred to in Haralick [17] and Roh [18], etc. In this section, we use the Gaussian noise model to compute error propagation for two-line collinearity and obtain a threshold value resulting from the error variance test to decide whether the two lines are collinear or not.

The line collinearity can be decomposed into two terms of parallelism and normal distance defined between the two lines being evaluated.

### *6.1 Parallelism*

The parallelism is a function of eight variables:

$$p = \cos^{-1}\left(\frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}|\,|\mathbf{b}|}\right), \text{ where } \mathbf{a} = \mathbf{x}_2 - \mathbf{x}_1 \text{ and } \mathbf{b} = \mathbf{x}_4 - \mathbf{x}_3 \tag{18.8}$$

$$\text{or } p = p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4), \text{ where } \mathbf{x}_i = (x_i, y_i, 1)^{\mathrm{T}} \tag{18.9}$$

The $\mathbf{x}_i (i = 1, \ldots, 4)$ denote image coordinates for four end points of the two lines and $|\mathbf{a}|$ presents the length of vector $\mathbf{a}$. To avoid the treatment of a trigonometric function in calculating the partial derivatives of function $p$ with respect to the image coordinates, we use a simpler function:

$$p' = \cos(p) = \frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}|\,|\mathbf{b}|} = p'(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) \tag{18.10}$$

Let $(x_i, y_i)$ be the true value and $(\tilde{x}_i, \tilde{y}_i)$ be the noisy observation of $(x_i, y_i)$, then we have

$$\tilde{x}_i = x_i + \xi_i \tag{18.11a}$$

$$\tilde{y}_i = y_i + \eta_i \tag{18.11b}$$

where the noise terms $\xi_i$ and $\eta_i$ denote independently distributed noise terms having mean 0 and variance $\sigma_i^2$. Hence:

$$E[\xi_i] = E[\eta_i] = 0 \tag{18.12}$$

$$V[\xi_i] = V[\eta_i] = \sigma_i^2 \tag{18.13}$$

$$E\left[\xi_i \xi_j\right] = \begin{cases} \sigma_0^2 & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \quad E\left[\eta_i \eta_j\right] = \begin{cases} \sigma_0^2 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{18.14a}$$

$$E\left[\xi_i \eta_j\right] = 0 \tag{18.14b}$$

From these noisy measurements, we define the noisy parallel function,

$$\tilde{p}'(\tilde{x}_1, \tilde{y}_1, \tilde{x}_2, \tilde{y}_2, \tilde{x}_3, \tilde{y}_3, \tilde{x}_4, \tilde{y}_4) \tag{18.15}$$

To determine the expected value and variance of $\tilde{p}'$, we expand $\tilde{p}'$ as a Taylor series at $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$:

$$\tilde{p}' \approx p' + \sum_{i=1}^{4}\left[(\tilde{x}_i - x_i)\frac{\partial \tilde{p}'}{\partial \tilde{x}_i} + (\tilde{y}_i - y_i)\frac{\partial \tilde{p}'}{\partial \tilde{y}_i}\right] = p' + \sum_{i=1}^{4}\left[\xi_i \frac{\partial \tilde{p}'}{\partial \tilde{x}_i} + \eta_i \frac{\partial \tilde{p}'}{\partial \tilde{y}_i}\right] \tag{18.16}$$

Then, the variance of the parallel function becomes:

$$\mathrm{Var}(p') = E\left[(\tilde{p}' - p')^2\right] = \sigma_0^2 \sum_{i=1}^{4}\left[\left(\frac{\partial \tilde{p}'}{\partial \tilde{x}_i}\right)^2 + \left(\frac{\partial \tilde{p}'}{\partial \tilde{y}_i}\right)^2\right] \tag{18.17}$$

Hence, for a given two lines, we can determine a threshold:

$$\Delta p = 3 \cdot \sqrt{E[(\tilde{p}' - p')^2]} \tag{18.18}$$

Because the optimal $p'$ equals 1, any parallel two lines have to satisfy the following condition:

$$1 - \tilde{p}' \leq \Delta p \tag{18.19}$$

## 6.2 Normal Distance

A normal distance between any two lines is selected from among two distances $d_1$ and $d_2$:

$$d_{\mathrm{norm}} = \max(d_1, d_2) \tag{18.20}$$

where

$$d_1 = \frac{|a_1 x'_m + b_1 y'_m + c_1|}{\left(a_1^2 + b_1^2\right)^{\frac{1}{2}}} \tag{18.21a}$$

$$d_2 = \frac{|a_2 x_m + b_2 y_m + c_2|}{\left(a_2^2 + b_2^2\right)^{\frac{1}{2}}} \tag{18.21b}$$

The $a_i, b_i$ and $c_i$ are line coefficients for the $i$-line and $(x_m, y_m)$ denotes the center point coordinate of the first line, and $(x'_m, y'_m)$ denotes the center of the second line. Similarly to the parallel case of Section 6.1, the normal distance is also a function of eight variables:

$$d_{\mathrm{norm}} = d(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) \tag{18.22}$$

Through all processes similar to the noise model of Section 6.1, we obtain:

$$\mathrm{Var}(p') = E\left[(\tilde{p}' - p')^2\right] = \sigma_0^2 \sum_{i=1}^{4}\left[\left(\frac{\partial \tilde{p}'}{\partial \tilde{x}_i}\right)^2 + \left(\frac{\partial \tilde{p}'}{\partial \tilde{y}_i}\right)^2\right] \tag{18.23}$$

For the given two lines, we can also determine a threshold for the normal distance:

$$\Delta d = 3 \cdot \sqrt{E[(\tilde{d} - d)^2]} \tag{18.24}$$

Because the optimal $d$ equals 0, the normal distance for any two collinear lines has to satisfy the following condition:

$$\tilde{d} \leq \Delta d \tag{18.25}$$

## 7. Energy Model for the Junction Groups

In this section, we test the robustness of the junction detection algorithm by counting the number of detected junctions as a function of the junction quality $Q_J$ of Equation (18.1). Figure 18.4 shows some images of 2D and 3D objects under well-controlled lighting conditions and a cluttered outdoor scene. We use Lee's method [19] to extract lines.

Most junctions (i.e. more than 80 %) extracted from possible combinations of the line segments are concentrated in the range 0.0 ~ 1.0 of the quality measure, as shown in Figure 18.5. The three experimental sets of Figure 18.4 give similar tendencies except for a small fluctuation at the quality measure 0.9, as shown in Figure 18.5. At the quality level 0.5, the occupied portion of the junctions relative to the whole range drops to less than 1 %.

When robust line features are extracted, $Q_J$, as a threshold for the junction detection, does not severely influence the number of extracted junctions. In good conditions, the extracted lines are clearly defined along the object boundary and few cluttered lines exist in the scene, as shown in Figure 18.4(a). Therefore, the extracted junctions are accordingly defined and give junctions with a high junction quality factor, as shown in Figure 18.4(a). The parts plot in Figure 18.5 graphically shows the high-quality junctions as the peak concentrated in the neighbor range of quality measure 0.9.

For $Q_J = 0.7$, the detection ratio 1.24 (i.e. number of junctions/number of primitive lines) of Figure 18.4(a) is decreased to 0.41 for the outdoor scene of Figure 18.4(c), indicating the increased effect of the threshold (see also Table 18.1). The effect of threshold level for the number of junctions resulting from distorted and broken lines is more sensitive to the outdoor scenes. That is, junction detection

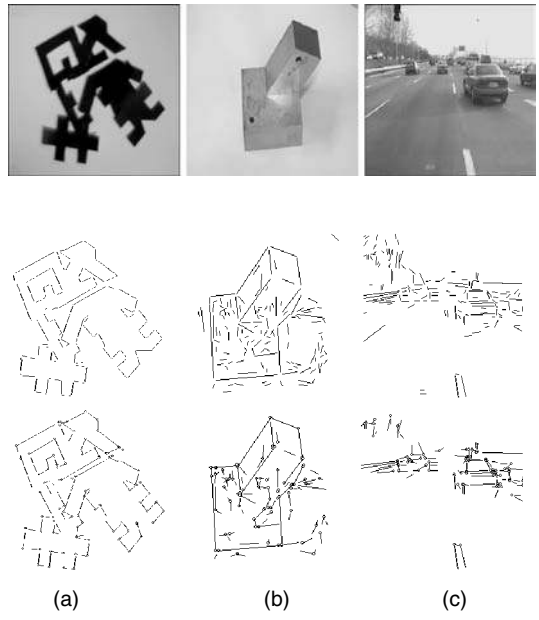



**Figure 18.4** Junction extraction: the number of junctions depends on the condition of the images. Each column consists of an original image, line segments and junctions and their intersecting points for quality measure 0.5, respectively. The small circles in the figure represent the intersection points of two-line junctions. (a) Parts: a 2D scene under controlled lighting conditions; (b) blocks: an indoor image with good lighting; and (c) cars: a cluttered image under an uncontrolled outdoor road scene.

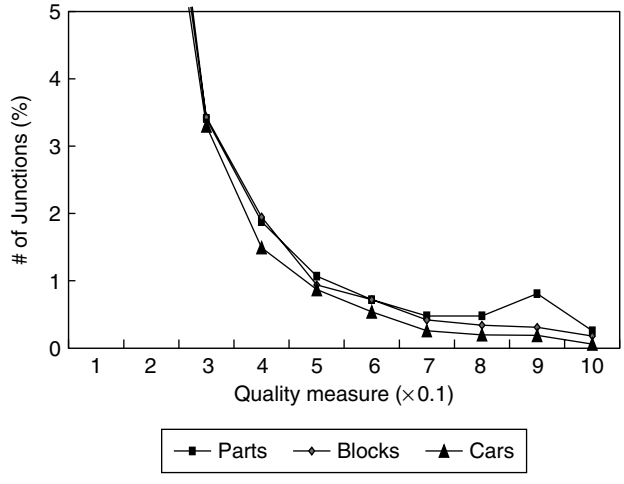



**Figure 18.5** The occupying percentage of junctions according to changes of the quality measure.

**Table 18.1** Junction number vs. quality measure.

| | | $Q_J = 0.3$ | | $Q_J = 0.5$ | | $Q_J = 0.7$ | |
|---|---|---|---|---|---|---|---|
| | # *of lines* ($L_n$) | # *of junctions* ($V_n$) | $\frac{V_n}{L_n}$ | $V_n$ | $\frac{V_n}{L_n}$ | $V_n$ | $\frac{V_n}{L_n}$ |
| Parts | 100 | 376 | 3.78 | 196 | 1.96 | 124 | 1.24 |
| Blocks | 130 | 543 | 4.11 | 196 | 1.51 | 90 | 0.7 |
| Cars | 137 | 466 | 3.4 | 180 | 1.31 | 56 | 0.41 |

under uncontrolled lighting conditions has a higher dependence on the change of junction quality as a detection threshold.

With some additional experiments, we identify that the number of junctions in scenes does not vary much, in spite of a low $Q_J$. Usually, a junction quality of 0.5 is sufficient to give an adequate number of junctions for most test scenes while not skipping the salient junctions, and without increasing the time complexity of the DP-based search.

## 8. Experiments

We applied the proposed method to find a group of optimal line junctions. Test scenes included some lines distorted by a variation of viewpoint. First, input images were processed to detect only the strongest step edges. Edges were further filtered by discarding shorter edge segments. Junctions were then inferred between the remaining line segments. When junctions were extracted from the lines, then relative relations between the junctions were searched by using the criterion of Section 4, with the collinear constraint that links two junctions.

To reduce the repeated computation for relations between line segments, all possible relations such as inter-angles, collinear properties and junction qualities between lines were previously computed.

## 8.1 Line Group Extraction

As an example of 2D line matching for a viewpoint variation, the rear-view of a vehicle was used. The description of the model lines was given as a trapezoidal object. The model pattern could have a clockwise combination of the constituting lines. Figure 18.6(a-1) shows the first and the last image among a sequence of 30 images to be tested. With the cluttered background lines, a meaningful boundary extraction of the object of interest was difficult, as shown in Figure 18.6(a-2). Figure 18.6 (a-3) shows the extraction of junctions in the two frames. A threshold for the quality measure $Q_J$ was set at 0.5. Figure 18.6(a-4) shows optimal matching lines having the smallest accumulation energy of Equation (18.7). In spite of some variations from the model shape, a reasonable matching result was obtained. Unary and binary properties of Equation (18.4) were both used. Figure 18.6(b) shows a few optimal matching results. In Figure 18.6(b), the model shape is well matched as the minimum DP energy of Equation (18.7), in spite of the distorted shapes in the scenes. Matching was successful for 25 frames out of 30 – a success ratio of 83 %. Failing cases result from line extraction errors in low-level processing, in which lines could not be defined on the rear windows of vehicles.

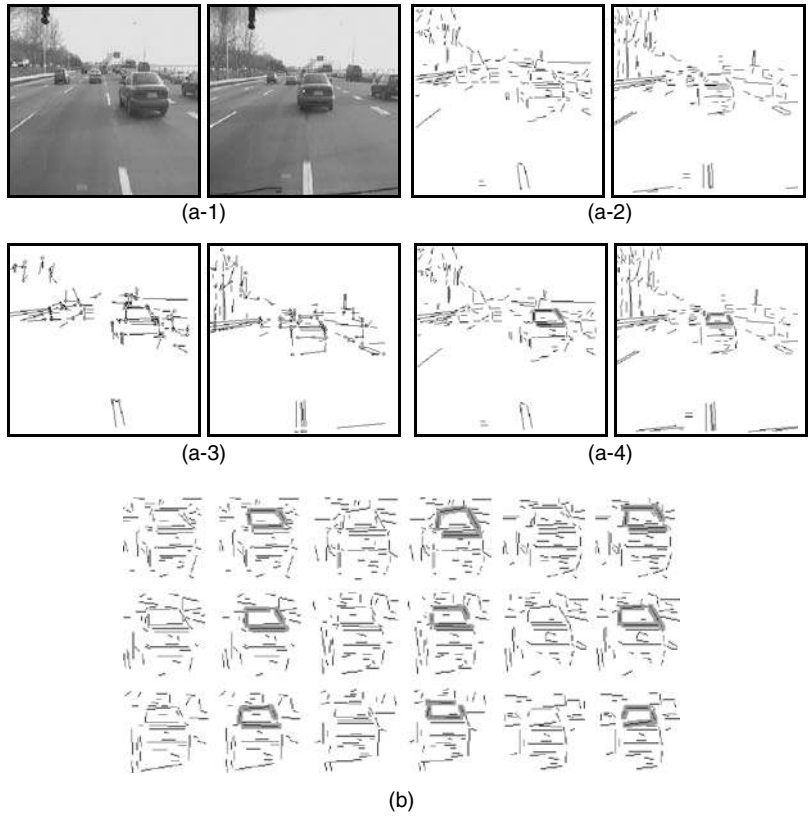



**Figure 18.6** Object matching under weak perspective projection: a rear-window of a vehicle on the highway was used. (a-1) The first and last images to be tested; (a-2) line extraction; (a-3) junction detection for $Q_J = 0.5$; (a-4) optimal model matching; (b) a few optimal matching results between the first and last images.

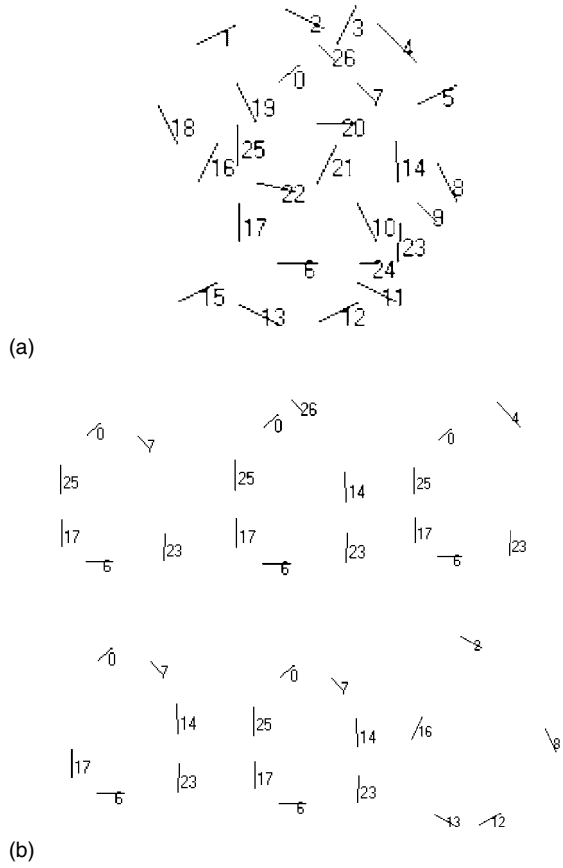



**Figure 18.7** Object matching in a synthetic image with broken and noisy lines.

Figure 18.7 shows experimental results for extracting a 2D polyhedral object. Figure 18.7(a) shows a model as a pentagon shape with noisy and broken lines in the background region. All lines except for the pentagon were randomly generated. Figure 18.7(b) shows a few matching results with small DP energy. A total of six candidates were extracted as the matched results for generating hypotheses for the object recognition. Each one is similar to the pentagon model shape. It is interesting to see the last result because we did not expect this extraction.

Two topological combinations of line junctions are shown as model shapes in Figure 18.8. $J_1$ and $J_2$ junctions are combined with a collinear constraint that also denotes the same rotating condition as the clockwise direction in the case of Figure 18.8(a). The three binary relations in Section 4 all appear in the topology of Figure 18.8. In the combination of $J_2$ and $J_3$, the rotating direction between the two junctions is reversed. In Figure 18.8(b), similar topology to Figure 18.8(a) is given, except for the difference of rotating direction of the constituting junctions. Figure 18.9 presents an example of extracting topological line groups to guide 3D object recognition. The topological shapes are invariant to wide changes of view. That is, if there is no self-occlusion on the object, the interesting line groups are possible to extract. Figure 18.9(a) shows the original image to be tested. After discarding the shorter lines, Figure 18.9(b) presents the extracted lines with the numbering indicating the line index, and Figure 18.9(c) and 18.9(d) give the matched line groups corresponding to the model shape of

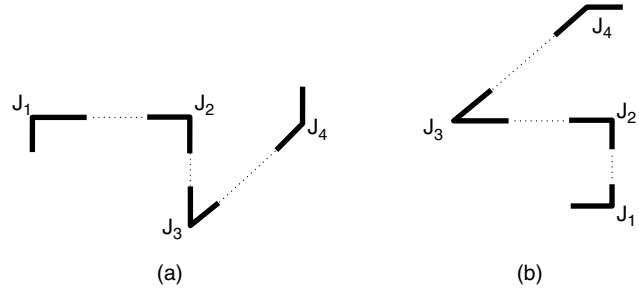



**Figure 18.8** Topological shapes for model description. (a) A model with clockwise rotation of the starting junction; (b) a model with counter-clockwise direction for the starting junction.

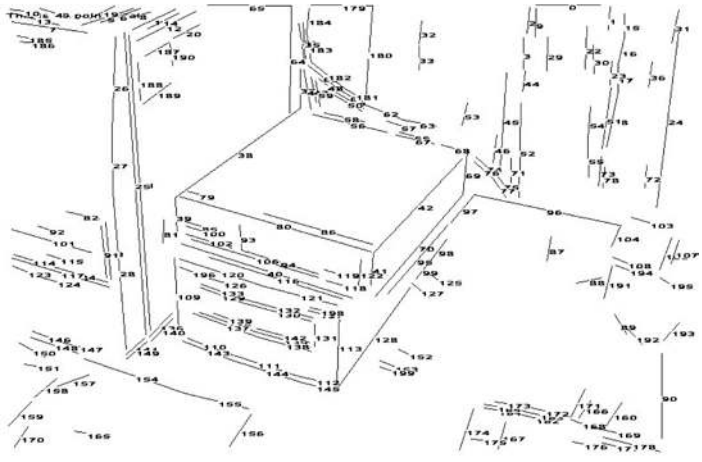
Figure 18.8(a) and 18.8(b), respectively. In each extraction, there are enough line groups to guide a hypothesis for 3D object recognition.

Table 18.2 presents the matching results in Figure 18.9 corresponding to the models of Figure 18.8. All extractions are enumerated for each model.

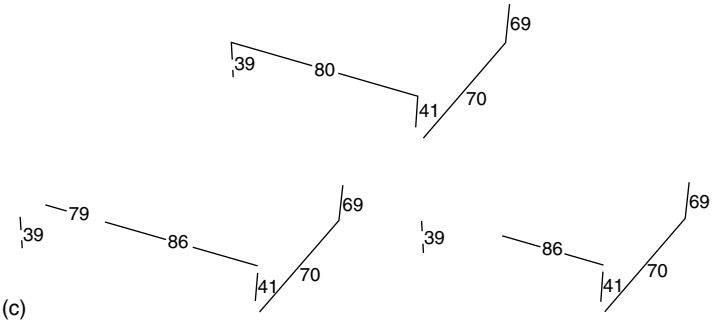
The above experimental results demonstrate that the proposed framework can find similar shapes even from a cluttered and distorted line set. By the local comparison-based matching criterion permitting a shape distortion for the reference model shape, reasonable line grouping and shape matching are possible, without increasing the time complexity. The grouping and the junction detection are all performed in 1 s, on a Pentium II-600 desktop machine. In all test sets, any length ratio of the model or scene lines has not been used, because the scene lines are easily broken in outdoor images. Angle relations and line ordering between two neighboring junctions are well preserved, even in broken and distorted line sets.

(a)

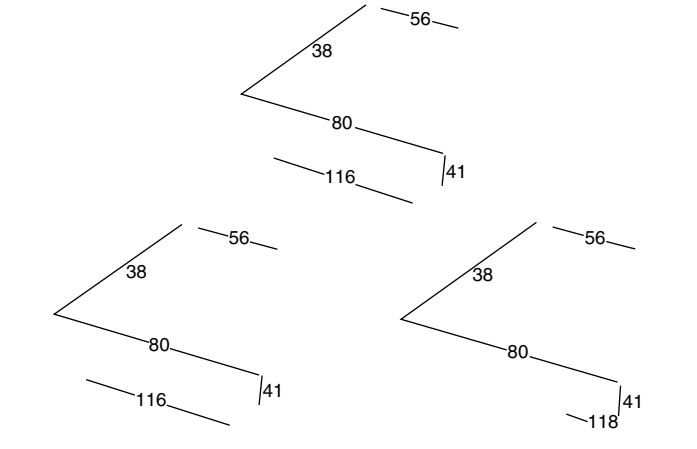
**Figure 18.9** A topological shape extraction for 3D object recognition. (a) Original image; (b) line extraction; and (c), (d) found topological shapes.



(b)




(c)




(d)

**Figure 18.9** (*continued*)

**Table 18.2** The topological matching results.

| | $J_1$ | $J_2$ | $J_3$ | $J_4$ |
|---|---|---|---|---|
| Model 1 | (39, 80) | (80, 41) | (41, 70) | (70, 69) |
| | (39, 79) | (86, 41) | (41, 70) | (70, 69) |
| | (39, 86) | (86, 41) | (41, 70) | (70, 69) |
| Model 2 | (118, 41) | (41, 80) | (80, 38) | (38, 56) |
| | (40, 41) | (41, 80) | (80, 38) | (38, 56) |
| | (106, 41) | (41, 80) | (80, 38) | (38, 56) |
| | (94, 41) | (41, 80) | (80, 38) | (38, 56) |
| | (116, 41) | (41, 80) | (80, 38) | (38, 56) |

## *8.2 Collinearity Tests for Random Lines*

We tested the stability of the two collinear functions proposed in Section 6 by changing the standard deviation as a threshold for four end points of the two constituting lines. The noise was modeled as Gaussian random noise having mean 0 and variance $\sigma_0^2$.

A total of 70 lines were randomly generated in a rectangular region of size $100 \times 100$ in Figure 18.10, hence the number of possible line pairs was ${}_{70}C_2 = 2415$. Finally, only two line pairs were selected as satisfying the two collinear conditions of Equation (18.19) and Equation (18.25) under $\sigma_0 = 0.1$. When we reduced the variation value, there were a few collinear sets. By a simple definition of collinear functions and control of the variance $\sigma_0$ as Gaussian perturbation, we could systematically obtain the collinear line set without referring to heuristic parameters.

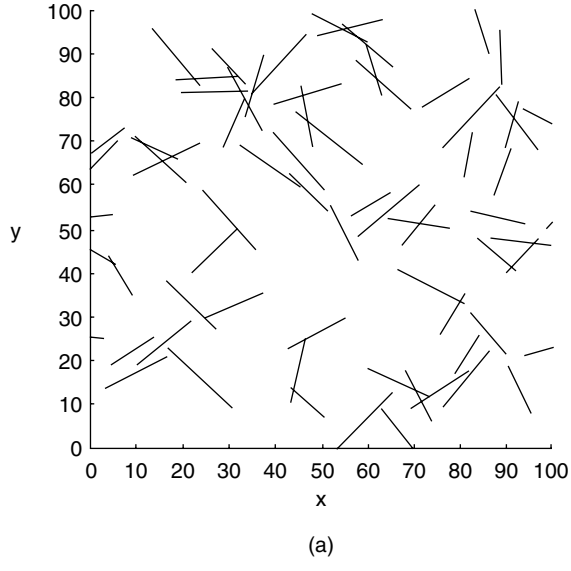



**Figure 18.10** Collinear lines according to the change of standard deviation $\sigma_0$ as a threshold. (a) The randomly generated original line set; (b) line pairs detected for $\sigma_0 = 0.4$; (c) for $\sigma_0 = 0.3$; and (d) for $\sigma_0 = 0.1$.

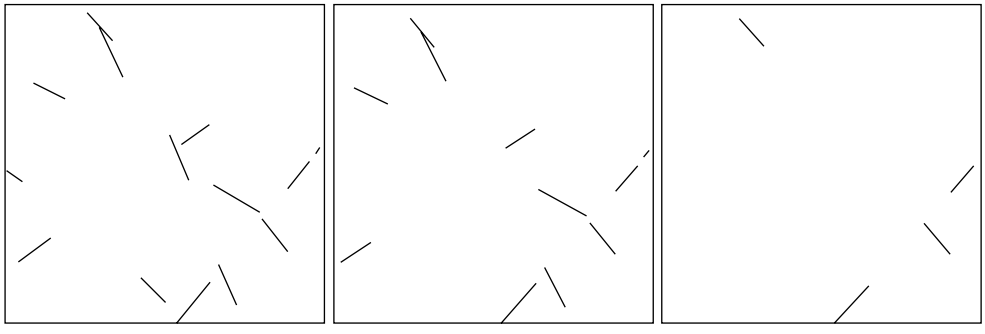



**Figure 18.10** (*continued*)

## 9. Conclusions

In this chapter, a fast and reliable matching and grouping method using dynamic programming is proposed to extract collections of salient line segments. We have considered the classical dynamic programming as an optimization technique for geometric matching and grouping problems. First, the importance of grouping to object recognition was emphasized. By grouping together line features that are likely to have been produced by a single object, it has long been known that significant speed-ups in a recognition system can be achieved, compared to performing a random search. This general fact was used as a motive to develop a new feature grouping method. We introduced a general way of representing line patterns and of using the patterns to consistently match 2D and 3D objects.

The main element in this chapter is a DP-based formulation for matching and grouping of line patterns by introducing a robust and stable geometric representation that is based on the perceptual organizations. The end point proximity and collinearity comprised of image lines are introduced as two main perceptual organizing groups to start the object matching or recognition. We detect the junctions as the end point proximity for the grouping of line segments. Then, we search a junction group again, in which each junction is combined by the collinear constraint between them. These local primitives, by including Lowe's perceptual organizations and acting as the search node, are consistently in a linked form in the DP-based search structure.

We could also impose several constraints, such as parallelism, the same line condition and rotational direction, to increasingly narrow down the search space for possible objects and their poses. The model description is predefined for comparison with the scene relation. The collinear constraint acts to combine the two junctions as a neighborhood for each other. The DP-based search algorithm reduces the time complexity for the search of the model chain in the scene.

Through experiments using images from cluttered scenes, including outdoor environments, we have demonstrated that the method can be applied to the optimal matching, grouping of line segments and 2D/3D recognition problems, with a simple shape description sequentially represented.

## References

[1] Ayache, N. and Faugeras, O. D. "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(1), pp. 44–54, 1986.

[2] Ballard, D. H. and Brown, C. M. *Computer Vision*, Prentice Hall, 1982.

[3] Grimson, W. E. L. and Lozano-Perez, T. "Localizing overlapping parts by searching the interpretation tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**, pp. 469–482, 1987.

[4] Hummel, R. A. and Zucker, S. W. "On the foundation of relaxation labeling processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**(3), pp. 267–286, 1983.

[5] Li, S. Z. "Matching: invariant to translations, rotations and scale changes," *Pattern Recognition*, **25**, pp. 583–594, 1992.

[6] Lowe, D. G. "Three-Dimensional Object Recognition from Single Two-Dimensional Images," *Artificial Intelligence*, **31**, pp. 355–395, 1987.

[7] Etemadi, A., Schmidt, J. P., Matas, G., Illingworth, J. and Kittler, J. "Low-level grouping of straight line segments," in *Proceedings of 1991 British Machine Vision Conference*, pp. 118–126, 1991.

[8] Fischler, M. and Elschlager, R. "The representation and matching of pictorial structures," *IEEE Transactions on Computers*, **C-22**, pp. 67–92, 1973.

[9] Grimson, W. E. L. and Huttenlocher, D. "On the Sensitivity of the Hough Transform for Object Recognition," *Proceedings of the Second International Conference on Computer Vision*, pp. 700–706, 1988.

[10] Li, S. Z. *Markov Random Field Modeling in Computer Vision*, Springer Verlag, New York, 1995.

[11] Bunke, H. and Buhler, U. "Applications of Approximate String Matching to 2D Shape Recognition," *Pattern Recognition*, **26**, pp. 1797–1812, 1993.

[12] Cox, I. J., Higorani, S. L. and Rao, S. B. "A Maximum Likelihood Stereo Algorithm," *Computer Vision and Image Understanding*, **63**(3), pp. 542–567, 1996.

[13] Ohta, Y. and Kanade, T. "Stereo by intra- and inter- scanline search using dynamic programming," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **7**(2), pp. 139–154, 1985.

[14] Jacobs, D. W. "Robust and Efficient Detection of Convex Groups," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–771, 1993.

[15] Henikoff, J. and Shapiro, L. G. "Representative Patterns for Model-based Matching," *Pattern Recognition*, **26**, pp. 1087–1098, 1993.

[16] Amini, A. A., Weymouth, T. E. and Jain, R. C. "Using dynamic programming for solving variational problems in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(9), pp. 855–867, 1990.

[17] Haralick, Y. S. and Shapiro, R. M. "Error Propagation in Machine Vision," *Machine Vision and Applications*, **7**, pp. 93–114, 1994.

[18] Roh, K. S. and Kweon, I. S. "2-D object recognition using invariant contour descriptor and projective invariant," *Pattern Recognition*, **31**(4), pp. 441–455, 1998.

[19] Lee, J. W. and Kweon, I. S. "Extraction of Line Features in a Noisy Image," *Pattern Recognition*, **30**(10), pp. 1651–1660, 1997.

# 19

# Holo-extraction and Intelligent Recognition of Digital Curves Scanned from Paper Drawings

**Ke-Zhang Chen**
Department of Mechanical Engineering, The University of Hong Kong, Hong Kong, China

**Xi-Wen Zhang**
Laboratory of Human Computer Interaction and Intelligent Information Processing, Institute of Software, the Chinese Academy of Sciences, Beijing 100080, China

**Zong-Ying Ou**

**Xin-An eng**
School of Mechanical Engineering, Dalian University of Technology, Dalian, 116024, China

*This chapter introduces a holo-extraction method of information from paper drawings, i.e. the networks of Single Closed Regions (SCRs) of black pixels, which not only provide a unified base for recognizing both annotations and the outlines of projections of parts, but can also build the holo-relationships among SCRs so that it is convenient to extract lexical, syntactic and semantic information in the subsequent phases for 3D reconstruction. Based on the holo-extraction method, this chapter further introduces an intelligent recognition method of digital curves scanned from paper drawings for subsequent pattern recognition and 3D reconstruction.*

## 1. Introduction

In order to survive worldwide competition, enterprises have tried to use the computer's huge memory capacity, fast processing speed and user-friendly interactive graphics capabilities to automate and tie together cumbersome and separate engineering or production tasks, including design, analysis,

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

optimization, prototyping, production planning, tooling, ordering materials, programming for Numerically Controlled (NC) machines, quality control, robot assembly and packaging. The technologies for all these tasks rely on three-dimensional (3D) computer feature models of products made during design. Some advanced computer-aided design systems have been developed so that people can use these systems to build 3D computer feature models for new product designs, and production without drawings has thus appeared in some enterprises of developed countries. But enterprises have had a great deal of two-dimensional (2D) mechanical paper drawings made in the past for their existing products, which need to be converted to 3D computer feature models for applications. The advanced computer-aided design systems can easily convert 3D computer models made within them into their 2D orthogonal projections, but they cannot convert the other way round. How to convert 2D mechanical paper drawings into 3D computer feature models is one of the major issues many enterprises are very concerned about.

In order to convert 2D mechanical paper drawings into 3D computer feature models, the 2D paper drawings are first scanned by an optical scanner and then the scanned results are input to a computer in the form of raster (binary) images. The conversion from the raster image to 3D model needs two processes: understanding and 3D reconstruction. The research on the understanding process has been implemented following three phases [1]:

1. *The lexical phase*. The raster image is converted into vectorized information, such as straight lines, arcs and circles.
2. *The syntactic phase*. The outlines of orthographic projections of a part and the annotations are separated; the dimension sets, including their values of both the nominal dimensions and the tolerances, are aggregated; the crosshatching patterns are identified; and the text is recognized.
3. *The semantic phase*. Much semantic information needed for 3D reconstruction is obtained by functional analyses of each view, including the analysis of symmetries, the recognition of technologically meaningful entities from symbolic representation (such as bearing and threading), and so on.

After the understanding process, a 3D computer model will then be reconstructed by using geometric matching techniques with the recognized syntactic and semantic information.

Up to now, the research on the conversion from 2D paper drawings to 3D computer feature models has been stuck in low-level coding: essential vectorization, basic layer separation and very limited symbol recognition [1]. One of the reasons for this is that the three phases of the understanding process have been isolated and people have been doing their research on only one of the phases since the whole conversion is very complicated and more difficult. For instance, the vectorization methods were developed only for getting straight lines, arcs, circles, etc., so that much information contained in the drawing was lost after the vectorization. Also, in some research, different methods were developed and applied for recognizing the text and the outlines of orthographic projections of parts, respectively. In fact, the 3D reconstruction needs not only the vectors themselves but also their relationships and the information indicated by various symbols, from which the syntactic and semantic information can be extracted later on. This chapter introduces a holo-extraction method of information from paper drawings, i.e. the networks of Single Closed Regions (SCRs) of black pixels, which not only provide a unified base for recognizing both annotations and the outlines of projections of parts, but also build the holo-relationships among SCRs so that it is convenient to extract lexical, syntactic and semantic information in the subsequent phases for 3D reconstruction. Based on the holo-extraction method, this chapter further introduces an intelligent recognition method of digital curves scanned from paper drawings for subsequent pattern recognition and 3D reconstruction.

## 2. Review of Current Vectorization Methods

Vectorization is a process that finds the vectors (such as straight lines, arcs and circles) from the raster images. Much research work on this area has been done, and many vectorization methods and their software have been developed. Although the vectorization for the lexical phase is more

mature than the technologies used for the other two higher level phases, it is yet quite far from being perfect.

Current vectorization methods can be categorized into six types: Hough Transform (HT)-based methods [2], thinning-based methods, contour-based methods, sparse pixel-based methods, mesh pattern-based methods and black pixel region-based methods.

## 2.1 The Hough Transform-based Method

This visits each pixel of the image in the $(x, y)$ plane, detects peaks in its transform $(m, c)$ space, and uses each peak to form a straight line defined by the following equation:

$$y = mx + c \tag{19.1}$$

where $m$ is its slope and $c$ is its intercept. Since the slopes and intercepts are sampled sparsely, they may not be as precise as the original straight lines. Moreover, this method cannot generate polylines [3].

## 2.2 Thinning-based Methods

Thinning is a process that applies certain algorithms to the input raster image and outputs one-pixel-wide skeletons of black pixel regions [4–7]. Three types of algorithm have been developed, i.e. iterative boundary erosion [5], distance transform [6] and adequate skeleton [7]. Although their speeds and accuracies are different, they all have disadvantages: high time complexities, loss of shape information (e.g. line width), distortions at junctions and false and spurious branches [3].

## 2.3 The Contour-based Method

This first finds the contour of the line object and then calculates the middle points of the pair of points on two opposite parallel contours or edges [8–10]. Although it is much faster than thinning-based methods and the line width is also much easier to obtain, joining up the lines for a merging junction or a cross intersection is problematic, and it is inappropriate for use in vectorization of curved and multi-crossing lines [3].

## 2.4 The Sparse Pixel-based Method

Here, the basic idea is to track the course of a one-pixel-wide 'beam of light', which turns orthogonally each time when it hits the edge of the area covered by the black pixels, and to record the midpoint of each run [11]. With some improvement based on the orthogonal zig-zag, the sparse pixel vectorization algorithm can record the medial axis points and the width of run lengths [3].

## 2.5 Mesh Pattern-based Methods

These divide the entire image using a certain mesh and detect characteristic patterns by only checking the distribution of the black pixels on the border of each unit of the mesh [12]. A control map for the image is then prepared using these patterns. Finally, the extraction of long straight-line segments is performed by analyzing the control map. This method not only needs a characteristic pattern database, but also requires much more processing time. Moreover, it is not suitable for detection of more complex line patterns, such as arcs and discontinuous (e.g. dashed or dash-dotted) lines [3].

## 2.6 Black Pixel Region-based Methods

These construct a semi-vector representation of a raster image first and then extract lines based on the semi-vector representation. The semi-vector representations can be a run graph (run graph-based methods)[13], a rectangular graph [14] or a trapezoidal graph [15]. A run is a sequence of black pixels in either the horizontal or vertical direction. A rectangle or trapezoid consists of a certain set of runs.

## 2.7 The Requirements for Holo-extraction of Information

It can be seen from the current vectorization methods that, except for black pixel region graph-based methods, other methods are mainly focused on the speed and accuracy of generating vectors themselves, not on holo-extraction of information. Although black pixel region graph-based methods build certain relationships between constructed runs, rectangles or trapezoids, the regions are so small that it is not appropriate for curve line vectorization and it is difficult to construct the relationships among vectors.

In fact, the understanding process for its subsequent 3D reconstruction is an iterative process for searching different level relationships and performing corresponding connections. For instance, linking certain related pixels can form a vector. Connecting a certain set of vectors can form primitives or characters of the text. Combining two projection lines and a dimension line containing two arrowheads with the values for normal dimension and tolerance can produce a dimension set, which can then be used with a corresponding primitive for parametric modeling. Aggregating the equal-spaced parallel thin lines and the contour of their area can form a section, which can then be used with certain section symbols for solid modeling. Connecting certain primitives referring to corresponding symbols recognized can form certain features (e.g. bearing and threading), which can be used for feature modeling. Matching primitives in different views according to orthogonal projective relationships can produce a 3D model. If the primitives extracted are accurate, their projective relationships can be determined by analyzing the coordinates of end points of these vectors. But the primitives extracted and their projective relationships are inaccurate in paper drawings, so that this method cannot be applied. It needs an expert system that simulates the experienced human designer's thinking mode to transform the inaccurate outlines of parts' orthographic projections into 3D object images, so that their relationships become more important and crucial. As mentioned in the first section of this chapter, the vectorization process in the first phase should not lose the information in a drawing or the information needed for 3D reconstruction, which are mainly different level relationships contained in the raster image. Accordingly, a holo-extraction of information from the raster image is needed. In order to facilitate the iterative process for searching different level relationships and performing corresponding connections, the method needs a compact representation of the raster image as a bridge from the raster image to understanding, which should satisfy the following requirements:

- it can distinguish different types of linking point for different relationships of the related elements (e.g. tangential point, intersecting point and merging junction) to provide necessary information for extracting lexical, syntactic and semantic information in the subsequent phases;
- it can provide a unified base for further recognizing both the outlines of orthogonal projections of parts and the annotations, and facilitate their separation;
- it can recognize line patterns and arrowheads, and facilitate the aggregation of related elements to form certain syntactic information, such as dimension sets;
- it can facilitate recognizing vectors quickly and precisely, including straight lines, arcs and circles;
- it can provide holo-graphs of all the elements as a base for the subsequent 3D reconstruction.

The networks of SCRs reported in this chapter are developed for these purposes.

## 3. Construction of the Networks of SCRs

The elements and workflow of constructing the networks of SCRs are shown in Figure 19.1, and illustrated in more detail as follows.

### *3.1 Generating Adjacency Graphs of Runs*

Let visiting a sequence be from top to bottom for a raster image and from left to right along the horizontal direction for each row. When visiting a row, the first black pixel converted from a white pixel is the starting point of a run, and the last black pixel, if the next pixel is white, is the end point of the run. The value of its end coordinates minus the starting coordinate in the horizontal direction is its run length, the unit of which is the pixel. If the difference of two runs' vertical coordinates is 1 and the value of their minimal end coordinates minus maximal starting coordinates is larger than or equal to 1, the two runs are adjacent. If A and B are adjacent and A's vertical coordinate is larger than that of B, A is called the predecessor of B, and B is called the successor of A. There are seven types of run according to adjacency relationships with other runs, as shown in Figure 19.2:

By recording the adjacency relationships of runs while visiting in the assumed way, the adjacency graphs of runs can be made using a node to represent a run, and a line between two nodes to

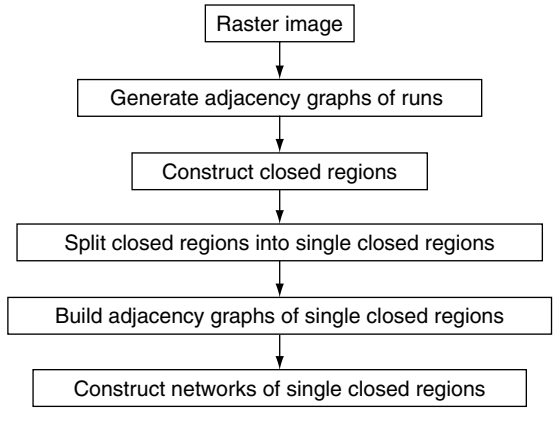



**Figure 19.1** Elements and workflow of constructing the networks of SCRs.

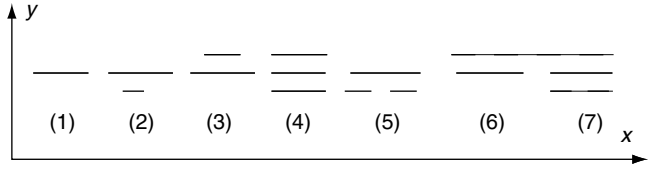



**Figure 19.2** Seven types of run. (1) Singular – has no predecessor and no successor; (2) beginning run – has no predecessor and only one successor; (3) end run – has only one predecessor and no successor; (4) regular run – has only one predecessor and only one successor; (5) branching run – has one predecessor at most and more than one successor; (6) merging run – has more than one predecessor and one successor at most; (7) cross run – has more than one predecessor and successor.

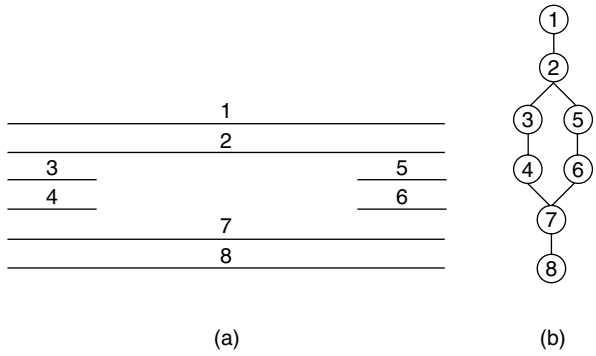



**Figure 19.3** The adjacency graph of runs for a run graph.

represent an adjacency relationship between the two runs. Different properties of runs (including their starting coordinates, end coordinates, lengths and types) are stored in the node data file. Figure 19.3(b) shows an adjacency graph of runs generated from the run graph in Figure 19.3(a). It is obvious that the adjacency graph of runs represents not only the runs themselves, but also their adjacency relationships.

## *3.2 Constructing Single Closed Regions (SCRs)*

Based on the adjacency graphs of runs constructed, a higher level representation of the raster image can be obtained by aggregating related runs into closed regions. These related runs must satisfy the following requirements:

- they are adjacent to each other;
- the beginning run is not a branching run or a cross run, and the end run is not a merging run or a cross run;
- the difference between their lengths is less than their shortest run length.

From the first run, the related runs can be searched via the adjacency graphs of runs, and their starting points and end points are then recorded to construct closed regions. Figure 19.4(c) shows a closed region graph constructed from the run graph in Figure 19.4(b) for the outlines in Figure 19.4(a).

According to the method, the possible results obtained may be the closed region for a point, a straight line, an arc, an arrowhead or a combined line, as shown in Figure 19.5(a), (b), (c), (d) and (e) respectively, which are identified using a fuzzy logic method [16]. A point may be an independent point, branching point, merging point, cross point or tangential point, as shown in Figures 19.5(a), 19.6(a), 19.6(b), 19.6(c) and 19.6(d) respectively. A straight line can be a sloping, vertical or horizontal straight line.

Although the closed region for a point, a straight line, an arc or an arrowhead is a single closed region, the closed region for a combined line is not an SCR, since a combined line may consist of straight lines and/or arcs, which are linked or tangential to each other. In order to construct the adjacency graphs of SCRs, the closed region for a combined line should be decomposed into several SCRs, each of which represents a straight line or an arc. The method for decomposition is to find their optimal split points with the least number of segments, each of which can be fitted to a straight line or an arc with least error. This is an optimization problem, which can be solved by using a genetic algorithm [17] and will be introduced in detail in Section 6. For example, with this method,

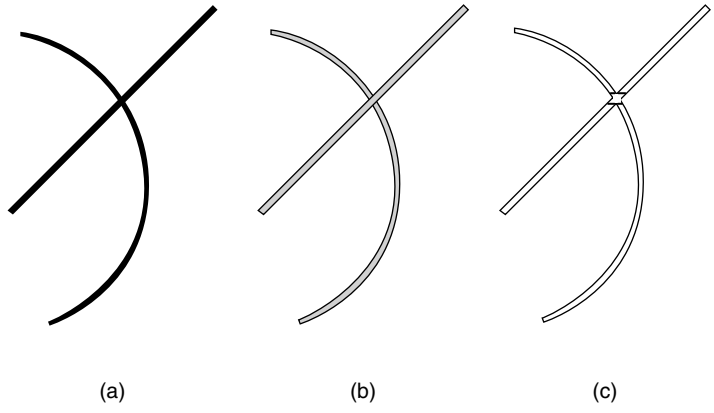



**Figure 19.4** A closed region graph.

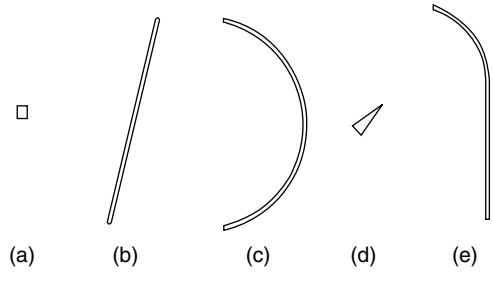



**Figure 19.5** Closed regions.

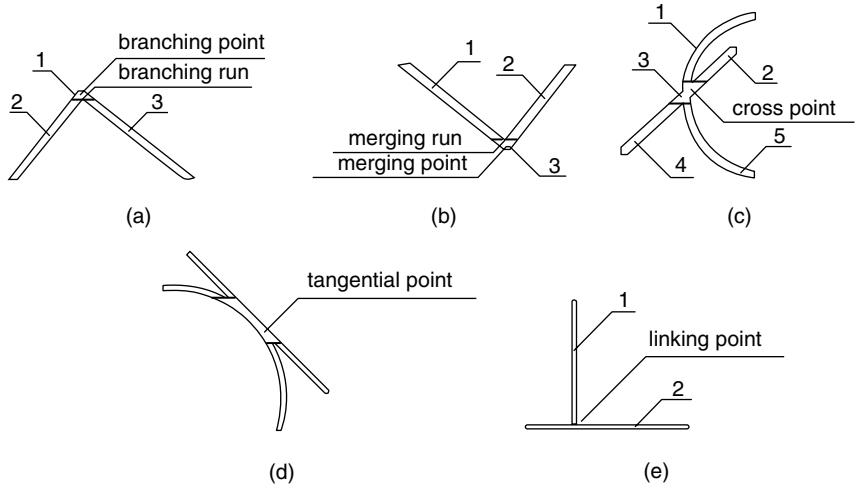



**Figure 19.6** Single closed regions for points.

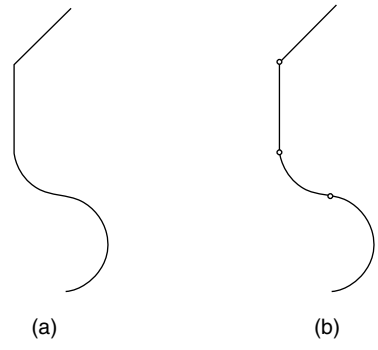



**Figure 19.7** The splitting of a combined line.

the combined line in Figure 19.7(a) can be split up into two straight lines and two arcs, as shown in Figure 19.7(b), where the split points are indicated by small circles.

After its split points are found, the closed region for a combined line can be decomposed into several SCRs. Thus, the original run graph can be transformed into the graph consisting of only SCRs. Each SCR represents a higher level element of primitives or annotations, so that the relationships among vectors can be extracted more easily.

## *3.3 Building Adjacency Graphs of SCRs*

Based on the adjacency graphs of runs and the single closed regions constructed, the adjacency graphs of SCRs for a raster image can be built using the following rules:

- use a node to represent a single closed region;
- add a black node for a split point of combined lines, or a linking point where there appears a branching run, a merging run or a run between which and its predecessor the length difference is larger than their individual shortest one, as indicated in Figure 19.6(e);
- apply a line between a node and a black node to represent an adjacency relationship between an SCR and a split point or linking point.

Figure 19.8(a), (b), (c) and (d) show the adjacency graphs of SCRs built for the single closed region graphs in Figure 19.6(a), 19.6(b), 19.6(c) and 19.6(e) respectively.

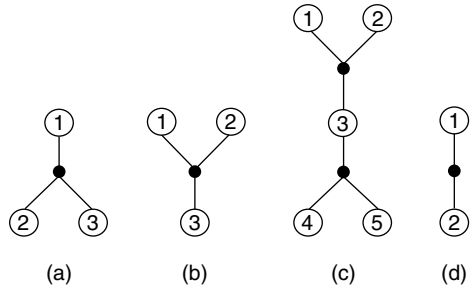



**Figure 19.8** The adjacency graphs of SCRs for different connections.

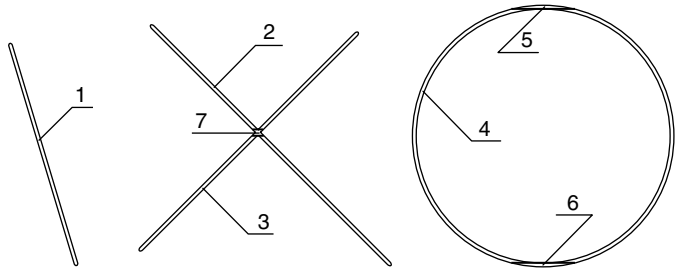



**Figure 19.9** Seven types of SCR.

The single closed regions constructed can also be categorized into seven types according to their adjacency relationships with other single closed regions, as shown in Figure 19.9:

1. A singular SCR – has no predecessor and no successor.
2. A beginning SCR – has no predecessor and only one successor.
3. An end SCR – has only one predecessor and no successor.
4. A regular SCR – has only one predecessor and only one successor.
5. A branching SCR – has one predecessor at most and more than one successor.
6. A merging SCR – has more than one predecessor and one successor at most.
7. A cross SCR – has more than one predecessor and successor.

Different properties of SCRs are stored in the node data file and include their types (one of the seven types above), the categories of segment represented by the SCR (i.e. point, arrowhead, straight line or arc) and characteristics. For a point, its characteristics are the coordinates of its center. For an arrowhead, its characteristics cover the coordinates of its top point, its angle to the horizontal ($x$) axis, length and width. For a straight line, its characteristics are its line width, starting coordinates, end coordinates, length, slope and intercept. For an arc, its characters cover its line width, the coordinates of its curvature center, its curvature radius, the starting angle to the $x$-axis (let the direction of the arc be counter-clockwise.) and the end angle to the $x$-axis. The coordinates of each linking point and the labels of SCRs adjacent to the linking point are stored in the black node data file as the properties of the linking point. It is obvious that the adjacency graphs of SCRs represent all the single closed regions, their linking points and their adjacency relationships.

## *3.4 Constructing the Networks of SCRs*

In the adjacency graphs of SCRs, there are some black nodes, which represent the points linking the SCRs of the elements of primitives or annotations, not the SCRs of the elements (i.e. straight lines or arcs) themselves. It is the linking points that contain useful information about the relationships among elements, such as their tangential relationship or intersection relationship at a certain angle. These linking points should be further analyzed to extract different types of relationship among elements for constructing the networks of SCRs.

Whether the elements adjacent to a linking point are intersected or tangential to each other can be determined by calculating the angles among them. Accordingly, the networks of SCRs can be further obtained by using the following procedure:

1. Delete each black node and all the lines between the black node and its related nodes; instead, connect, with a line, each pair of all the nodes having an adjacency relationship with the black node.

2. Eliminate the following nodes and the lines between the node and its related nodes:

   — the node representing a branching SCR if the extensions of its successors pass the center point of the branching SCR;
   — the node representing a merging SCR if the extensions of its predecessors pass the center point of the merging SCR;
   — the node representing a cross SCR if the extensions of its predecessors and successors pass the center point of the cross SCR. For this case, every two of the nodes for all predecessors and successors will be connected with a line if there is no line between them.

3. Record the properties of the relationship for each pair of related nodes:

   — If they represent two straight lines, compare their slopes to determine the type of their relationship. When their slopes are very close or equal to each other, they are collinear. Then, the type of their relationship (i.e. collinear) and the coordinates of their linking point are recorded as the properties of the line between the two nodes in the line data file. If not, they intersect each other, and the angle between them is calculated. The type of their relationship (i.e. intersection), the degree of angle between them, and the coordinates of their intersecting point are thus recorded as the properties of the line between the two nodes in the line data file.
   — If they represent two arcs, compare their curvature radii and curvature center coordinates to determine the type of their relationship. When they have the same or very close curvature radius and curvature center coordinates, they are concyclic. Then, the type of their relationship (i.e. concyclic) and the coordinates of their linking point are recorded as the properties of the line between the two nodes in the line data file. If not, the distance between the two curvature centers is calculated. When the distance is equal to the sum of their curvature radii, they are externally tangential to each other. If the distance is equal to the difference of their curvature radii, they are internally tangential to each other. Otherwise, they intersect each other. The type of their relationship and the coordinates of their tangential point or intersecting point are thus recorded as the properties of the line between the two nodes in the line data file.
   — If one represents a straight line and another an arc, calculate the distance between the straight line and the curvature center of the arc to determine the type of their relationship. When the distance is equal to the curvature radius of the arc, the straight line is tangential to the arc. If not, they intersect each other. Then, the type of their relationship and the coordinates of their tangential point or intersecting point are recorded as the properties of the line between the two nodes in the line data file.

4. Find out the node representing an SCR of the straight line, the slope of which is very close to or equal to zero (i.e. a horizontal straight line), and check whether the intersecting point between it and one of its predecessors is very close to or at the intersecting point between it and one of its successors. If yes, apply a line to connect the related predecessor and successor.

Figure 19.10(a), (b), (c) and (d) show the networks of SCRs generated for the adjacency graphs of SCRs in Figure 19.8(a), (b), (c) and (d) respectively.

For a view, like the view shown in Figure 19.11(a), its SCR graph, adjacency graphs of SCRs, and networks of SCRs can in turn be obtained using this method, as shown in Figure 19.11(b), (c) and (d), respectively. In its networks of SCRs, a node represents the properties of an SCR for a related element (a straight line or an arc) and a line between two nodes represents the properties of the relationship between the two elements. Therefore, the networks of SCRs satisfy the first requirement of a compact representation of the raster image as a bridge from the raster image to understanding, mentioned in the second section of this chapter. That is, the networks of SCRs distinguish different types of linking

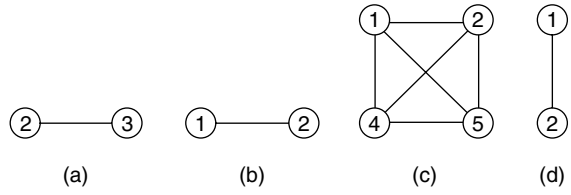



**Figure 19.10** The networks of SCRs for different connections.

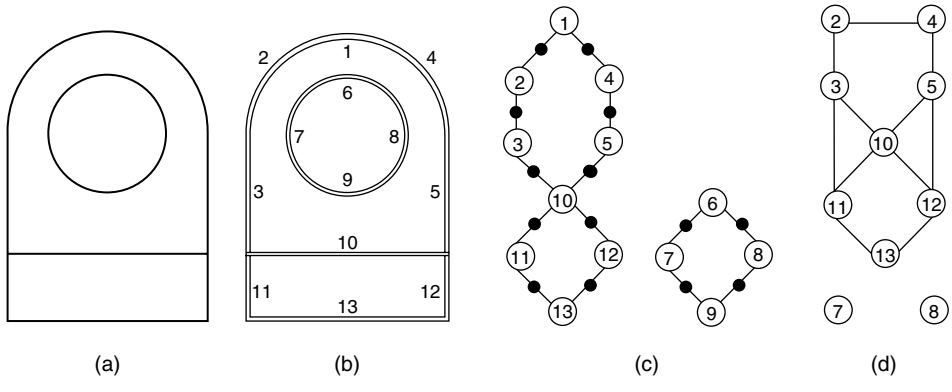



**Figure 19.11** The SCR graph, adjacency graphs of SCRs, and networks of SCRs for a view.

point for different relationships of the related elements to provide necessary information for extracting lexical, syntactic and semantic information in the subsequent phases.

## 4. A Bridge from the Raster Image to Understanding and 3D Reconstruction

The networks of SCRs are holo-extractions of information from a raster image, and can be used for subsequent different level understanding and 3D reconstruction. The main elements and workflow of further understanding and 3D reconstruction are shown in Figure 19.12, and illustrated in more detail as follows.

### *4.1 Separating the Annotations and the Outlines of Projections of Parts*

In an engineering drawing, there are outlines of orthogonal projections of parts and annotations. They should be separated first so that the former can be used for reconstructing a 3D model of a part, and the latter can be recognized as providing necessary information for parameters, features, etc. The former consists of straight lines, arcs and/or points. The latter includes non-outlines (e.g. center lines), symbols (e.g. machining and surface texture symbols) and text. The character of text may be Arabic numbers, English characters, Chinese characters or other languages' characters. They all consist of straight lines, arcs and/or points as well. These straight lines or arcs may be split up into segments (i.e. shorter straight lines or arcs), by branching, merging, cross, tangential or split points, which have been represented

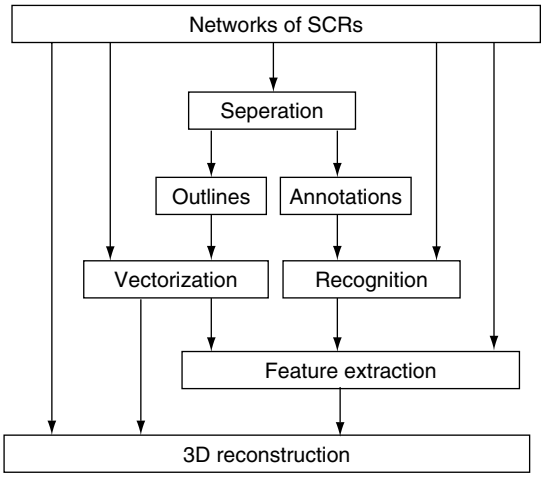



**Figure 19.12** The main elements and workflow of further understanding for 3D reconstruction.


**Figure 19.13** The SCR graphs for text characters.

by their corresponding SCRs. Figure 19.13(b) shows three graphs of SCRs generated for the English character, Arabic number and Chinese character in Figure 19.13(a), respectively. The networks of SCRs for each character or symbol can also be constructed using the same method for the outlines of projections of parts introduced above. The networks of SCRs thus provide a unified base for further recognizing the primitives and the annotations.

Normally, the extraction of text is implemented based on the areas of the region containing a set of connected segments and the densities of black pixels in the region, since the area for the character of text is smaller and has a higher density of black pixels than the primitives. If the character string is not arranged horizontally, it should be identified and rotated to the horizontal position, and then recognized using further methods. The extraction of outlines of projections of parts is done according to their line widths, since their line width is thicker than others. The separation between non-outlines and symbols is carried out using pattern recognition. Since one network of SCRs represents a set of connected segments, and the networks of SCRs contain all the necessary information, including line widths and those necessary for calculating their areas, densities, sloping angles to the $x$-axis and pattern, these extractions or separations can be easily implemented and further recognizing methods can then be applied, all based on their networks of SCRs. Therefore, the networks of SCRs meet the second requirement of a compact representation of the raster image as a bridge from the raster image to understanding, as mentioned in the second section of this chapter. That is, the networks of SCRs provide a unified base for recognizing both the outlines of orthogonal projections of parts and the annotations, and facilitate their separation.

## *4.2 Vectorization*

### 4.2.1 Recognition of Horizontal Thick Lines and Arrowheads

Since the line widths can be calculated and have been stored in the node data file, thick lines or thin lines can be detected by comparing the line widths of all the nodes in the networks of SCRs for a drawing. An arrowhead can also be extracted according to the change of width of its SCR. But it is difficult to identify a horizontal thick line or a horizontal arrowhead connected with a horizontal thin line, since an SCR for a thin line will contain a part of the SCR for a thick line or an arrowhead connected with the thin line, as shown in Figures 19.14(a) and 19.14(b) respectively.

In these cases, the SCR for a thin line will be divided into several parts according to its adjacency relationships with other related SCRs, as shown in Figures 19.14(c) and 19.14(d) respectively. That part of the SCR for a thin line, which has adjacency relationships with other related SCRs, will then be aggregated into a horizontal thick line or a horizontal arrowhead, as shown in Figures 19.14(e) and 19.14(f) respectively. Since all the information needed for the operations is contained in their networks of SCRs, the extraction of horizontal thick lines or arrowheads connected with horizontal thin lines can be implemented conveniently based on their networks of SCRs. Therefore, the networks of SCRs also satisfy the third requirement of a compact representation of the raster image as a bridge from the raster image to understanding, as mentioned in the second section of this chapter. That is, the networks of SCRs facilitate recognition of line patterns and arrowheads.

### 4.2.2 Recognition of Complete Arcs and Circles

In a 2D drawing, there may be many arcs or circles, which are divided into several segments by other straight lines, arcs or circles. These segments have their own nodes in the networks of SCRs. A circle may also be divided into two SCRs for two halves of a circle, which have been represented in the network of SCRs. In order to extract the vectors (i.e. the complete arcs or circles), the SCRs for these segments should be combined together. The method is to search for the first SCR representing the arc in the networks of SCRs as a seed SCR; then to find the SCRs that are adjacent to the seed SCR and belong to the same arc, via the networks of SCRs; and finally, to combine the related SCRs to get a complete arc or circle. If there is no further related SCR for an arc, the second seed SCR for the arc will be found in the networks, and the above process will be repeated until no seed SCR for an arc can be found. Figure 19.11(a) can be used as an example for recognizing a complete arc and a circle. As shown in Figure 19.11(b), the arcs represented by SCRs 2 and 4 belong to the same arc, and are

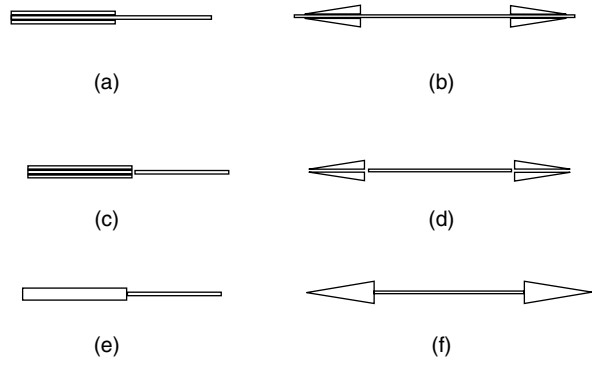



**Figure 19.14** Identification of horizontal thick lines and arrowheads.

connected by a point represented by SCR 1. The arcs represented by SCRs 7 and 8 belong to the same circle, and are connected by linking points represented by SCRs 6 and 9. While searching for the SCR for an arc in the networks of SCRs for the view, the SCR for arc 2 will be found as a seed SCR first, and then the SCR for arc 4 will be found adjacent to it and belonging to the same arc via the network of SCRs and will be combined with the seed SCR to obtain an extended arc. Since there is no other related arc, the complete arc is extracted. Then the SCR for arc 7 will be found in the networks of SCRs as the second seed arc, and the same process will be repeated to connect the SCR for arc 8 with it into a complete circle. Since there are no further SCRs for arcs left, the process will be stopped and another process for recognizing complete straight lines will be started.

### 4.2.3 Recognition of Complete Straight Lines

Complete straight lines can also be obtained using the same method. Figure 19.11(b) can still be used as an example of recognizing complete straight lines. The straight lines represented by SCRs 3 and 11 belong to the same straight line, and the straight lines represented by SCRs 5 and 12 belong to another straight line. While searching for the SCR for a straight line in the networks of SCRs for this view, SCR 3 will be found as a seed SCR first, then SCR 11 will be found adjacent to it and belonging to the same line via the network of SCRs, since the type of their relationship is stored in the line data file, and SCRs 3 and 11 will be combined together as an extended straight line. Since there is no SCR for straight lines found belonging to the same straight line, the complete straight line is extracted. Then, SCR 5 will be found as the second seed SCR, from which SCR 12 will be found adjacent to it and belonging to the same straight line via its network of SCRs and combined with the seed line to form an extended straight line. Since there are no other SCRs for straight lines, which are collinear, the vectorization process will be stopped.

### 4.2.4 Software Prototype

Based on the method introduced above, a software prototype has been made in C++. Many paper drawings have been vectorized successfully by using this software prototype. As an example, Figure 19.15(c) shows the results of running the software prototype for a view of a drawing shown in Figure 19.15(a). Figure 19.15(b) shows its SCR graph for illustration. It is obvious that the vectorization method based on the networks of SCRs is feasible, and that the networks of SCRs meet the fourth requirement of a compact representation of the raster image as a bridge from the raster image to understanding, as mentioned in the second section of this chapter. That is, the networks of SCRs facilitate recognition of vectors quickly and precisely.

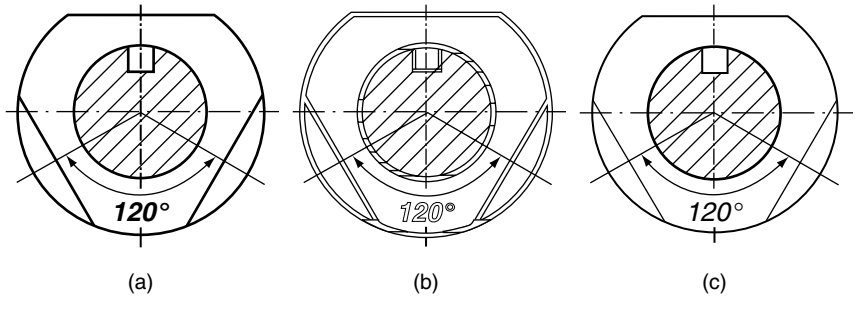



**Figure 19.15** The results of vectorization using the software prototype based on the networks of SCRs.

## *4.3 3D Reconstruction*

For 3D reconstruction, two families of methods have been developed. The first family is known as the 'fleshing out projections' concept [18,19]. Another family is volume-oriented instead of privileging wire frames [20].

The former is based on the reconstruction of a wire-frame model by matching vertices and edges; the real faces of the object are then found by propagating constraints on this wire frame. The idea is to search for the circuits on each view by sorting the edges in trigonometric order with respect to the normal to the surface, at each vertex, and by traversing the resulting graph of oriented edges; to create a face for each finite circuit and a hole for each infinite circuit; and to make up a real solid with a group of faces without violating the Moebius rule [21,22]. The Moebius rule says that when a real edge is on the border of two real faces, the traversals along the circuits of these two faces follow the edge in opposite directions. Figure 19.16(b) [21] shows the circuits for the view in Figure 19.16(a). Therefore, circuits are crucial for this method of 3D reconstruction and, sometimes, a vector-based representation yielded is not very useful [21].

The latter method decomposes each view into several predefined types of subview [23] or exterior boundary and interior boundary [24], generates their 3D subparts and then combines them to build the complete object [20]. Each view and its predefined types of subview or exterior boundary and interior boundary are all circuits. Therefore, circuits are also crucial for this method and, sometimes, a vector-based representation yielded is not very useful either.

Since a circuit consists of elements, represented by SCRs, and the networks of SCRs provide holo-relationships of all the elements, the circuits can be searched easily via the networks of SCRs. Moreover, 3D reconstruction also needs various features, which are also extracted from the networks of SCRs, as indicated in Figure 19.12. It is obvious that the networks of SCRs are more useful and can be used as a base for the subsequent 3D reconstruction. Therefore, the networks of SCRs meet the fifth requirement of a compact representation of the raster image as a bridge from the raster image to understanding, as mentioned in the second section of this chapter.

Based on the networks of SCRs, a 3D reconstruction method has been developed [25]. As an example, Figure 19.17 shows the results of running the software prototype for a drawing shown in Figure 19.18, where each SCR is represented by its corresponding symbol (e.g. $F_4$ is the symbol for the bottom SCR in front view). Figure 19.19 shows its networks of SCRs generated according to the drawing, where V indicates an intersecting point by two SCRs and C by more than two SCRs.

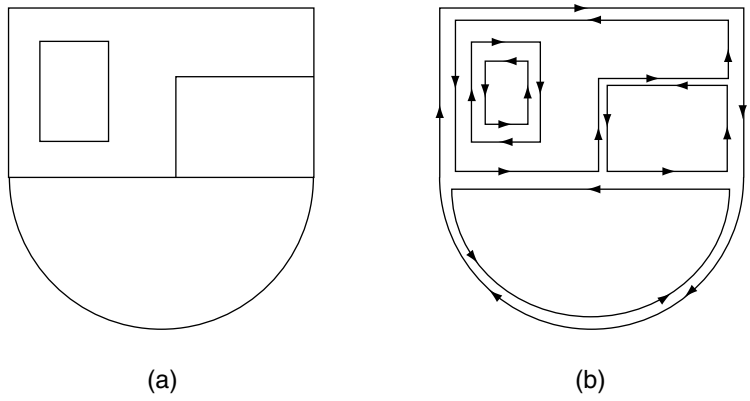



**Figure 19.16** A search for circuits to create faces of 3D models.

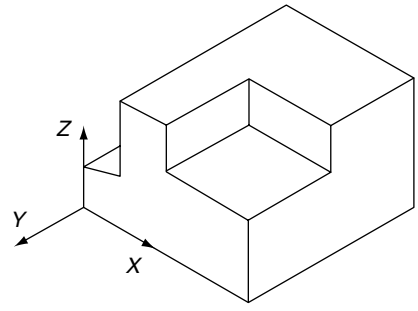



**Figure 19.17** An example of 3D reconstruction.

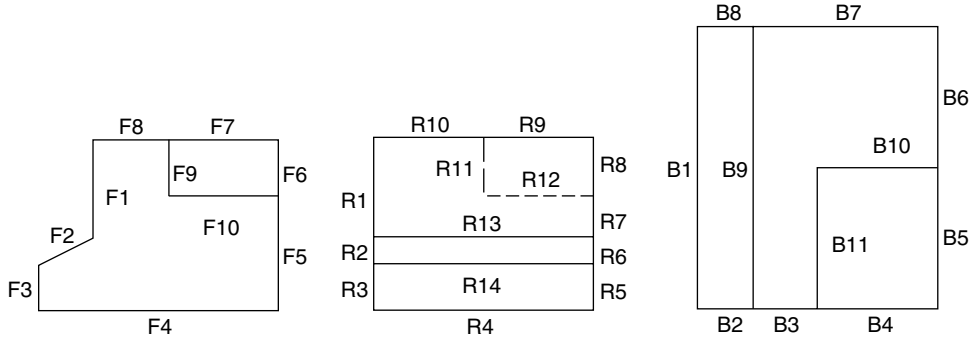



**Figure 19.18** A 2D drawing of an object.

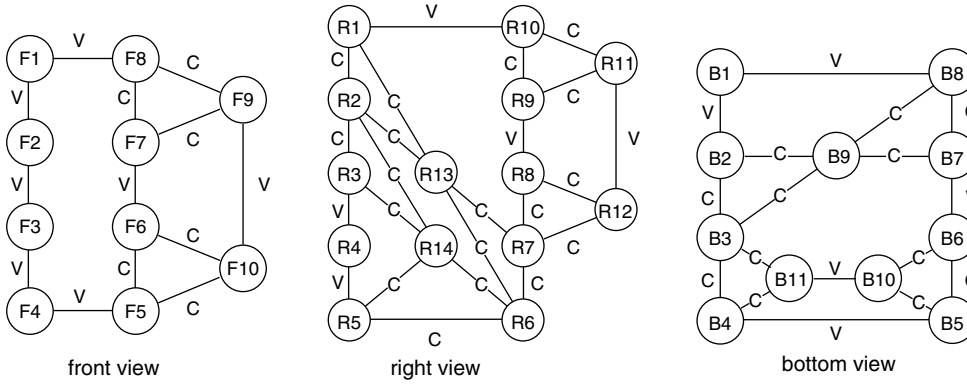



**Figure 19.19** The networks of SCRs generated from the 2D drawing.

Figure 19.20 shows circuits obtained via the networks of SCRs, which are indicated by closed chain lines and/or symbols. For example, $L_{F2}$ is the symbol for the circuit represented by a closed chain line in front view, and $L_{F1}$ is the symbol for the circuit covering $L_{F2}$ and $L_{F3}$, i.e. the outside outline of the front view.

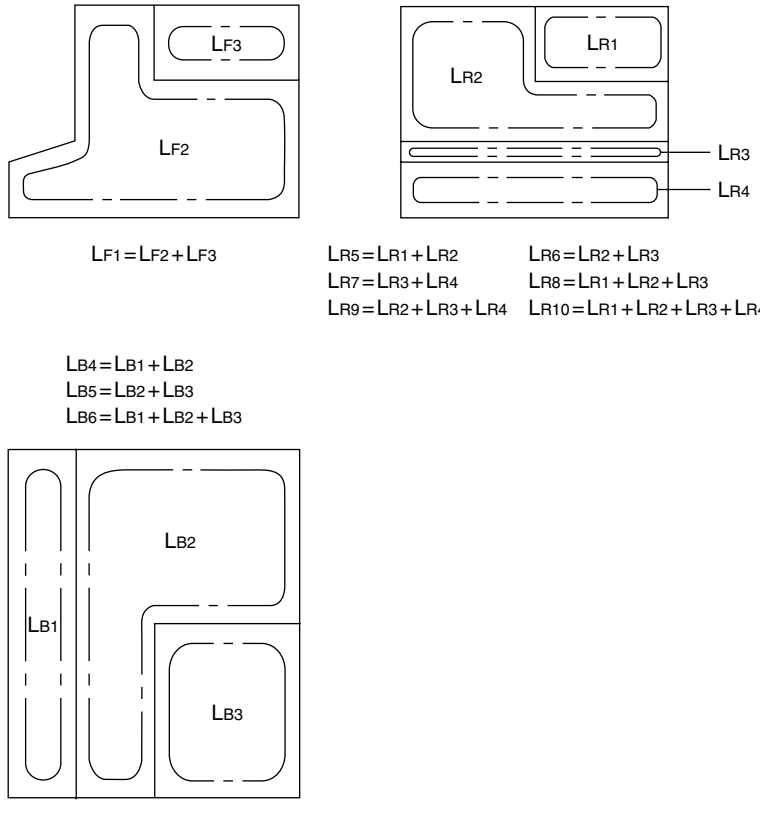



**Figure 19.20** The circuits obtained via the networks of SCRs.

## 5. Classification of Digital Curves

According to the holo-extraction method developed, the possible results obtained may be the closed region for a straight line, a circular arc, an elliptical arc, a spline curve, a combined line, a point or an arrowhead, as shown in Figure 19.21(a), (b), (c), (d), (e), (f) and 19.21(g), respectively. A point may be an independent point or intersecting point, as shown in Figure 19.21(f) and Figure 19.4(c) respectively, and can be recognized according to the length of its closed region and the width change of its closed region. An arrowhead can also be extracted according to the width change of its closed region. Therefore, the rest of them are digital curves (i.e. straight lines, circular arcs, elliptical arcs, spline curves or combined lines), which need to be recognized.

However, ellipses and spline curves in paper drawings can be considered as combined lines. Thus, the first task for recognition of digital curves is to classify all the digital contours into three types: straight-line segments, circular arcs and combined lines. The procedure of classification is as follows.

### *5.1 Extracting the Representative Points of Digital Curves*

For each closed region, the midpoint of each run in it can be recorded as the points that form a digital curve. In order to reduce computing load, some equal-spaced points among these points are selected

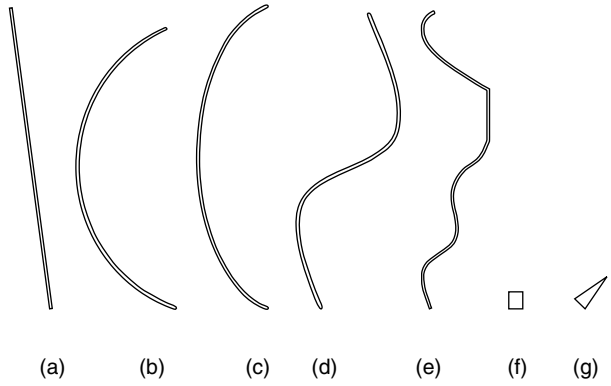



**Figure 19.21** Types of closed region.

as the representing points for curve fitting. The number ($N$) of points is determined according to the length of segment using:

$$N = \begin{cases} \left[\sqrt{d(n-1)}\right], & \text{if } d(n-1) > 10 \text{ mm} \\ 10, & \text{if } d(n-1) \leq 10 \text{ mm} \end{cases} \tag{19.2}$$

where $d$ is the distance between two adjacent points, $n$ is the total number of points in the segment, $d(n-1)$ denotes the length of segment and $\left[\sqrt{d(n-1)}\right]$ is the integral part of $\sqrt{d(n-1)}$. The longer the segment, the more equal-spaced points will be selected for curve fitting. But the number of selected points is not less than ten to ensure effective curve fitting. These points can be written as:

$$P = \{P_i = (x_i, y_i); i = 1, \ldots, N\} \tag{19.3}$$

## 5.2 Fitting a Straight line to the Set of Points

The straight line that best fits a given set of points can be determined using the least squares criterion that minimizes the fitting error. According to least squares regression [26], the $N$ points can be fitted by a straight line:

$$y = kx + c \tag{19.4}$$

where $k$ and $c$ denote the slope and intercept respectively, and can be estimated by the following formulas:

$$k = \frac{N\sum_{i=1}^{N} x_i y_i - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i}{N\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2} \tag{19.5}$$

$$c = \frac{\sum_{i=1}^{N} x_i^2 \sum_{i=1}^{N} y_i - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} x_i y_i}{N\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2} \tag{19.6}$$

Thus, the average distance ($d_s$) from these points ($x_i, y_i$) to the fitted straight line can be calculated using the following formula:

$$d_s = \frac{\sum_{i=1}^{N} |(kx_i + c) - y_i|}{N\sqrt{k^2+1}} \tag{19.7}$$

## *5.3 Fitting a Circular Arc to the Set of Points*

On the other hand, according to least squares regression, these points can also be fitted by a circular arc:

$$(x-a)^2 + (y-b)^2 = R^2 \tag{19.8}$$

where $(a, b)$ and $R$ denote the coordinates of the circle's center and its radius respectively, and can be estimated using the following formulas [27]:

$$a = \frac{b_1 a_{22} - b_2 a_{12}}{\Delta} \tag{19.9}$$

$$b = \frac{b_2 a_{11} - b_1 a_{21}}{\Delta} \tag{19.10}$$

$$R = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} x_i^2 - 2\sum_{i=1}^{N} x_i a + Na^2 + \sum_{i=1}^{N} y_i^2 - 2\sum_{i=1}^{N} y_i b + Nb^2\right)} \tag{19.11}$$

where

$$a_{11} = 2\left[\left(\sum_{i=1}^{N} x_i\right)^2 - N\sum_{i=1}^{N} x_i^2\right] \tag{19.12}$$

$$a_{12} = a_{21} = 2\left(\sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i - N\sum_{i=1}^{N} x_i y_i\right) \tag{19.13}$$

$$a_{22} = 2\left[\left(\sum_{i=1}^{N} y_i\right)^2 - N\sum y_i^2\right] \tag{19.14}$$

$$b_1 = \sum_{i=1}^{N} x_i^2 \sum_{i=1}^{N} x_i - N\sum_{i=1}^{N} x_i^3 + \sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i^2 - N\sum_{i=1}^{N} x_i y_i^2 \tag{19.15}$$

$$b_2 = \sum_{i=1}^{N} x_i^2 \sum_{i=1}^{N} y_i - N\sum_{i=1}^{N} y_i^3 + \sum_{i=1}^{N} y_i \sum_{i=1}^{N} y_i^2 - N\sum_{i=1}^{N} x_i^2 y_i \tag{19.16}$$

$$\Delta = a_{11}a_{22} - a_{21}a_{12} \tag{19.17}$$

Thus, the average distance ($d_c$) from these points ($x_i, y_i$) to the fitted circular arc can be calculated using the following formula:

$$d_c = \frac{\sum_{i=1}^{N} \left|\sqrt{(x_i-a)^2 + (y_i-b)^2} - R\right|}{N} \tag{19.18}$$

### 5.4 Determining the Type

If the average distance ($d_s$) from the points in a digital curve to a fitted straight line is equal to zero, the digital curve is exactly a straight line according to the least squares criterion. If the average distance ($d_c$) from the points in a digital curve to a fitted circular arc is equal to zero, the digital curve is exactly a circular arc according to the least squares criterion. But the digital curves are scanned from paper drawings that are not accurate. Therefore, an error range (double the width of the closed black pixel region) is given. Thus, only if $d_s$ is smaller than both $d_c$ and double the width of the closed black pixel region, is this digital curve a straight line (rule 1). Only if $d_c$ is smaller than both $d_s$ and double the width of the closed black pixel region, is this digital curve a circular arc (rule 2). If both $d_s$ and $d_c$ are greater than double the width of the closed black pixel region, this digital curve is neither a straight line nor a circular arc, and should be the third possibility, i.e. a combined line (rule 3). Therefore, after the two average distances ($d_s$ and $d_c$) of a digital curve are obtained, the type of the digital curve can be determined by using these three rules.

After classification, the more precise regression equation can be obtained to fit to all the midpoints of runs in the closed region by using the above formulas if the digital curve is determined as a straight line or a circular arc. When it is a combined line, it will then be decomposed into straight-line segments and/or circular arcs.

## 6. Decomposition of Combined Lines Using Genetic Algorithms

The method for decomposition is to find the optimal split (break) points with the least number of segments, each of which can be fitted to a straight line or a circular arc with least error. This is an optimization problem, which can be solved by using genetic algorithms [17].

The block diagram of the algorithm is shown in Figure 19.22. This algorithm includes an encoding scheme, a fitness function, genetic operators (crossover, mutation and selection) and control parameters; it will be illustrated in more detail in the following sections.

### 6.1 Initial Population

In genetic algorithms, a possible solution (a possible combined line for this case) should be encoded as a chromosome with length $n$ (the number of black pixels), which is a binary gene string as follows:

$$C = \{g_p \,|\, p = 1, 2, \ldots, n\} \tag{19.19}$$

where $g_p = 1$ if point $p$ is a break point; otherwise $g_p = 0$. An initial population with a set of chromosomes is generated randomly [17], entered into a mating pool for subsequent operations, and defined as:

$$\{C_i \,|\, i = 1, 2, \ldots, m\} \tag{19.20}$$

where $m$ is the number of chromosomes in a population and is equal to $n/2$ in this case. In digital curves, two break points should not be adjacent or very close to each other. To avoid this, each chromosome will be modified using the following rules:

- if the distance between two adjacent 1 genes is less than 15, change the second gene from 1 to 0;
- if there are several 1 genes, which are continuous, change the genes, except the first and the last gene, from 1 to 0.

For example, Figure 19.23(a) shows a combined line, which consists of an elliptical arc (AB), two straight-line segments (BC and CD), two circular arcs (DE and EF) and a spline curve (FG), and has 384 points in total. An initial population with 172 chromosomes is first generated randomly. One of the 172 chromosomes has 17 genes labeled 1, as shown in Figure 19.24. It means that there are 17 split (break) points and 18 segments in the combined line represented by this chromosome.

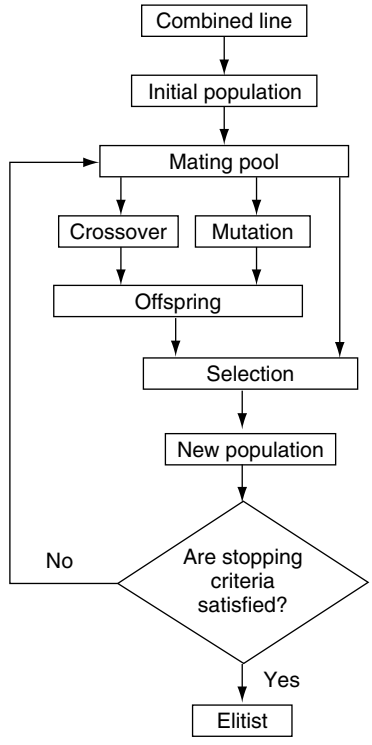



**Figure 19.22** The block diagram of the algorithm.

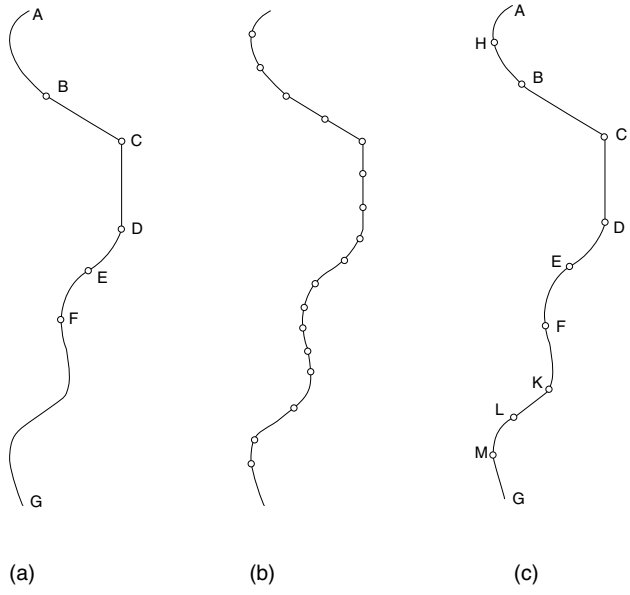



**Figure 19.23** A combined line.

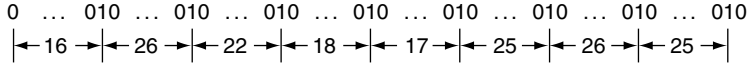



**Figure 19.24** A chromosome for a combined line.

## 6.2 Fitness Function

Each segment in a combined line represented by a chromosome can be fitted by a straight line or a circular arc. Their approximation errors ($d_s$ and $d_c$) can be calculated using Equation (19.7) and Equation (19.18) respectively. The smaller error will be recorded as the approximation error of the segment. Thus, the approximation error ($\mathrm{Err}(C_i)$) for the whole combined line is the sum of errors of segments in the combined line. The fitness function is used for evaluating how well each chromosome approximates the original shape, and is defined for the $i$th chromosome $C_i$ as [28]:

$$\mathrm{Fit}(C_i) = \frac{1}{\mathrm{Err}(C_i) \times S^k} \tag{19.21}$$

where $S$ is the total number of segments in $C_i$ and $k$ is a parameter which can adjust the effect the number of segments will have, so that it is possible to obtain the least segments with the least fitting error. The larger $k$ is, the more effect the number of segments will have. For this case, $k$ is selected to be 0.5. For the chromosome in the initial population of the combined line shown in Figure 19.24, its fitness function value is equal to 0.0385.

## 6.3 Crossover

To create the next generation, new chromosomes, called offspring, are formed first using a crossover operator. In this algorithm, crossover proceeds in three steps. First, $m/2$ pairs of chromosomes are randomly picked up from the mating pool. Secondly, the distance between the two chromosomes in each pair is calculated using the following formula:

$$D_{i,j} = \sum_{l=1}^{N} \left| g_i(l) - g_j(l) \right| \tag{19.22}$$

Since the crossover between two similar chromosomes is not useful for efficient evolution, the probability of crossover should be reduced if their distance is smaller, and increased if their distance is larger. The probability of crossover for each pair is adjusted according to the following formula [28]:

$$P_c(i,j) = \frac{1}{1 + e^{-(D_{i,j} - \alpha)/\beta}} \tag{19.23}$$

where $\alpha$ and $\beta$ are the parameters which make $P_c(i,j)$ change with $D_{i,j}$ in a certain way. The pairs of chromosomes for crossover are then selected by a roulette selection scheme based on their probabilities calculated. Thirdly, a pair of crossover points is selected randomly for each selected pair of chromosomes, and the segments bounded by the crossover points are exchanged.

## 6.4 Mutation

New chromosomes or offspring can also be formed using a mutation operator, which involves modification of the values of genes in a chromosome and increases the variability of a population. In the case when fitness functions of chromosomes in a population converge to a small range or local optimum, it is difficult for crossover to generate offspring with more improved fitness function values, but mutation can play an important role here. In order to find a global optimum rapidly, the probability

of mutation should be increased when the fitness functions of a population converge to a small range. The probability of mutation can be adjusted using the formula [29]:

$$P_m = \begin{cases} t \times \dfrac{\text{Fit}_{\max} - \text{Fit}(C_i)}{\text{Fit}_{\max} - \overline{\text{Fit}}}, & \text{if } \text{Fit}(C_i) \geq \overline{\text{Fit}} \\ t, & \text{if } \text{Fit}(C_i) < \overline{\text{Fit}} \end{cases} \tag{19.24}$$

where $t = 0.35$, $\text{Fit}_{\max}$ is the maximum fitness function value in the population, and $\overline{\text{Fit}}$ is the average fitness function value of the population. In the mutation process, a random number in [0,1] for every gene position within the chromosome string is generated and checked. If the random number is smaller than the probability of mutation, the bit value of the gene is altered, otherwise, it is kept unchanged.

## 6.5 Selection

After crossover and mutation operations, an elitist selection scheme is used. The chromosome with the highest fitness function value among the old population and its offspring is selected as an elitist and copied directly into the new population of the next generation. With this operation, nature's survival-of-the- fittest mechanism can be guaranteed. The other chromosomes are selected by a roulette selection scheme. A roulette wheel, on which each chromosome in the old population and its offspring is represented by a slot with slot size proportional to its fitness function values, is utilized. The probability that a chromosome $C_i$ is selected as a member of the next generation, is:

$$P(C_i) = \frac{\text{Fit}(C_i)}{\sum_{j=1}^{M} \text{Fit}(C_j)} \tag{19.25}$$

where $M = (m + \lambda)$ and $\lambda$ is the number of offspring. The $m$ selected chromosomes are then entered into a mating pool as the next generation. The first iteration of the genetic algorithm is thus completed.

## 6.6 Convergence and Control Parameters

After the second generation is generated, the above crossover, mutation and selection operations will be repeated until the best chromosome is obtained. Since the digital curve varies and the improvement of fitness is not always continuous, it is not appropriate to set up the number of iterations or error threshold. In this case, a threshold for the number of generations that have the same best chromosomes is used. That is, if $n_i - n^*$ is greater than a threshold $q = 20$, the iteration process can be stopped, where $n_i$ is the current generation number and $n^*$ denotes the generation number when the best chromosome, among all generations, is first found. Then, the best chromosome can be output as the optimal solution. For the example of the combined line in Figure 19.23, the best combined line represented by the best chromosome was obtained as shown in Figure 19.23(c). It can be seen that the elliptical arc (AB) consists of two circular arcs (AH and HB), and that the spline curve (FG) comprises two circular arcs (FK and LM) and two short straight-line segments (KL and MG).

## 6.7 Determination of the Relationships Between the Segments

In the best chromosome, the point corresponding to the gene labeled 1 is a break point. The segment bounded by two break points has been fitted by a straight line or a circular arc with the least error. The relationship between two adjacent segments can be determined using the following rules:

- If they are both straight lines, they intersect each other.
- If they are both circular arcs, the distance between their centers will be calculated. When the distance is equal to the sum of their radii, the two arcs are externally tangential to each other. If the distance

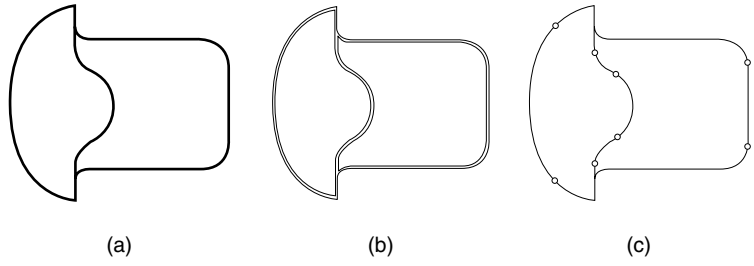



**Figure 19.25** An example of digital curve recognition.

is equal to the difference of their radii, the two circular arcs are internally tangential to each other. Otherwise, they intersect each other.

- If one segment is a straight line and another a circular arc, the distance from the center of the circular arc to the straight line will be calculated. If the distance is equal to the radius of the circular arc, the straight line is tangential to the circular arc. Otherwise, they intersect each other.

With the types of segment and their relationships determined, the combined line can thus be reconstructed.

### *6.8 Software Prototype*

Based on the method introduced above, a software prototype has been made in C++. Many digital curves have been recognized successfully by using this software prototype. As an example, Figure 19.25(c) shows the results of running the software prototype for a view of a drawing shown in Figure 19.25(a). Figure 19.25(b) shows its closed region graph for illustration. It is obvious that the recognition method based on the genetic algorithm is feasible.

## 7. Conclusions

Research on the conversion from 2D paper drawings to 3D computer feature models has been stuck in low-level coding [1]. One of the reasons for this is that the three phases of understanding process have been isolated, and people have been doing their research on only one of the phases since the whole conversion is complicated and more difficult. For instance, the vectorization methods for the first phase were developed only for getting straight lines, arcs, circles, etc., so that much information contained in the drawing was lost after the vectorization. In fact, the understanding process for its subsequent 3D reconstruction is an iterative process for searching different level relationships and performing corresponding connections. In order to facilitate the iterative processes, a holo-extraction of information from the raster image is needed, and a compact representation of the raster image should be generated as a bridge from the raster image to understanding. Such a holo-extraction method of information from paper drawings has been developed [30] by constructing the networks of single closed regions of black pixels. The networks of SCRs are different from the vectors recognized by vectorization. A network of SCRs represents a set of connected segments or elements and contains not only the information about the related segments themselves but also the types of linking point among these segments for different relationships of the related segments (e.g. tangential point, intersecting point and merging junction) to provide necessary information for extracting lexical, syntactic and semantic information in the subsequent phases. Therefore, the networks of SCRs are the bridges from the raster image to understanding. The recognition method [31] of digital curves introduced in

this chapter is developed based on the new holo-extraction methods and used for subsequent pattern recognition and 3D reconstruction. This method first constructs the networks of single closed regions of black pixels with all the information about both segments and their linking points, in order to classify all the digital contours or segments represented by SCRs into three types: straight-line segments, circular arcs and combined lines, and then decomposes the combined lines into least basic sublines or segments (straight-line segments or circular arcs) with least errors using genetic algorithms and determines their relationships (intersecting or tangential to each other). In the algorithm, the operation has been improved by using sparse points, adaptive probabilities of crossover, adaptive probabilities of mutation and selection based on an enlarged sampling space (including the old population and offspring). It is easy to implement evolution based on an enlarged sampling space [17]. Based on the method developed, a software prototype has been made in C++ [30,31]. Many digital curves have been recognized successfully by using this software prototype. It has been verified that the recognition method based on the networks of SCRs and the genetic algorithm is feasible and efficient. This method and its software prototype can be used as a base for further work on subsequent engineering drawing understanding and 3D reconstruction.

## References


[1] Dori, D. and Tombre, K. "From engineering drawings to 3D CAD models: are we ready now?" *Computer-Aided Design*, **27**(4), pp. 243–254, 1995.

[2] Hough, P. V. C. *A method and means for recognizing complex patterns*, USA Patent 3,096,654, 1962.

[3] Liu, W. and Dori, D. "From raster to vectors: extracting visual information from line drawings," *Pattern Analysis and Applications*, **2**(2), pp. 10–21, 1999.

[4] Lam, L., Lee, S. W. and Suen, C.Y. "Thinning methodologies – A comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(9), pp. 869–887, 1992.

[5] Naccache, N. J. and Shinghal, R. "SPTA: a proposed algorithm for thinning binary patterns," *IEEE Transactions on Systems, Man, and Cybernetics*, **14**, pp. 409–418, 1984.

[6] Peleg, S. and Rosenfeld, A. "A min-max medial axis transformation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **3**, pp. 208–210, 1981.

[7] Davies, E. R. and Plummer, A. P. N. "Thinning algorithms: a critique and a new methodology," *Pattern Recognition*, **14**, pp. 53–63, 1981.

[8] Han, C. C. and Fan, K. C. "Skeleton generation of engineering drawings via contour matching," *Pattern Recognition*, **27**(2), pp. 261–275, 1994.

[9] Jimenez, J. and Navalon, J. L. "Some experiments in image vectorization," *IBM Journal of Research and Development*, **26**, pp. 724–734, 1982.

[10] Shapiro, B., Pisa, J. and Sklansky, J. "Skeleton generation from $x-y$ boundary sequences," *Computer Graphics and Image Processing*, **15**, pp. 136–153, 1981.

[11] Liu, W. and Dori, D. "Sparse Pixel Tracking: A first vectorization algorithm applied to engineering drawings," *Proceedings of the 13th International Conference on Pattern Recognition*, Vienna, Austria, **III** (Robotics and Applications), pp. 808–811, 1996.

[12] Lin, X., Shimotsuji, S., Minoh, M. and Sakai, T. "Efficient diagram understanding with characteristic pattern detection," *Computer Vision, Graphics and Images Processing*, **30**, pp. 84–106, 1985.

[13] Monagan, G. and Roosli, M. "Appropriate base representation using a run graph," *Proceedings of the 2nd International Conference on Document Analysis and Recognition*, Tsukuba, Japan, pp. 623–626, 1993.

[14] Bley, H. "Segmentation and preprocessing of electrical schematics using picture graphs," *Computer Vision, Graphics and Image Processing*, **28**, pp. 271–288, 1984.

[15] Zenzo, S.D., Cinque, L. and Levialdi, S. "Run-based algorithms for binary image analysis and processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(1), pp. 83–89, 1996.

[16] Tsoukalas, L. H. and Uhrig, R. E. *Fuzzy and Neural Approaches in Engineering*, John Wiley & Sons, Inc., New York, 1997.

[17] Gen, M. and Cheng, R. *Genetic Algorithms and Engineering Design*. John Wiley & Sons, Inc., New York, 1997.

[18] Wesley, M. A. and Markowsky, G. "Fleshing out projections," *IBM Journal of Research and Development*, **25**(6), pp. 934–954, 1981.

[19] Preiss, K. "Constructing the solid representation from engineering projections," *Computers and Graphics*, **8**(4), pp. 381–389, 1984.
[20] Yoshiura, H., Fujimura, K. and Kumii, L. "Top-down construction of 3-D mechanical object shapes from engineering drawings," *IEEE Computer Magazine*, **17**(12), pp. 32–40, 1984.
[21] Ah-Soon, C. and Tombre, K. "A step towards reconstruction of 3-D CAD models from engineering drawings," *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, pp. 331–334, 1995.
[22] Hoffmann, C. M. and Hopcroft, J. E. H. "Geometric ambiguities in boundary representations," *Computer-Aided Design*, **19**(3), pp. 141–147, 1987.
[23] Chen, Z. and Perng, D. B. "Automatic reconstruction of 3D solid objects from 2D orthographic views," *Pattern Recognition*, **21**(5), pp. 439–449, 1988.
[24] Shum, S. S. P., Lau, W. S., Yuen, M. M. F. and Yu, K. M. "Solid reconstruction from orthographic views using 2-stage extrusion," *Computer-Aided Design*, **33**, pp. 91–102, 2001.
[25] Chen, K. Z. and Feng, X. A. "Solid model reconstruction from engineering paper drawing using genetic algorithms," *Computer-Aided Design*, **35**(13), pp. 1235–1248, 2003.
[26] Birkes, D. and Dodge, Y. *Alternative Methods of Regression*. John Wiley & Sons, Inc., New York, 1993.
[27] Thomas, S. M. and Chan, Y. T. "A simple approach for the estimation of circular arc center and its radius," *Computer Vision Graphics Image Process*, **45**, pp. 362–370, 1989.
[28] Guo, Z. and Zhuang, Z. "Map vectorization algorithm based on GAs," *Journal of China University of Science and Technology*, **28**(4), pp. 476–481, 1998.
[29] Srinivas, M. and Patnaik, L. M. "Adaptive probabilities of crossover and mutation in genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics*, **24**(4), pp. 656–667, 1994.
[30] Chen, K. Z., Zhang, X. W., Ou, Z. Y. and Feng, X. A. "Holo-extraction of information from paper drawings for 3D reconstruction," *Computer-Aided Design*, **34**(9), pp. 665–677, 2002.
[31] Chen, K. Z., Zhang, X. W., Ou, Z. Y. and Feng, X. A. "Recognition of digital curves scanned from paper drawings using genetic algorithms," *Pattern Recognition*, **36**(1), pp. 123–130, 2003.

# 20

# Topological Segmentation and Smoothing of Discrete Curve Skeletons

**Wenjie Xie**
**Renato Perucchio**
Department of Mechanical Engineering and Biomedical Engineering,
University of Rochester, Box 270132, Rochester NY 14627, USA

**David Sedmera**
**Robert P. Thompson**
Cell Biology and Anatomy, Medical University of South Carolina,
Charleston, SC 29425, USA

*A skeleton modeling procedure for computing the continuous structural orientation from the discrete curve skeleton yielded by 3D image thinning is presented. The proposed modeling procedure consists of three consecutive operations: (1) topological segmentation to divide a curve skeleton into branches; (2) branch filtering to remove spurious branches induced by noise; and (3) data smoothing to correct local noise distortions and to build a polynomial curve representation. The structural orientation is computed directly based on the curve representation and then applied to the entire 3D image through a uniform skeleton dilation.*

## 1. Introduction

In a recent paper [1] we introduced a new 3D topology-preserving thinning algorithm specifically designed for the reduction of complex 3D digital images of trabeculated biological tissues to 1D voxel representations (curve skeletons). The algorithm is based on a rigorously defined classification procedure and follows a directional subiteration approach. Central to the algorithm is a set of simple, albeit rigorous and efficient, conditions for identifying ambiguous simple sets to beoperated upon in

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

a parallel thinning procedure. The algorithm is shown to yield skeletons only moderately sensitive to noise and to the image rotation.

Thinning, defined as an iterative reduction process that transforms a discrete structure to its lower dimension, is an important operation in both 2D and 3D digital image processing, especially in biomedical image reduction [2–5]. We use thinning to determine the local structural orientation of trabeculated biological tissues, such as those found in the embryonic heart [6] and in trabecular bone [7]. According to experimental observations, the mechanical properties of trabeculated tissues depend greatly on the structural orientation. In particular, the accurate measurement of structural orientation is critical for the finite element modeling of the trabeculated myocardial tissue for biomechanical studies of early heart development [8–10]. Structural orientation can be computed only on the 1D representation derived from the original 3D image of the trabeculated tissue. To solve this problem, we developed a topology-preserving 3D thinning algorithm for computing directly the curve skeleton of a trabeculated tissue from its 3D image, based on the definition of 3D simple (removable) voxels [1]. In the present chapter we extend and refine our modeling procedure by developing algorithms for topological segmentation, branch filtering and data smoothing of discrete 3D curve skeletons obtained through image thinning.

Skeleton segmentation is the basic operation in the postprocessing of a discrete skeleton, either for extracting topological properties of the original image [11] or for filtering spurious skeleton branches [12]. We divide the curve skeleton into linear paths on which branch filtering and smoothing can be performed. The major challenge in the curve skeleton segmentation is the topological classification of skeleton junctions. To address this problem, we introduce a new two-phase junction detection algorithm.

The structural orientation can be evaluated by computing the slope at the centroid of each skeleton voxel. However, due to the discrete nature of digital images, continuous and smooth structural orientation cannot be determined directly from the *raw* skeleton, which is a subset of the original image. Also, image noise may induce spurious skeleton branches and significant geometrical distortions. In order to overcome these problems, we develop a global data regression procedure, which is capable of creating a stable polynomial curve representation from sparse and nonuniform data.

Finally, in the numerical modeling of biological structures with complex geometry, such as trabecular bone and myocardium, computational meshes are often directly derived from digital images interpreted as voxel sets [7,9]. When this happens, it becomes necessary to apply the structural orientation computed on the skeleton back onto the original voxel set. Our thinning accomplishes this through a simple skeleton dilation operation.

In the following sections, we first review the basic notions and definitions pertaining to a 3D digital image and its curve skeleton. Then we introduce the three major operational components – segmentation, filtering and smoothing – constituting the skeleton refining procedure. Finally, we present two sets of results obtained respectively with trabecular myocardium and bone specimens.

## 2. Basic Definitions

In [1] we established the basic definitions and notation related to the 3D digital image and the thinning process. Here, we summarize the notation that will be used in the present chapter.

- $(P, B, W)$ denotes a binary 3D digital image $P$, consisting of a finite array of solid cubic elements, or *voxels*. $B(B \subset P)$, called the original object, is the set containing all *black* or *object* voxels in the image $P$. $W(W = P - B)$ is the collection of all *white* or *background* voxels.
- $p(x_1, x_2, x_3)$ denotes a single voxel $p$, where $x_1$, $x_2$ and $x_3$ are the coordinates of the voxel centroid in the 3D Cartesian space.
- *α-adjacency* describes the relation between two individual voxels [13]. Two distinct voxels $p_1(x_1, x_2, x_3)$ and $p_2(y_1, y_2, y_3)$ are said to be *26-adjacent* if $|x_i - y_i| = 0$ or 1 for $i = 1, 2, 3$, *18-adjacent* if the voxels are 26-adjacent and $\sum_{i=1}^{3} |x_i - y_i| = 2$, or *6-adjacent* if $\sum_{i=1}^{3} |x_i - y_i| = 1$.

- $N^{\alpha}(p)$, called the *α-dilation* of $p$, is the voxel set consisting of $p$ and all its $\alpha$-neighbors.
- $N_{e}^{\alpha}(p) = N^{\alpha}(p) - p$ defines the *α-envelope* of $p$, i.e. the set of all $\alpha$-neighbors of $p$.
- $S$ (or any other capital letter) is used to denote a voxel set. The notion of $\alpha$-adjacency can be extended directly to voxel sets. Two distinct voxel sets $S_1$ and $S_2$ are $\alpha$-adjacent if there exist two $\alpha$-adjacent voxels $p_1 \in S_1$ and $p_2 \in S_2$. $S$ is *α-connected* if $S$ cannot be partitioned into two subsets that are not $\alpha$-adjacent.
- An *α-component* of voxel set $S$ is an $\alpha$-connected subset of $S$ which is *not* $\alpha$-adjacent to any other voxels in $S$. To avoid ambiguity, we adopt the definition of a 26-component for black voxels and a 6-component for white voxels.
- $N_{e}^{\alpha}(S)$ denotes the $\alpha$-envelope of a voxel set $S = \{p_i | i = 0, \ldots, n\}$:

$$N_{e}^{\alpha}(S) = \bigcup_{p_i \in S} N^{\alpha}(p_i) - S \tag{20.1}$$

- An *α-path* is an $\alpha$-connected voxel set $\{p_0, p_1, \ldots, p_n\}$ with $p_i$ $\alpha$-adjacent to $p_{i-1}$ for $i = 1, \ldots, n$. A path is said to be *closed* if $p_0$ is $\alpha$-adjacent to $p_n$, or *linear* if $p_0$ is *not* $\alpha$-adjacent to $p_n$.
- $f_{v}^{26b}(p)$ is the number of black *voxels* within $N_{e}^{26}(p)$.
- $f_{c}^{26b}(p)$ is the number of black *components* within $N_{e}^{26}(p)$.
- $f_{c}^{18w}(p)$ is the number of white components within $N_{e}^{18}(p)$, each containing at least one white 6-neighbor of $p$. If $f_{c}^{18w}(p) = 0$, all 6-neighbors of $p$ are black, and thus $p$ is a *3D internal* voxel. If $f_{c}^{18w}(p) > 1$, $p$ is classified as a *surface internal* voxel [14]. Conversely, $p$ is a *border* voxel if $f_{c}^{18w}(p) = 1$.
- $max(\{f_i | i = 1, \ldots, n\})$ is the maximum value of a set of scalars $\{f_i | i = 1, \ldots, n\}$.

The skeleton $F$, produced by the curve thinning of an image $(P, B, W)$, is called a *raw* curve skeleton of $P$. For simplicity, hereafter we omit the term curve in referring to the 3D skeleton. As a *voxel representation*, $F$ is represented as a subset of $B$, containing all non-removable black voxels after the thinning process is completed. Accordingly, voxels in $F$ are called skeleton voxels. Since the original black set $B$ is 26-connected, the skeleton $F$ is also 26-connected. When concerned only with the geometry of $F$, we use a *point representation*, which describes $F$ as the collection of voxel centroids joined with straight lines (Figure 20.1). While the voxel representation is essential for the topological processing of the skeleton (e.g. skeleton segmentation), the point representation provides the necessary basis for geometrical operations on the skeleton (e.g. skeleton smoothing).

Generally, the raw skeleton derived from the 3D image of a trabecular tissue specimen is a complex tree-like structure of inter connected *branches*. A skeleton branch, which is the basic entity to be

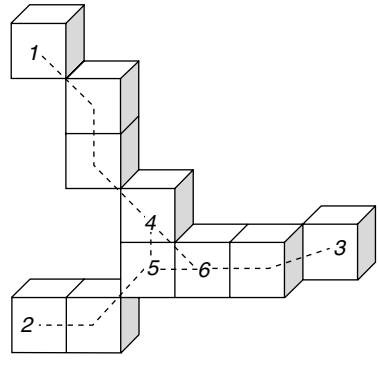



**Figure 20.1** Voxel (solid cubes) and point (dashed lines) representations of a raw curve skeleton.

operated upon in both topological and geometrical processing of the raw skeleton, is defined as follows:

*Definition 1*: A subset of the skeleton $F$ is called a *branch* if it is a 26-path in $F$ and contains two skeleton *ends*. A branch is classified as *linear* or *closed* if it can be represented by a linear 26-path or by a closed 26-path, respectively.

The first voxel $p_0$ and the last voxel $p_n$ of a linear branch $\{p_0, p_1, .., p_n\}$ correspond to the two skeleton ends. However, there is no unique definition of end voxels in a closed branch. We will address this problem in the next section.

We cannot directly evaluate the structural orientation from the raw skeleton for the following reasons:

1. With a point representation, a branch of a raw skeleton $F$ can be viewed as a *0*th continuous curve formed with straight-line segments connecting the centroids of the skeleton voxels. Thus, the first spatial derivatives, representing the structural orientation, are discontinuous at each voxel centroid shared by two nonparallel line segments.
2. The noise effects, in the form of spurious branches and local geometrical distortions, may severely undermine the accuracy of the computed structural orientation.

We solve this problem by refining the raw skeleton through topological segmentation, branch filtering and branch smoothing.

## 3. Topological Segmentation

### *3.1 Component Counting and Labeling*

Component counting is an important operation in general 2D and 3D digital image processing. Many advanced component-counting algorithms have been introduced [15–17]. In this work, we expand a simple recursive scanning procedure called an *α-scan* [1] for counting and labeling components. An $\alpha$-scan over a voxel set $S$ involves the following steps:

1. Initialize the scanning front $S' : S' \subset S$.
2. Set initial scanned set $A : A = S'$.
3. Find the $\alpha$-envelope $E$ of $S'$ within $S : E = N_e^{\alpha}(S') \cap S$.
4. Update first the scanning front $S'$ and then the scanned set $A : S' = E - A$, $A = A \cup E$.
5. Check the conditions for terminating the $\alpha$-scan. If stopping conditions are not satisfied, update the scanned set $A(A = A \cup E)$ and go back to step 3.

For an *unrestricted* $\alpha$-scan, the stopping condition is simply $S' = 0$. The final scanned set $A$ obtained from an unrestricted $\alpha$-scan is an $\alpha$-component in $S$ if the initial scanning front $S'$ is $\alpha$-connected. Obviously, a single voxel in $S$ constitutes the smallest $\alpha$-connected initial scanning front.

*Component labeling* is the process of applying a certain property, e.g. an integer number indicating uniquely a component, to the final scanned set. This type of operation can be easily achieved by gradually assigning the property associated with the initial scanning front to the scanned set in an $\alpha$-scan.

### *3.2 Classification of Skeleton Voxels*

We need to determine the topological classification of each skeleton voxel before we can segment the raw skeleton $F$ into branches. A skeleton voxel can be classified as a *node, tip* or *junction*. The definitions of *node* and *tip* are as follows:

*Definition 2*: A skeleton voxel $p$ is called a branch *node* if $p$ is 26-adjacent to two skeleton voxels, i.e. $f_v^{26b}(p) = 2$.

*Definition 3*: A skeleton voxel $p$ is called a branch *tip* if $p$ is 26-adjacent to one skeleton voxel, i.e. $f_{\mathrm{v}}^{26\mathrm{b}}(p) = 1$.

The definition of a branch *junction* is not as straightforward. Conceptually, within a voxel representation of the skeleton, a junction is a voxel linking three or more branches. Two typical junction definitions can be found in the literature:

*Definition*: A *voxel-based* definition [12]: a junction is a skeleton voxel 26-adjacent to more than two skeleton voxels, i.e. $f_{\mathrm{v}}^{26\mathrm{b}}(p) > 2$.

*Definition*: A *component-based* definition [11,14]: a junction is a skeleton voxel 26-adjacent to more than two black components within $N_{\mathrm{e}}^{26}(p)$, i.e. $f_{\mathrm{c}}^{26\mathrm{b}}(p) > 2$.

Both definitions are based on the voxel adjacency being restricted to the local $3 \times 3 \times 3$ neighborhood of a skeleton voxel under classification. Neither of these two definitions leads to a unique and comprehensive detection of all valid junctions. As shown in Figure 20.1, while voxels 4, 5 and 6 are classified as junctions with a voxel-based definition, none of them is recognized as a junction with a component-based definition. This ambiguity in the junction classification is specifically associated with 'thick' junctions, each consisting of two or more 26-connected junctions detected with a voxel-based definition. We will give later a rigorous definition of thick junctions. Presently, in order to solve this problem, we introduce a two-phase junction classification procedure, consisting of a local classification step and a thick junction resolution step.

## *3.3 Local Junction Classification*

At the local level, we identify those voxels that are either an unambiguous junction or a junction *candidate* requiring further investigation. The local classification is based on the following definitions:

*Definition 4*: A skeleton voxel $p$ is a *junction* if it satisfies conditions: (1) $f_{\mathrm{c}}^{26\mathrm{b}}(p) > 2$ or (2) $f_{\mathrm{c}}^{18\mathrm{w}}(p) \neq 1$ (i.e. voxel $p$ is internal).

*Definition 5*: A skeleton voxel $p$ is a *junction candidate* if it satisfies conditions: (1) $f_{\mathrm{c}}^{26\mathrm{b}}(p) \leq 2$, (2) $f_{\mathrm{c}}^{18\mathrm{w}}(p) = 1$ (i.e. voxel $p$ is a border voxel), and (3) $p$ is not 26- adjacent to any junctions.

*Observation 1*: Our curve thinning algorithm reduces a 3D image to single-stranded border voxels by progressively transforming 3D and surface internal voxels into border voxels [1]. However, the resulting skeleton $F$ may contain an internal voxel $p$ if $p$ is surrounded by other skeleton voxels. Figure 20.2 illustrates a typical 3D nonreducible voxel configuration derived from the 2D example in [18]. A surface skeleton voxel $p$ is naturally identified as a junction, since $p$ is the merging location of its neighboring skeleton voxels in a local surface-like configuration. Adopting this classification approach, we can always segment a surface- like skeleton into 26-paths, which then can be stored with compact formats, e.g. a chain code [19], and further processed to determine the structural orientation.

The 26-connected junction candidates are the source of ambiguity in the junction classification. Let $S^{\mathrm{c}}$ be the voxel set containing all junction candidates identified in the local classification. The following definition of a *thick* junction is adopted:

*Definition 6*: A 26-connected voxel set $T = \{p_i | i = 1, \ldots, n\}$, where $n > 1$ and $T \subset S^{\mathrm{c}}$, is called a *thick* junction if there exists a junction candidate $p_{\mathrm{j}} \in T$ that satisfies the conditions:

1. $T \subset N^{26}(p_{\mathrm{j}}) \cap S^{\mathrm{c}}$
2. $f_{\mathrm{v}}^{26\mathrm{b}}(p_{\mathrm{j}}) = \max(\{f_{\mathrm{v}}^{26\mathrm{b}}(p_i) | i = 1, \ldots, n\})$.

We call such a voxel $p_{\mathrm{j}}$ the *core* of the thick junction $T$. This definition allows us to associate a single junction (the core) to each thick junction.

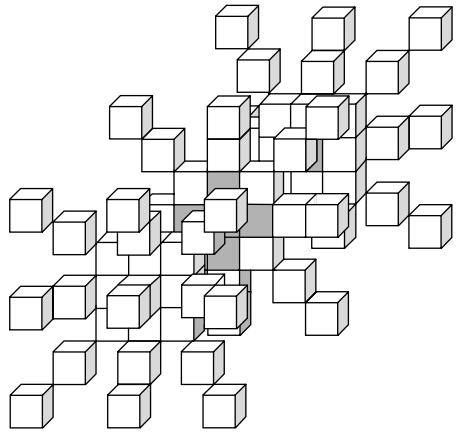


**Figure 20.2** A voxel configuration which cannot be reduced to a unit-width skeleton with a 3D thinning algorithm based on 26-adjacency for black voxels. Shaded voxels are 3D internal skeleton voxels.

## *3.4 Thick Junction Resolution*

In the second phase of junction classification, we eliminate thick junctions by identifying a core from each thick junction and restoring the status of the other junction candidates to branch nodes.

Lee *et al.* [12] proposed a local procedure for resolving thick junctions in a 3D curve skeleton and a similar approach can be found in Abdullah *et al.* [20] for 2D junction classification. According to this approach, any junction candidate $p$ first encountered in the sequential visit of $S^c$ may be selected as a junction, and all other junction candidates 26-adjacent to $p$ are then transformed into nodes. Doing so, however, may produce redundant junctions if the selected voxel $p$ is not the *core* of the thick junction. As shown in Figure 20.3, voxels $p$, $q$, $r$ and $s$ are all junction candidates according to Definition 5. These junction candidates form a single thick junction with voxel $r$, $f_v^{26b}(r) = 4$, as the junction core (according to Definition 6). However, if Lee's algorithm happens to select voxel $p$, $f_v^{26b}(p) = 3$, as the

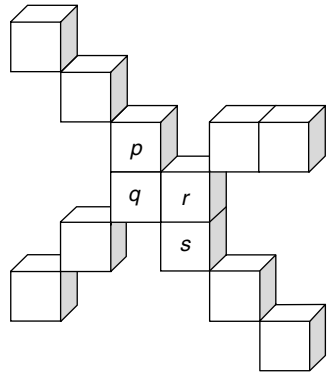



**Figure 20.3** An example showing that the random selection of a junction from junction candidates may lead to the creation of redundant branches.

junction, voxels $q$ and $r$ are transformed into nodes, and the remaining candidate $s$ is then identified as another junction in a new sequential visit of the candidate junction set, likely yielding an undesirable skeleton model.

To overcome this deficiency, we propose a global approach for the resolution of thick junctions:

1. Re-sort the candidate junction set $S^{\mathrm{c}} = \{p_i | i = 1, \ldots, n\}$ in a descending order by the value of $f_{\mathrm{v}}^{26\mathrm{b}}$ of each voxel in $S^{\mathrm{c}}$ such that $f_{\mathrm{v}}^{26\mathrm{b}}(p_i) \geq f_{\mathrm{v}}^{26\mathrm{b}}(p_{i+1})$ for $i = 1, \ldots, n-1$.
2. Sequentially visit each voxel $p$ in $S^{\mathrm{c}}$ and
   (a) set $p$ as a junction;
   (b) transform all junction candidates in $N_{\mathrm{e}}^{26}(p) \cap S^{\mathrm{c}}$ to nodes; and
   (c) update candidate voxel set $S^{\mathrm{c}} : S^{\mathrm{c}} = S^{\mathrm{c}} - N^{26}(p) \cap S^{\mathrm{c}}$.

## 3.5 Branch Formation

After all skeleton voxels are classified, the raw skeleton is ready for the formation of branches. Here, we introduce two additional values associated with a skeleton voxel:

*Definition*: $f^{\mathrm{t}}(p)$ is the number of branches containing a skeleton voxel $p$. If $p$ is a tip or node, then $f^{\mathrm{t}}(p) = 1$. If $p$ is a junction, then $f^{\mathrm{t}}(p) = f_{\mathrm{v}}^{26\mathrm{b}}(p)$.

*Definition*: $f^{\mathrm{s}}(p)$ indicates the number of times a voxel $p$ has been visited in the skeleton segmentation process. Assuming each skeleton voxel $p$ is visited only once in the formation of each single branch containing voxel $p$, we call $p$ *unsaturated* if $f^{\mathrm{s}}(p) < f^{\mathrm{t}}(p)$, or *saturated* if $f^{\mathrm{s}}(p) = f^{\mathrm{t}}(p)$ (i.e. all branches containing $p$ have been determined).

Let $S^{\mathrm{u}}$ be the set consisting of all unsaturated skeleton voxels, and voxel $p$ ($f^{\mathrm{t}}(p) = n$, $f^{\mathrm{s}}(p) = m$ and $m < n$) be an end in $S^{\mathrm{u}}$. We build the $(m-n)$ undetermined branches containing $p$ with the following branch formation procedure:

1. Identify a voxel set $E$, which consists of all unsaturated skeleton voxels within $N^{26}(p) : E = S^{\mathrm{u}} \cap N^{26}(p)$.
2. Form a branch $L$, which contains an unsaturated skeleton voxel $e (e \in E$ and $e \neq p)$:
   (a) Increase $f^{\mathrm{s}}(e)$ and $f^{\mathrm{s}}(p)$ by one.
   (b) If $e$ is an *end*, build the branch $L$, simply with voxels $p$ and $e : L = \{p, e\}$, Figure 20.4(a).
   (c) If $e$ is a *node*, then perform a 26-scan over the voxel set $S^{\mathrm{u}}$ with the voxel $e$ as the initial scanning front to produce a final scanned set $A$. Voxel set $E$ is excluded in the first scan iteration. The 26-scan stops when one or more unsaturated ends are reached – Figure 20.4(b).
   (d) Increase $f^{\mathrm{s}}(p_i)$ by one, for each skeleton voxel $p_i$ in the scanned set $A$.
   (e) Let $T$ be a voxel set containing all unsaturated end voxels detected in this 26-scan. We increase $f^{\mathrm{s}}$ of all end voxels in $T$ by one and pick any voxel $q$ in $T$ as the terminating end for the branch $L$. Thus, the branch $L$ under determination is given by $\{p, A, q\}$. Note that this branch determination procedure can also detect closed branches containing only one distinct skeleton end, i.e. $p = q$ – Figure 20.4(c).
   (f) Update $E : E = E - \{e\}$.
3. Repeat branch identification (steps 1 and 2) until the end voxel $p$ is saturated.

By performing the above procedure on each unsaturated skeleton end, we segment into branches a skeleton containing at least one skeleton end. However, this procedure is not suitable for a skeleton $F$ that corresponds to a single *isolated* closed path. We solve this problem simply by picking any skeleton voxel $p$ in $F$ as a virtual junction $f^{\mathrm{t}}(p) = 2$ and then use the above procedure to form a closed path starting with $p$ (similar to the case illustrated in Figure 20.4(c)).

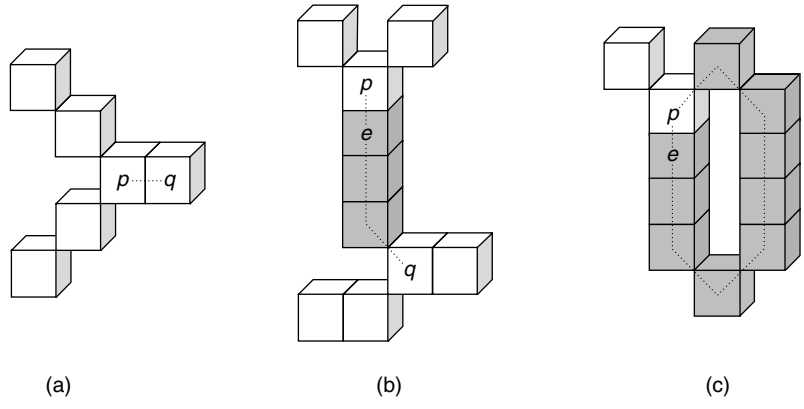



**Figure 20.4** Determination of a branch $L$ containing an end voxel $p$: (a) $L$ consists of two ends only; (b) $L$ consists of two distinct ends and a string of nodes (shaded voxels) identified in a 26-scan starting with voxel $e$; (c) the closed branch $L$ contains only one junction.

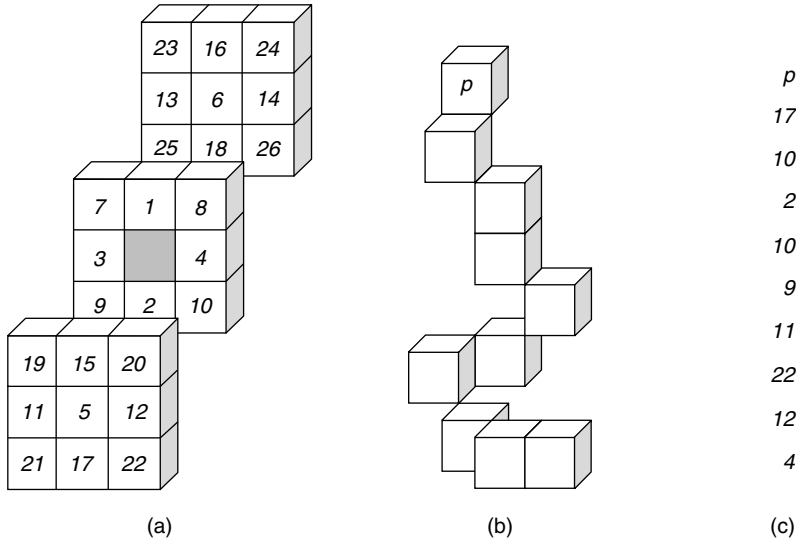



**Figure 20.5** 26-chain representation of a branch starting with end voxel $p$. (a) Directional code for the 26-neighborhood of voxel $p$; (b) a voxel representation of a branch; and (c) the corresponding 26-chain code.

Since all voxels are geometrically identical, we use a 26-chain code derived from the 4-chain code in [21] to describe the 26-connected branches (Figure 20.5). The advantages of using a chain code representation are obvious: (1) it is *compact* – a 26-chain code is a list of small integers ranging from 1 to 26; and (2) Lit is *portable* – with a chain code representation and the known physical dimensions of a voxel, the connectivity and geometry of a branch can be easily and quickly reconstructed.

# 4. Branch Filtering

## *4.1 Branch Classification*

We call a branch produced by the image noise a *noise branch*. An excessive number of noise branches leads to an undesirable skeleton representation of the original image. Therefore, noise branches must be identified and filtered out of the skeleton.

*Definition 7*: A branch $L$ is said to be *free* if $L$ includes at least one skeleton tip, or *fixed* if $L$ contains no skeleton tips.

*Observation 2*: Curve skeletons are suitable for describing elongated objects. For a 3D solid object without cavity, its topological property can be defined in terms of the numbers of components and tunnels [22]. A topology-preserving thinning algorithm produces a skeleton which has the same topological property as the original object. Let $L$ be a branch of the skeleton $F$. If $L$ is free, the removal of $L$ does not lead to the complete elimination of a component in $F$ unless $L = F$, or the deletion of a tunnel, since $L$ is not included in any closed path in $F$ [23]. However, deleting a fixed branch $L$ may break up a component in $F$ if $L$ links two branches that cannot be joined through other paths, or may destroy a tunnel if $L$ is contained in a closed path in $F$.

Based on this observation, in order to preserve the topology of the skeleton we only consider free branches for filtering. Note that deleting a free branch $L$ may transform fixed branches adjacent to $L$ into free branches. Therefore, an iterative branch filtering procedure is desirable when the compound noise effect is significant.

A free branch can be further classified based on the following definition:

*Definition 8*: A free branch $L$ is called a *structural*, *type-1 noise* or *type-2 noise* branch if $L$ has complete, empty, or partial overlap with the geometrical midline of the original object (Figure 20.6).

Type-1 noise branches are approximately perpendicular to the midline of the original object and have a characteristic short length, i.e. half the local width of the object. Based on this property of type-1 noise branches, Lee *et al.* propose a simple branch filtering criterion [12]. If the length of a free branch $L$, in terms of the total number of voxels in $L$,is below a critical value $c$, $L$ is identified as a noise branch and removed. However, this length-based criterion suffers several drawbacks:

1. A viable critical branch length $c$ is difficult to determine.
2. Short structural branches (such as branch 1 in Figure 20.6) may be identified mistakenly as noise branches.
3. Long type-2 noise branches (e.g. branch 3 in Figure 20.6) cannot be detected and corrected.

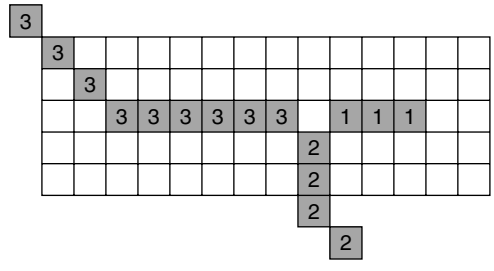



**Figure 20.6** Three types of free branch of an elongated object: (1) a structural branch (branch 1); (2) a type-1 noise branch (branch 2); and (3) a type-2 noise branch (branch 3).

In order to overcome these problems, we propose a new branch filtering procedure based on the thickness instead of the length of a branch. The definition of thickness associated with a skeleton voxel or a branch is given as follows:

*Definition 9*: T he thickness of voxel $p$, denoted by $t(p)$, is the number of thinning iterations performed before $p$ is determined as a skeleton voxel – see the definition of our thinning procedure in [1]. Accordingly, the thickness of branch $L = \{p_1, \ldots, p_n\}$, denoted by $t(L)$, is the averaged thickness of all voxels in $L$, i.e. $t(L) = \text{round}(\sum_{j=1}^{n} t(p_j)/n + 0.5)$.

We notice that a noise branch usually starts off a short protrusion on the surface of the original object and then penetrates into the object until it collides with the geometrical midline (see Figure 20.6). We call a 26-connected subset $N$ of a noise branch $L$ a *noise segment* of $L$, if $N$ is composed only with skeleton voxels that are not located at the midline of the object. Accordingly, the subset $S = L - N$ is called the *structural segment* of $L$. Note that a noise branch $L$ may have two noise segments each starting with an end of $L$. Clearly, a type-1 noise branch has a void structural segment.

We propose now a viable mathematical basis for detecting noise branches and determining the length of a noise segment. Let $L = \{p_1, \ldots, p_m, \ldots, p_n\}$ be a generic noise branch and the voxel sets $\{p_1, \ldots, p_m\}$ and $\{p_{m+1}, \ldots, p_n\}$ denote the noise and structural segments of branch $L$, respectively. We make the following simplifying assumption on the architecture of branch $L$:

1. $t(p_{i+1}) - t(p_i) = 1$ for $i = 1, \ldots, m-1$. Therefore,
2. $t(p_m) = m + t(p_1) - 1$, and
3. $t(p_i) = t(p_m)$ for $i = m+1, \ldots, n$.

On equating this assumed local thickness distribution of $L$ with the average branch thickness, we obtain

$$(m + t(p_1) - 1)! - (t(p_1) - 1)! + (m + t(p_1) - 1) \times (n - m) = t(L) \times n \tag{20.2}$$

from which the length of the noise segment $m$ can be determined. Since Equation (20.2) is nonlinear in $m$, a numerical solution procedure is required. For simplicity, we approximate Equation (20.2) as:

$$m = t(L) + \text{round}\left(\sum_{i=1}^{n} \frac{|t(L) - t(p_i)|}{n} + 0.5\right) - t(p_1) + 1 \tag{20.3}$$

where the second term is the adjustment of the branch thickness $t(L)$ to compensate the variation induced by the nonuniform noise segment. For example, consider branch 3 in Figure 20.6. The local thickness $t(p)$ associated with each voxel in branch 3 is $\{0, 1, 2, 2, 2, 2, 2, 2, 2\}$ and $t(L) = 2$. Applying Equation (20.3) we obtain $m = 2 + 1 - 0 + 1 = 4$, while the approximate solution of Equation (20.2) yields $m = 3.95$.

## *4.2 Noise Segment Removal*

We now introduce our thickness-based branch filtering procedure, structured as follows:

1. Sequentially examine each free branch $L = \{p_1, \ldots, p_n\}$ in the skeleton $F$.
2. Check each tip of $L$ for the start of a noise segment. If voxel $p_1$ is a tip and $t(p_1) \leq 0.25 \times t(L)$, compute the length $m$ using Equation (20.3) and remove a noise segment $L' = \{p_1, \ldots, p_m\}$ from $L$.

Similarly, if $p_n$ is detected as a noise tip, a subset $L' = \{p_m, \ldots, p_n\}$ of $L$ is selected as a noise segment, where $m$ is given by:

$$m = n - t(L) - \text{round}\left(\sum_{i=1}^{n} \frac{|t(L) - t(p_i)|}{n} + 0.5\right) + t(p_n) \tag{20.4}$$

Removing the noise segment $L'$ involves the following steps:

(a) Delete all branch nodes and branch tips in $L'$.

(b) Decrease $f^{\mathrm{t}}(p)$ by one for any junction $p$ in $L'$. Based on the new value of $f^{\mathrm{t}}(p)$, the junction $p$ is either deleted (if $f^{\mathrm{t}}(p) = 0$) or transformed into a tip or node (if $f^{\mathrm{t}}(p) = 1$).

A change in the status of a junction $p$ may affect the status of a branch containing $p$. Thus, a branch segmentation must be performed again after branch filtering is completed to properly register all skeleton voxels and branches.

Figure 20.7 shows the effect of image noise on the thinning procedure. The resulting raw skeleton exhibits numerous type-1 and type-2 noise branches. As shown in Figure 20.8(a), the above branch filtering procedure effectively filters out both types of noise branch. However, local geometrical

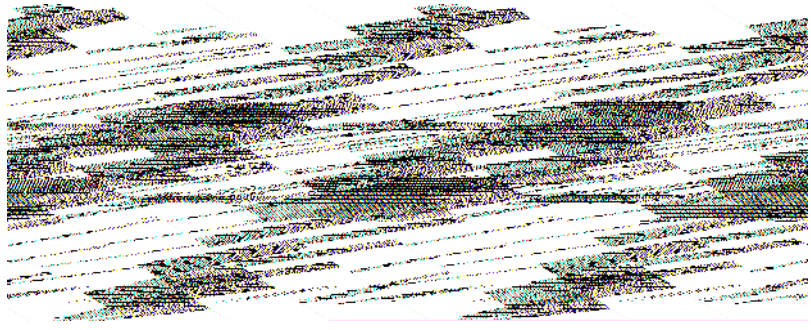


**Figure 20.7** Raw skeleton resulting from a noisy input image. (a) 3D image of an artificial object with 20 % random noise; and (b) raw skeleton with spurious noise branches.

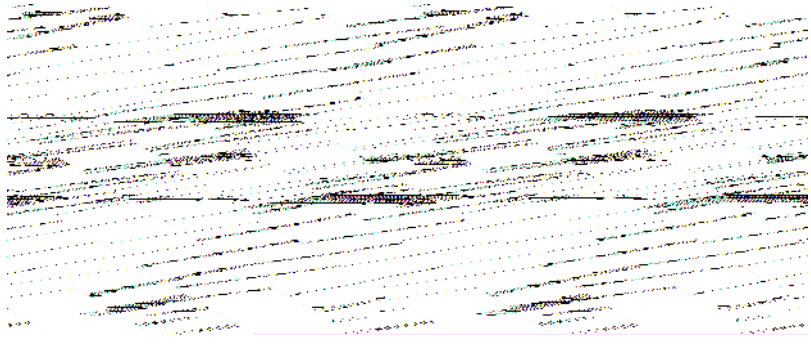


**Figure 20.8** Skeleton after branch filtering. (a) Voxel representation of the filtered skeleton; and (b) point representation showing significant noise distortion.

distortions caused by image noise – clearly visible in Figure 20.8(b) – still need to be corrected. This problem is addressed in the next section.

## 5. Branch Smoothing

### 5.1 Polynomial Branch Representation

Let $L = \{(x_i, y_i, z_i) | i = 0, \ldots, n\}$ be the point representation of a skeleton branch, where $(x_i, y_i, z_i)$ are the global Cartesian coordinates of the centroid of a voxel $p_i$ in $L$. In order to derive the continuous structural orientation from $L$, we first need to construct an $m$-continuous ($m > 1$) curve representation of the branch. This is usually done by constructing $x$, $y$ and $z$ as three independent $m$-continuous functions of a local variable $r$ [24]:

$$x = x(A, r), y = y(B, r), z = z(C, r) \tag{20.5}$$

where $A = \{a_i | i = 1, \ldots, m\}$, $B = \{b_i | i = 1, \ldots, m\}$ and $C = \{c_i | i = 1, \ldots, m\}$ are parameters to be determined through either interpolation or data fitting of the centroidal coordinates of voxels in $L$. A common interpolation algorithm for geometric modeling with discrete data is the cubic piecewise spline interpolant [25]. Although cubic spline interpolation yields a robust curve representation, curve interpolation is in general not effective for data smoothing, since the inclusion of noise data within the control points of the interpolant leads to undesirable oscillatory curve behavior. Data fitting procedures, on the contrary, use the entire set of discrete data to project the trend rather than to simply build a curve that matches exactly the selected original points. Thus, data fitting is usually adopted when data smoothing is concerned. There are two basic approaches for curve fitting: fitting with piecewise base functions, e.g. B-splines [26], and fitting with global base functions, e.g. polynomials defined over the entire data range.

The classical problem in curve fitting with noisy data is how to define the proper flexibility for the selected curve function, so that the fitted curve can adequately predict the trend of the original data, while sufficiently reducing the noise effects. While a piecewise fitting approach controls the curve flexibility by adjusting the number and locations of 'knots' bounding the piecewise base functions, a global fitting procedure achieves the same result by varying the order of the global base functions.

Curve fitting with piecewise base functions involves lower order base functions and hence yields better numerical stability in the fitted function. However, the number and locations of 'knots' are difficult to define, unless a complex nonlinear fitting procedure is employed [26]. Global fitting is relatively simple and leads to a more compact parameter set, although it may suffer from severe numerical fluctuations when the order of the base functions becomes too high.

In this work, we adopt the classical linear least squares fitting method [24] for modeling the raw skeleton. The branch functions in Equation (20.5) are rewritten as follows:

$$x(A, r) = \sum_{i=1}^{m} a_i N_i(r), y(B, r) = \sum_{i=1}^{m} b_i N_i(r), z(C, r) = \sum_{i=1}^{m} c_i N_i(r) \tag{20.6}$$

where $N_i, i = 0, \ldots, m$, are polynomial base functions. We use one-dimensional isoparametric finite element shape functions [27] as the base functions in order to control the numerical fluctuation in the $x$, $y$ and $z$ curves [28]. The normalized indices of the skeleton voxels are selected as the local coordinate $r$, with $0 < r < 1$. Thus, the local coordinate of voxel $p_j$ in branch $L = \{p_0, p_1, \ldots, p_n\}$ is $r(p_j) = j/n$.

## *5.2 Augmented Merit Functions*

With a standard least squares fitting method, the parameter sets $A$, $B$ and $C$ in Equation (20.6) can be computed by minimizing the following three merit independent functions:

$$\begin{aligned}\chi_1(A) &= \sum_{i=1}^{n}[x_i - x(A, r_i)]^2 \\ \chi_2(B) &= \sum_{i=1}^{n}[y_i - y(B, r_i)]^2 \\ \chi_3(C) &= \sum_{i=1}^{n}[z_i - z(C, r_i)]^2\end{aligned} \tag{20.7}$$

where $n$ is the total number of voxels in a branch and $r_i$ is the local coordinate of voxel $p_i = p(x_i, y_i, z_i)$. These standard merit functions are best suited for least squares data modeling with dense and evenly located discrete data. However, nonuniformity in skeleton voxel location and insufficient input data for a high-order polynomial fitting can lead to artificial fluctuations in the resulting curve. In order to solve this problem, we expand the above merit functions with additional controlling terms:

$$\begin{aligned}\chi_1(A) &= Q_1(A) + \sum_{i=1}^{n}[x_i - x(A, r_i)]^2 \\ \chi_2(B) &= Q_2(B) + \sum_{i=1}^{n}[y_i - y(B, r_i)]^2 \\ \chi_3(C) &= Q_3(C) + \sum_{i=1}^{n}[z_i - z(C, r_i)]^2\end{aligned} \tag{20.8}$$

$Q_1, Q_2$ and $Q_3$ are stabilizing penalty functionals designed to impose smoothness in the fitted curve and are defined as:

$$\begin{aligned}Q_1(A) &= \int_0^1 \{\alpha[\frac{\partial x(A, r)}{\partial r}]^2 + \beta[\frac{\partial^2 x(A, r)}{\partial r^2}]^2\}\mathrm{d}r \\ Q_2(B) &= \int_0^1 \{\alpha[\frac{\partial y(B, r)}{\partial r}]^2 + \beta[\frac{\partial^2 y(B, r)}{\partial r^2}]^2\}\mathrm{d}r \\ Q_3(C) &= \int_0^1 \{\alpha[\frac{\partial z(C, r)}{\partial r}]^2 + \beta[\frac{\partial^2 z(C, r)}{\partial r^2}]^2\}\mathrm{d}r\end{aligned} \tag{20.9}$$

where $\alpha$ and $\beta$ are predefined weight coefficients.

The functionals in Equation (20.9) are analogous to the strain energy of an elastic beam subjected to tension (the first term in the integral) and bending (the second term). Thus, the smoothing weight $\alpha$ restricts large stretches in the fitted curve, while $\beta$ restricts excessive bending deformation. The larger the smoothing weights, the 'stiffer' the fitted curve will become (see Figure 20.9). The selection of smoothing weights is heuristic and problem dependent. Based on the classical beam theory, we set $\beta = 2.0 \times \alpha$. Therefore, only one independent smoothing weight $\alpha$ needs to be specified.

The saddle points of each merit function–Equation (20.8)–are obtained through first-order differentiation with respect to the parameter sets $A$, $B$ and $C$, respectively. For example, for the solution of parameters $A$, the condition $\partial\chi_1(A)/\partial a_i = 0$ yields the set of linear algebraic equations

$$t_{ij}a_j = f_i \text{ for } i = 1, \ldots, m \text{ and } j = 1, \ldots, m \tag{20.10}$$

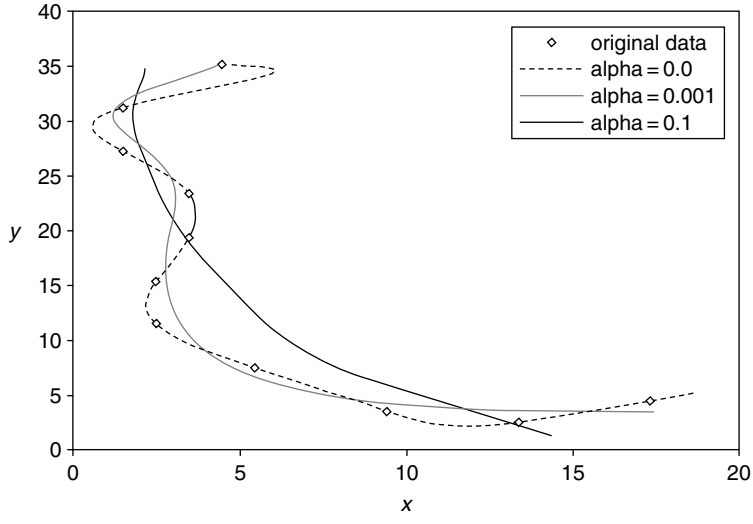



**Figure 20.9** Polynomial curves fitted with various smoothing weights.

where coefficients $t_{ij}$ and $f_i$ are given by:

$$t_{ij} = \sum_{k=1}^{n} N_i(r_k) N_j(r_k) + \int_0^1 [\alpha \frac{\partial N_i(r)}{\partial r} \frac{\partial N_j(r)}{\partial r} + \beta \frac{\partial^2 N_i(r)}{\partial r^2} \frac{\partial^2 N_j(r)}{\partial r^2}] \mathrm{d}r$$

$$f_i = \sum_{k=1}^{n} x_k N_i(r_k) \tag{20.11}$$

The linear system given above can be solved with a standard Gaussian elimination procedure [29]. Similarly, we compute the parameter sets $B$ and $C$ by solving respectively the linear systems obtained from $\partial \chi_2(B)/\partial b_i = 0$ and $\partial \chi_3(C)/\partial c_i = 0$. As shown in Figure 20.10, our global data modeling

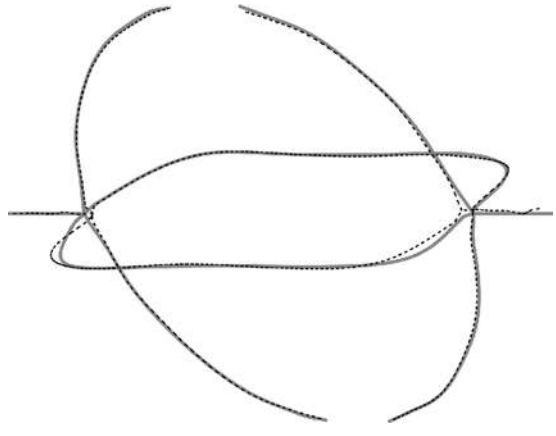


**Figure 20.10** Superimposed smoothed skeletons for the object in Figure 20.7 obtained from an image with random noise (dashed black curve) and an image without noise (thick gray curve).

procedure produces a smooth and stable curve representation of the discrete skeleton and effectively corrects local distortions due to image noise.

## 6. Results

We present the skeleton modeling results for two typical examples of trabecular biological tissue. Shown in Figure 20.11(a) and Figure 20.12(a) are the 3D binary images of trabeculated myocardial tissue in an HH21 chick embryonic heart, and trabecular bone tissue sampled from the human iliac crest, respectively. Figures 20.11(b) and 20.12(b) show the segmented and filtered discrete curve skeletons for the trabeculated myocardium and the trabecular bone. The relevant topological and geometrical

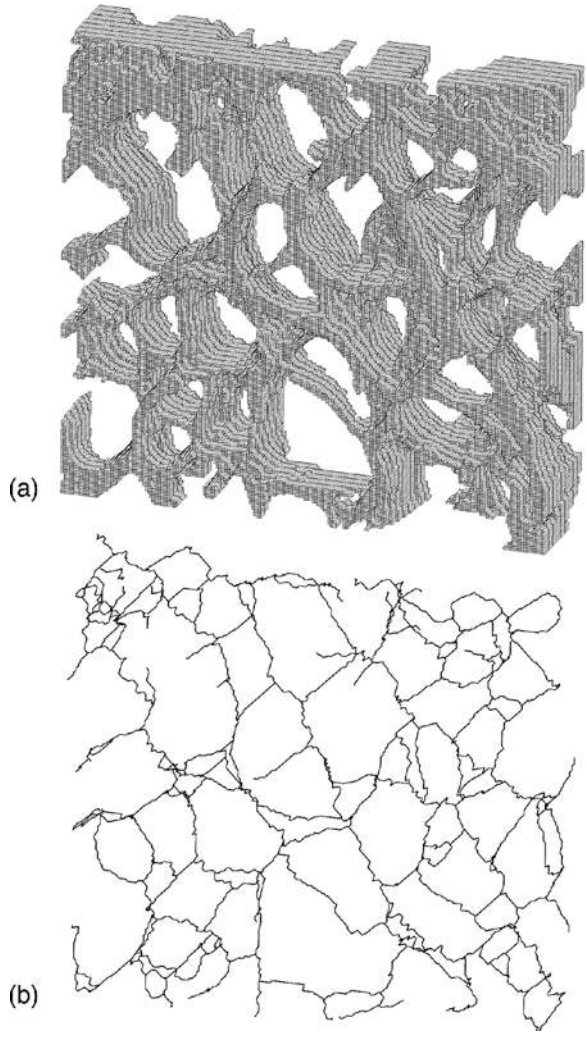



**Figure 20.11** Trabeculated myocardium specimen in an HH stage 21 chick embryo. (a) 3D binary image; and (b) computed raw curve skeleton (point representation).

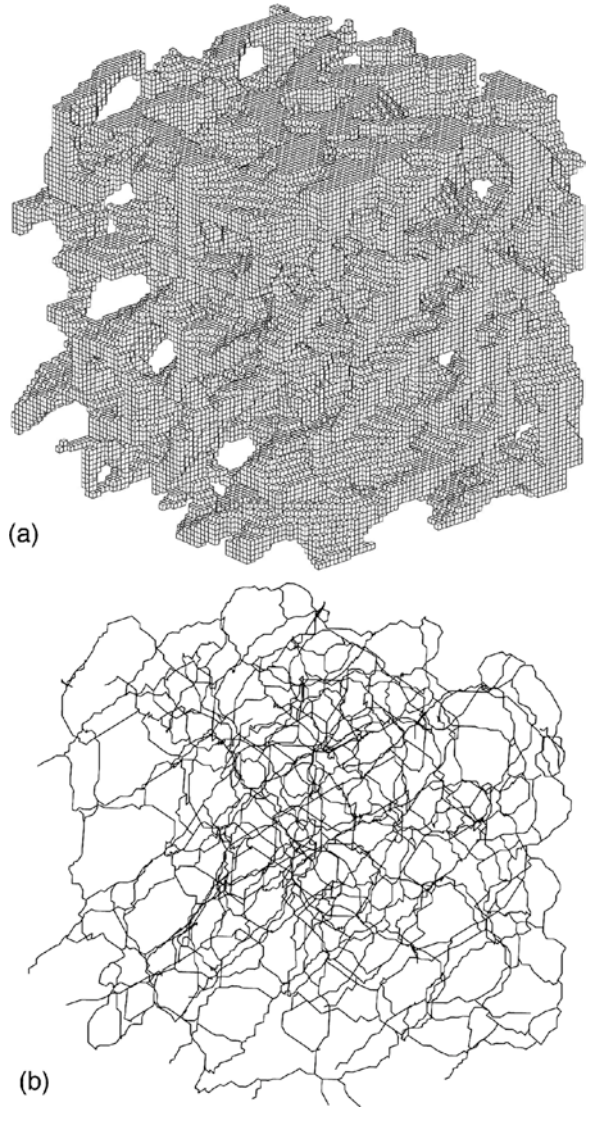



**Figure 20.12** Trabecular bone specimen from a human iliac crest. (a) 3D binary image; and (b) computed raw curve skeleton (point representation).

properties of the two skeletons are given in Table 20.1. The smoothing weight $\alpha$ for the data modeling of a branch is determined with the following heuristic formula:

$$\alpha = \begin{cases} (10^{-4} \times m)/n & \text{trabeculated myocardium} \\ (10^{-3} \times m)/n & \text{trabecular bone} \end{cases} \tag{20.12}$$

where $m$ is the order of polynomial curve function and $n$ is the total number of skeleton voxels in a branch. The smoothed polynomial curve representations of the discrete skeletons are shown in Figure 20.13(a) and Figure 20.13(b).

**Table 20.1** Topological and geometrical properties of skeletons before and after filtering.

| | Specimen 1 | Specimen 2 |
|---|---|---|
| # skeleton voxels | 3895 | 5000 |
| # branches before filtering | 387 | 831 |
| # junctions before filtering | 235 | 525 |
| # tips before filtering | 61 | 47 |
| average branch length | 63.6 microns | 353.5 microns |
| average branch thickness | 34.2 microns | 208.4 microns |
| # type-1 noise branches removed | 6 | 17 |
| # type-2 noise branches truncated | 5 | 6 |
| # voxels deleted in filtering | 86 | 112 |

Specimen 1: Trabeculated myocardium in an HH21 chick embryo (Figure 20.11(a)) Image size: $x \times y \times z = 220 \times 220 \times 9$ voxels Voxel resolution: $3.3 \times 3.3$ microns in the $xy$ plane, 18 microns along $z$

Specimen 2: Trabecular bone tissue from human iliac crest (Figure 20.12(a)) Image size: $x \times y \times z = 77 \times 77 \times 77$ voxels Voxel resolution: $x \times y \times z = 50 \times 50 \times 50$ microns

After the high-order continuous curve function of the skeleton is obtained, we simply evaluate the structural orientation $\overline{v}$ at any location on the skeleton by taking the first derivative of this function relative to the local coordinate $r$:

$$\overline{v}(r) = \frac{\left(\frac{\partial x}{\partial r}, \frac{\partial y}{\partial r}, \frac{\partial y}{\partial r}\right)}{\sqrt{\left(\frac{\partial x}{\partial r}\right)^2 + \left(\frac{\partial y}{\partial r}\right)^2 + \left(\frac{\partial z}{\partial r}\right)^2}} \tag{20.13}$$

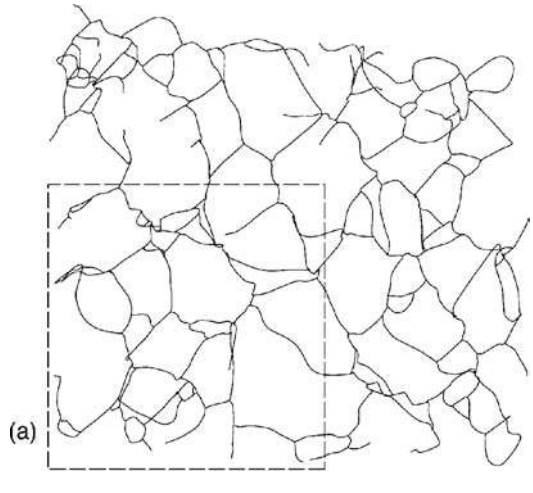



**Figure 20.13** (a) Smoothed polynomial representation of the skeleton shown in Figure 20.11; and (b) structural orientation in the region indicated by dashed lines in (a).

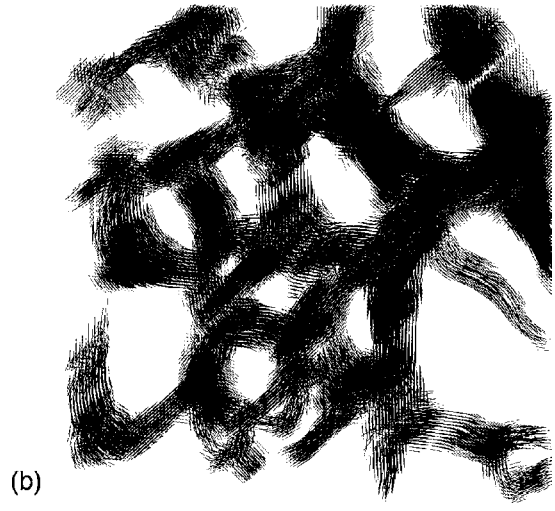



**Figure 20.13** (*continued*)

In order to transfer the structural orientation determined on the skeleton to non-skeleton voxels, the algorithm performs a uniform skeleton dilation, consisting of a 26-scan over the original object set with all skeleton voxels as the initial scanning front, and recursively copies the structural orientation from the visited voxels to the unvisited voxels (*orientation labeling*). A simple averaging operation is performed when an unvisited voxel $p$ is reached by multiple visited black voxels in $N_e^{26}(p)$.

The vector plots in Figures 20.13(b) and 20.14(b) show respectively the structural orientations of trabeculated myocardium and trabecular bone computed through skeleton dilation.

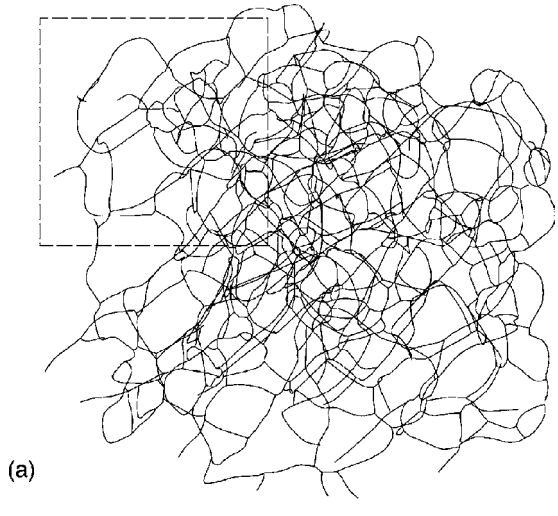



**Figure 20.14** (a) Smoothed polynomial representation of the skeleton shown in Figure 20.12; and (b) structural orientation in the region indicated by dashed lines in (a).

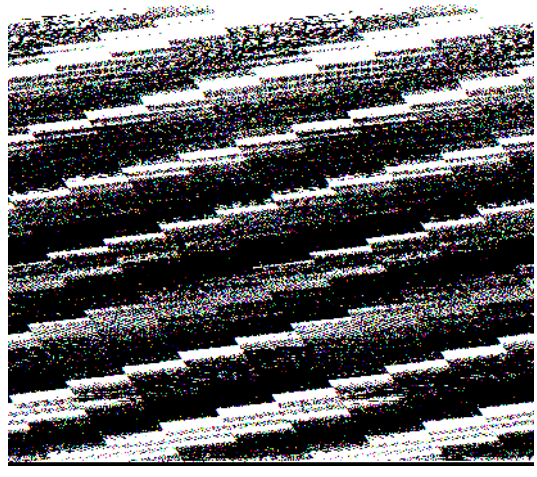


**Figure 20.14** (*continued*)

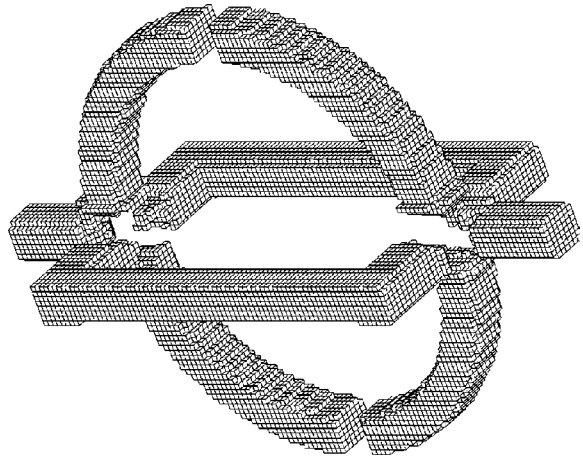


**Figure 20.15** Topological segmentation of the object shown in Figure 20.7.

Furthermore, assuming that each skeletal branch can be identified with a unique integer, we can also propagate this label to the original object set through skeleton dilation (*branch labeling*). Thus, branch formation and labeling of the skeleton can be effectively used to topologically segment the original object, as illustrated in Figure 20.15 for the object shown in Figure 20.7.

## 7. Discussion

Results indicate that the skeleton refining procedures described in the present chapter effectively reduce noise effects in terms of spurious branches and geometrical distortions. In the biomechanical modeling of trabecular tissues, these procedures allow the extraction of crucial topological and geometrical properties from 3D digital images of trabeculated tissues.

In the process of performing topological segmentation of the raw skeleton, we identify and eliminate thick junctions with a new two-phase junction classification procedure. The proposed approach associates a core voxel to each thick junction based on the highest $f_v^{26b}$ value. If, however, all voxel candidates have the same number of 26-adjacent skeleton neighbors, additional geometry-based conditions are required to determine an optimal junction [20].

In order to filter out noise branches/segments, we introduce a thickness-based branch-filtering algorithm, which does not require the definition of critical geometrical values. Results indicate that this filtering procedure can effectively detect and correct noise branches.

The proposed least squares fitting procedure incorporating global smoothing constraints yields a smooth polynomial curve for each skeletal branch built on the discrete centroidal coordinates. The procedure effectively controls numerical fluctuations induced by data nonuniformity and sparsity. Finally, we adopt a uniform skeleton dilation approach to assign the computed structural orientation from the skeleton to all non-skeletal voxels.

The complete skeleton modeling and dilation procedure is applicable to other problems in biomechanics and in image processing and intelligent recognition. For example, by labeling each object voxel with a unique branch index, we can divide the original image into separate domains based on the image topology. In biomechanical applications, this topological segmentation allows us to isolate and investigate a specific trabecula, or to filter out small trabecular columns in order to simplify the computational modeling of trabeculated tissues.

## Acknowledgments

We thank Larry Taber for continuous advice and support throughout this research. We also thank Scott Hollister for useful discussions and for providing the images of the trabecular bone. Finally, we gratefully acknowledge the support of NIH through grants R01 46367 and R01 HL64347-02.

## References

[1] Xie, W., Thompson R. and Perucchio, R. "A topology-preserving parallel 3D thinning algorithm for extracting the curve skeleton," *Pattern Recognition*, **36**, pp. 1529–1544, 2003.

[2] Chatzis, V. and Pitas, I. "Interpolation of 3-D binary images based on morphological skeletonization," *IEEE Transactions on Medical Imaging*, **19**, pp. 699–710, 2000.

[3] Hafford, K. J. and Preston, K. "Three-dimensional skeletonization of elongated solids," *Computer Vision, Graphics and Image Processing*, **27**, pp. 78–91, 1984.

[4] Leboucher, L., Irinopoulou, T. and Hazout, S. "Gray-tone skeletons of elongated objects using the concept of morphological automation: Application to images of DNA molecules," *Pattern Recognition Letters*, **15**, pp. 309–315, 1994.

[5] Ma, C. M. and Sonka, M. "A fully parallel 3D thinning algorithm and its applications," *Computer Vision and Image Understanding*, **64**, pp. 420–433, 1996.

[6] Sedmera, D., Pexieder, T., Vuillemin, M., Thompson, R. P. and Anderson, R. H. "Developmental patterning of the myocardium," *Anatomical Record*, **258**, pp. 319–337, 2000.

[7] Hollister, S. J., Brennan, J. M. and Kikuchi, N. "A homogenization sampling procedure for calculating trabecular bone effective stiffness and tissue level stress," *Journal of Biomechanics*, **27**, pp. 433–444, 1994.

[8] Taber L. A. and Perucchio R. "Modeling heart development," *Journal of Elasticity*, **61**, pp. 165–197, 2000.

[9] Xie, W. and Perucchio, R. "Multiscale finite element modeling of the trabeculated embryonic heart: Numerical evaluation of the constitutive relations for the trabeculated myocardium," *Computer Methods in Biomechanics and Biomedical Engineering*, **4**, pp. 231–248, 2001.

[10] Xie, W., Sedmera D. and Perucchio, R. "Biomechanical Modeling of the Trabeculated Embryonic Heart: Image-based Global FE Mesh Construction and Model Calibration," ASME-BED *Proceedings of 2003 Summer Bioengineering Conference*, Key Biscayne, FL, pp. 1091–1092, 2003.

[11] Malandain, G., Betrand, G. and Ayache, N. "Topological segmentation of discrete surfaces," *International Journal of Computer Vision*, **10**, pp. 183–197, 1993.

[12] Lee, T. C., Kashyap, R. L. and Chu, C. N. "Building skeleton models via 3-D medial surface/axis thinning algorithm," *Computer Vision, Graphics and Image Processing*, **56**, pp. 462–478, 1994.
[13] Kong, T. Y. and Rosenfeld, A. "Digital Topology: Introduction and survey," *Computer Vision, Graphics and Image Processing*, **48**, pp. 357–393, 1989.
[14] Saha, P. K. and Chaudhuri, B. B. "3D digital topology under binary transformation with applications," *Computer Vision and Image Understanding*, **63**, pp. 418–429, 1996.
[15] Ahmed, P., Goyal, P., Narayanan, T. S. and Suen, C. Y. "Linear time algorithms for an image labeling machine," *Pattern Recognition Letter*, **7**, pp. 273–278, 1988.
[16] Ronse, C. and Devijver, P. A. *Connected Components in Binary Images: the Detection Problem*, John Wiley & Sons, Inc., New York, 1984.
[17] Thanisch, P., McNally, B. V. and Robin, A. "Linear time algorithm for finding a picture's connected components," *Image and Vision Computing*, **2**, pp. 191–197, 1984.
[18] Arcelli, C. "Pattern thinning by contour tracing," *Computer Vision, Graphics and Image Processing*, **17**, pp. 130–144, 1981.
[19] Davies, E. R. and Plummer, A. P. N. "Thinning algorithms: A critique and a new methodology," *Pattern Recognition*, **14**, pp. 53–63, 1981.
[20] Abdullah, W. H., Saleh, A. O. M. and Morad, A. H. "A preprocessing algorithm for handwritten character recognition," *Pattern Recognition Letter*, **7**, pp. 13–18, 1988.
[21] Bribiesca, E. "A chain code for representing 3D curves," *Pattern Recognition*, **33**, pp. 755–765, 2000.
[22] Saha, P. K., Chaudhuri, B. B. and Majumder, D. D. "A New Shape Preserving Parallel Thinning Algorithm for 3D Digital Images," *Pattern Recognition*, **30**, pp. 1939–1955, 1997.
[23] Bertrand, G. "Simple points, topological numbers and geodesic neighborhoods in cubic grids," *Pattern Recognition Letter*, **15**, pp. 1003–1011, 1994.
[24] Yakowitz, S. and Szidarovszky, F. *An Introduction to Numerical Computations*, Macmillan, New York, 1989.
[25] Schumaker, L. L. *Spline Functions: Basic Theory*, John Wiley & Sons, Inc., New York, 1981.
[26] Dierckx, P. *Curve and Surface Fitting with Splines*, Oxford University Press, Oxford, 1993.
[27] Bathe, K. J. *Finite Element Procedures*, Prentice Hall, Englewood Cliffs, NJ, 1996.
[28] Hashima, A. R., Young, A. A., McCulloch, A. D. and Waldman, L. K. "Nonhomogeneous analysis of epicardial strain distribution during acute myocardium ischemia in the dog," *Journal of Biomechanics*, **26**, pp. 19–35, 1993.
[29] Press, W. H., Teukolsky, S. A., Vellerling, W. T. and Flannery, B. P. *Numerical recipes in C*, Cambridge University Press, Cambridge, UK, 1992.
[30] Borgefors, G., Nystrom, I. and Baja, G. S. "Computing skeletons in three dimensions," *Pattern Recognition*, **32**, pp. 1225–1236, 1999.
[31] Saha, P. K. and Chaudhuri, B. B. "Detection of 3-D simple points for topology preserving transformations with application to thinning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**, pp. 1028–1032, 1994.

# 21

# Applications of Clifford-valued Neural Networks to Pattern Classification and Pose Estimation

**Eduardo Bayro-Corrochano**
**Nancy Arana-Daniel**
Geovis Laboratory, Computer Science Department, CINVESTAV Centro de Investigación y de Estudios Avanzados, Apartado Postal 31-438, Plaza la Luna, Guadalajara, Jal. 44550, México

*This chapter shows the analysis and design of feed-forward neural networks and support vector machines using the coordinate-free system of Clifford or geometric algebra. It is shown that real-, complex- and quaternion-valued neural networks are simply particular cases of geometric algebra multidimensional neural networks. We design kernels for Support Multivector Machines (SMVMs) which involve the Clifford product. The conformal neuron is used for clustering data; this idea is very useful for alleviating the complexity and computational demand in classification problems. In neural computing, the preprocessing is of foremost importance, that is why we introduce a novel method of geometric preprocessing utilizing hypercomplex or Clifford moments. This method is applied together with geometric MLPs for tasks of 2D pattern classification. The experimental part illustrates the potential of geometric neural networks for a variety of real applications using multidimensional representations.*

## 1. Introduction

The literature on neurocomputing shows that there are basically two mathematical systems used in neural computing: tensor algebra [1,2] and matrix algebra [3,4]. In contrast, in this chapter we choose the coordinate-free system of Clifford, or geometric, algebra for the analysis and design of Clifford-valued feed-forward neural networks and Support Multivector Machines (SMVMs). The chapter shows that real-, complex- and quaternion-valued neural networks are merely particular cases of geometric algebra multidimensional neural networks andthat some of them can also be generated

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

using support multivector machines. In particular, the generation of RBF networks in geometric algebra is easier using the SMVM, as it allows us to find the optimal parameters automatically. In this chapter we design kernels involving the Clifford product and we show the use of the conformal neuron as preliminary data clustering. We believe that the use of SVMs in the geometric algebra framework expands their sphere of applicability for multidimensional learning.

The chapter also introduces a novel method for geometric preprocessing using generalized hypercomplex moments. The experimental part illustrates the potential of geometric neural networks for a variety of real applications using multidimensional representations. The organization of this chapter is as follows: Section 2 outlines geometric algebra. Section 3 reviews the computing principles of feed-forward neural networks, underlining their most important characteristics. Section 4 deals with the extension of the Multi-Layer Perceptron (MLP) to complex and quaternionic MLPs. Section 5 presents the generalization of feed-forward neural networks in the geometric algebra system. Section 6 describes the generalized learning rule across different geometric algebras and explains the training of geometric neural networks using genetic algorithms. Section 7 introduces support multivector machines and it explains the design of kernels involving the Clifford product, as well as the role of the conformal neuron. Section 8 introduces hypercomplex moments as a preprocessing method useful for pattern classification. Section 9 presents and compares the Clifford MLP and the real-valued MLP; interesting pattern classification problems are solved using SMVMs. The last section is dedicated to conclusions.

## 2. Geometric Algebra: An Outline

The algebras of Clifford and Grassmann are well known to pure mathematicians, but were long ago abandoned by physicists in favor of the vector algebra of Gibbs, which is indeed what is commonly used today in most areas of physics. The approach to Clifford algebra we adopt here was pioneered in the 1960s by David Hestenes [5] who has, since then, worked on developing his version of Clifford algebra–which will be referred to as *geometric algebra*–into a unifying language for mathematics and physics [6–8].

### *2.1 Basic Definitions*

Let $G_n$ denote the geometric algebra of $n$ dimensions–this is a graded linear space. As well as vector addition and scalar multiplication, we have a noncommutative product which is associative and distributive over addition–this is the *geometric* or *Clifford product*. A further distinguishing feature of the algebra is that any vector squares to give a scalar. The geometric product of two vectors $\mathbf{a}$ and $\mathbf{b}$ is written $\mathbf{ab}$ and can be expressed as a sum of its symmetric and antisymmetric parts:

$$\mathbf{ab} = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b} \tag{21.1}$$

where the inner product $\mathbf{a} \cdot \mathbf{b}$ and the outer product $\mathbf{a} \wedge \mathbf{b}$ are defined by:

$$\mathbf{a} \cdot \mathbf{b} = \frac{1}{2}(\mathbf{ab} + \mathbf{ba}) \tag{21.2}$$

$$\mathbf{a} \wedge \mathbf{b} = \frac{1}{2}(\mathbf{ab} - \mathbf{ba}) \tag{21.3}$$

The inner product of two vectors is the standard *scalar* or *dot* product and produces a scalar. The outer, or wedge product of two vectors is a new quantity which we call a *bivector*. We think of a bivector as an oriented area in the plane containing $\mathbf{a}$ and $\mathbf{b}$, formed by sweeping $\mathbf{a}$ along $\mathbf{b}$–see Figure 21.1(a).

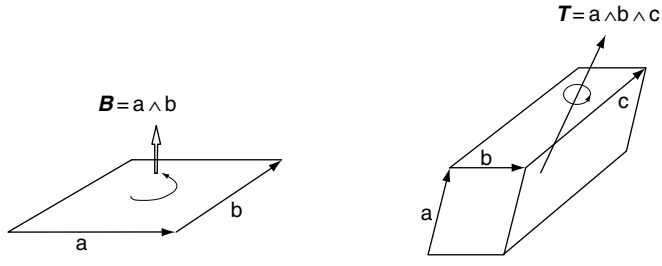



**Figure 21.1** (a) The directed area, or bivector, $\mathbf{a} \wedge \mathbf{b}$; (b) the oriented volume, or trivector, $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c}$.

Thus, $\mathbf{a} \wedge \mathbf{b}$ will have the opposite orientation, making the wedge product anti-commutative as given in Equation (21.3). The outer product is immediately generalizable to higher dimensions–for example, $(\mathbf{a} \wedge \mathbf{b}) \wedge \mathbf{c}$, a *trivector*, is interpreted as the oriented volume formed by sweeping the area $(\mathbf{a} \wedge \mathbf{b})$ along vector $\mathbf{c}$ see Figure 21.1(b). The outer product of $k$ vectors is a $k$-vector or $k$-blade, and such a quantity is said to have *grade* $k$ A multivector (a linear combination of objects of different types) is homogeneous if it contains terms of only a single grade. Geometric algebra provides a means of manipulating multivectors, which allows us to keep track of different graded objects simultaneously – much as one does with complex number operations.

In a space of three dimensions we can construct a trivector $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c}$, but no 4-vectors exist since there is no possibility of sweeping the volume element $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c}$ over a fourth dimension. The highest grade element in a space is called the pseudoscalar. The unit pseudoscalar is denoted by $\mathbf{I}$ and is crucial when discussing duality.

## 2.2 The Geometric Algebra of nD Space

In an $n$-dimensional space, we can introduce an orthonormal basis of vectors $\{\sigma_i\}, i = 1, \ldots, n$, such that $\sigma_i \cdot \sigma_j = \delta_{ij}$. This leads to a basis for the entire algebra:

$$1, \{\sigma_i\}, \{\sigma_i \wedge \sigma_j\}, \{\sigma_i \wedge \sigma_j \wedge \sigma_k\}, \ldots, \sigma_i \wedge \sigma_j \wedge \ldots \wedge \sigma_n \tag{21.4}$$

Note that the basis vectors are not represented by bold symbols. Any multivector can be expressed in terms of this basis. In this chapter we will specify a geometric algebra $G_n$ of the $n$-dimensional space by $G_{p,q,r}$, where $p, q$ and $r$ stand for the number of basis vectors which square to 1, $-1$ and 0 respectively, and fulfill $n = p + q + r$. Its even subalgebra will be denoted by $G^+_{p,q,r}$. For example, $G^+_{0,2,0}$ has the basis:

$$\{1, \sigma_1, \sigma_2, \sigma_1 \wedge \sigma_2\} \tag{21.5}$$

where $\sigma_1^2 = -1, \sigma_2^2 = -1$. This means $p = 0, q = 2$ and $r = 0$. Thus, the dimension of this geometric algebra is $n = p + q + r = 2$.

In the $n$D space, there are multivectors of grade 0 (scalars), grade 1 (vectors), grade 2 (bivectors), grade 3 (trivectors), etc… up to grade $n$. Any two such multivectors can be multiplied using the geometric product. Consider two multivectors $\boldsymbol{A}_r$ and $\boldsymbol{B}_s$ of grades $r$ and $s$ respectively. The geometric product of $\boldsymbol{A}_r$ and $\boldsymbol{B}_s$ can be written as:

$$\boldsymbol{A}_r \boldsymbol{B}_s = \langle \boldsymbol{AB} \rangle_{r+s} + \langle \boldsymbol{AB} \rangle_{r+s-2} + \cdots + \langle \boldsymbol{AB} \rangle_{|r-s|} \tag{21.6}$$

where $\langle \boldsymbol{M} \rangle_t$ is used to denote the $t$-grade part of multivector $\boldsymbol{M}$, e.g. consider the geometric product of two vectors $\mathbf{ab} = \langle \mathbf{ab} \rangle_0 + \langle \mathbf{ab} \rangle_2 = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b}$.

## 2.3 The Geometric Algebra of 3D Space

The basis for the geometric algebra $G_{3,0,0}$ of the 3D space has $2^3 = 8$ elements and is given by:

$$\underbrace{1}_{\text{scalar}} \underbrace{\{\sigma_1, \sigma_2, \sigma_3\}}_{\text{vectors}}, \underbrace{\{\sigma_1\sigma_2, \sigma_2\sigma_3, \sigma_3\sigma_1\}}_{\text{bivectors}}, \underbrace{\{\sigma_1\sigma_2\sigma_3\} \equiv \mathbf{I}}_{\text{trivector}} \tag{21.7}$$

Since the basis vectors are orthogonal, i.e. $\sigma_1\sigma_2 = \sigma_1 \cdot \sigma_2 + \sigma_1\sigma_2 = \sigma_1 \wedge \sigma_2$, we write simply $\sigma_1\sigma_2$.

It can easily be verified that the trivector or pseudoscalar $\sigma_1\sigma_2\sigma_3$ squares to 1 and commutes with all multivectors in the 3D space. We therefore give it the symbol $\mathbf{i}$; noting that this is not the uninterpreted commutative scalar imaginary j used in quantum mechanics and engineering.

## 2.4 Rotors

Multiplication of the three basis vectors $\sigma_1, \sigma_2$ and $\sigma_3$ by $\boldsymbol{I}$ results in the three basis bivectors $\sigma_1\sigma_2 = \boldsymbol{I}\sigma_3$, $\sigma_2\sigma_3 = \boldsymbol{I}\sigma_1$ and $\sigma_3\sigma_1 = \boldsymbol{I}\sigma_2$. These simple bivectors rotate vectors in their own plane by 90°, e.g. $(\sigma_1\sigma_2)\sigma_2 = \sigma_1$, $(\sigma_2\sigma_3)\sigma_2 = -\sigma_3$, etc. Identifying the $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ of the quaternion algebra with $\boldsymbol{I}\sigma_1, -\boldsymbol{I}\sigma_2, \boldsymbol{I}\sigma_3$, the famous Hamilton relations $\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$ can be recovered. Since the $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are bivectors, it comes as no surprise that they represent 90° rotations in orthogonal directions and provide a well-suited system for the representation of general 3D rotations, see Figure 21.2.

In geometric algebra, a rotor (short name for rotator), $\boldsymbol{R}$, is an even-grade element of the algebra which satisfies $\boldsymbol{R}\tilde{\boldsymbol{R}} = 1$, where $\tilde{\boldsymbol{R}}$ stands for the conjugate of $\boldsymbol{R}$. If $\boldsymbol{A} = \{a_0, a_1, a_2, a_3\} \in G_{3,0,0}$ represents a unit quaternion, then the rotor which performs the same rotation is simply given by:

$$\boldsymbol{R} = \underbrace{a_0}_{\text{scalar}} + \underbrace{a_1(\boldsymbol{I}\sigma_1) - a_2(\boldsymbol{I}\sigma_2) + a_3(\boldsymbol{I}\sigma_3)}_{\text{bivectors}} = a_0 + a_1\sigma_2\sigma_3 + a_2\sigma_3\sigma_1 + a_3\sigma_1\sigma_2 = a_0 + \mathbf{a} \tag{21.8}$$

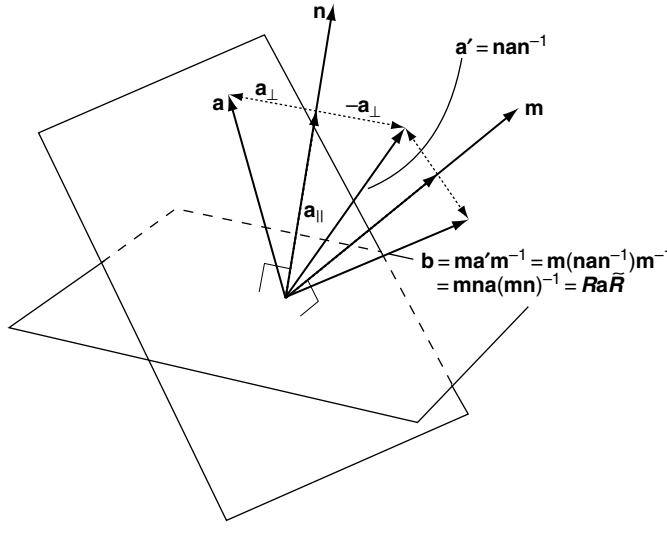



**Figure 21.2** The rotor in the 3D space formed by a pair of reflections.

The quaternion algebra is therefore seen to be a subset of the geometric algebra of 3D space. The conjugate of a rotor is given by:

$$\tilde{\boldsymbol{R}} = a_0 - a_1\sigma_2\sigma_3 + a_2\sigma_3\sigma_1 + a_3\sigma_1\sigma_2 = a_0 - \mathbf{a} \tag{21.9}$$

A rotation can be performed by a pair of reflections, see Figure 21.2. It can easily be shown that the result of reflecting a vector $\mathbf{a}$ in the plane perpendicular to a unit vector $\mathbf{n}$ is $\mathbf{a}_\perp - \mathbf{a}_{||} = \mathbf{a}' = -\mathbf{nan}^{-1}$, where $\mathbf{a}_\perp$ and $\mathbf{a}_{||}$ respectively denote projections of $\mathbf{a}$ perpendicular and parallel to $\mathbf{n}$. Thus, a reflection of $\mathbf{a}$ in the plane perpendicular to $\mathbf{n}$, followed by a reflection in the plane perpendicular to another unit vector $\mathbf{m}$ results in a new vector $\mathbf{b} = -\mathbf{m}(-\mathbf{nan}^{-1})\mathbf{m}^{-1} = (\mathbf{mn})\mathbf{a}(\mathbf{mn})^{-1} = \boldsymbol{R}\mathbf{a}\tilde{\boldsymbol{R}}$. Using the geometric product we can show that the rotor $\boldsymbol{R}$ of Equation (21.8) is a multivector consisting of both a scalar part and a bivector part, i.e. $\boldsymbol{R} = \mathbf{mn} = \mathbf{m}\cdot\mathbf{n} + \mathbf{m}\wedge\mathbf{n}$. These components correspond to the scalar and vector parts of an equivalent unit quaternion in $G_{3,0,0}$. Considering the scalar and the bivector parts, we can further write the Euler representation of a rotor as follows:

$$\boldsymbol{R} = e^{n\frac{\theta}{2}} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2} \tag{21.10}$$

where the rotation axis $\boldsymbol{n} = n_1\sigma_2\sigma_3 + n_2\sigma_3\sigma_1 + n_3\sigma_1\sigma_2$ is spanned by the bivector basis. The transformation in terms of a rotor $\mathbf{a} \mapsto \boldsymbol{R}\mathbf{a}\tilde{\boldsymbol{R}} = \mathbf{b}$ is a very general way of handling rotations; it works for multivectors of any grade and in spaces of any dimension, in contrast to quaternion calculus. Rotors combine in a straightforward manner, i.e. a rotor $\boldsymbol{R}_1$ followed by a rotor $\boldsymbol{R}_2$ is equivalent to a total rotor $\boldsymbol{R}$, where $\boldsymbol{R} = \boldsymbol{R}_1\boldsymbol{R}_2$.

## 2.5 Conformal Geometric Algebra

In order to explain the conformal geometric algebra we introduce the Minkowski plane, $\Re^{1,1}$, which has an orthonormal basis $\{e_+, e_-\}$ with the property $e_+^2 = +1$, $e_-^2 = -1$. Using this basis two extra null bases can be generated $e_\infty = (e_+ + e_-)$ and $e_0 = \frac{1}{2}(e_+ + e_-)$, so that $e_\infty^2 = e_0^2 = 0$ and $e_0 \cdot e_\infty = 1$.

Given the Euclidean space $\Re^n$, the conformal space is obtained via $\Re^{n+1,1} = \Re^n \otimes \Re^{1,1}$. Its associated conformal geometric algebra is $G_{n+1,1}$. Any vector $\mathbf{x} \in \Re^n$ can be embedded in the conformal space using the following mapping:

$$\begin{aligned}\mathbf{x} = \mathrm{F}(\mathbf{x}) &= -(\mathbf{x} - e_+)\boldsymbol{e}_\infty(\mathbf{x} - e_+) \\ &= \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0\end{aligned} \tag{21.11}$$

so that $\boldsymbol{x}^2 = 0$. A null vector, like $\boldsymbol{x}$, whose component $e_0$ is unity, is called a normalized vector. Given the normalized null vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, we can compute a Euclidean metric:

$$\boldsymbol{x}\cdot\boldsymbol{y} = -\frac{1}{2}(\mathbf{x} - \mathbf{y})^2 \tag{21.12}$$

which corresponds to the length of a cord joining two points lying on the null cone, see [8] for more details.

### 2.5.1 Lines, Planes and Spheres of the 3D Euclidean Space

Lines, planes and hyperplanes are represented in the conformal space by wedging points of the conformal space with the point at infinity:

$$\begin{aligned}\boldsymbol{l} &= e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \\ \phi &= e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \\ \boldsymbol{h}_n &= e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \wedge \ldots \wedge \boldsymbol{x}_n\end{aligned} \tag{21.13}$$

The equation of a hypersphere can be formulated considering the hypersphere centered at point $\mathbf{x} \in \Re^n$ with radius $\rho^2 = (\boldsymbol{x} - \boldsymbol{p})^2$, here $\boldsymbol{p}$ is any point lying on the sphere. However, we can express this using Equation (21.12) in terms of homogeneous points:

$$\boldsymbol{x} \cdot \boldsymbol{p} = -\frac{1}{2}\rho^2 \tag{21.14}$$

Using $\boldsymbol{x} \cdot e = 1$, we can simplify this equation to $\boldsymbol{x} \cdot \boldsymbol{s} = 0$, where

$$\boldsymbol{s} = \boldsymbol{p} - \frac{1}{2}\rho^2 e_\infty = \boldsymbol{p} + \frac{1}{2}\left(\boldsymbol{p}^2 - \rho^2\right) e_\infty + e_0 \tag{21.15}$$

The vector $\boldsymbol{s}$ has the properties $\boldsymbol{s}^2 = \rho^2$ and $e \cdot \boldsymbol{s} = -1$. Note that if $\rho = 0$, Equation (21.15) becomes the equation of a point (Equation 21.10). The same result can be obtained by first computing the volume of

$$\tilde{\boldsymbol{s}} = e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \ldots \wedge \boldsymbol{x}_{n+1} \tag{21.16}$$

and then taking the dual of the latter with the pseudoscalar $\boldsymbol{I} = \sigma_1\sigma_2\sigma_3 e_1 e_2$

$$\boldsymbol{s} = \tilde{\boldsymbol{s}} \cdot \boldsymbol{I} \tag{21.17}$$

Let us give as illustration the computation of a plane and a sphere of $\Re^3$, given four points in general positions:

$$\begin{aligned} \phi &= e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \\ \boldsymbol{s} &= \tilde{\boldsymbol{s}} \cdot \boldsymbol{I} = (e_\infty \wedge \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \wedge \boldsymbol{x}_4) \cdot \boldsymbol{I} = \boldsymbol{p} - \frac{1}{2}\rho^2 e_\infty \end{aligned} \tag{21.18}$$

## 3. Real-valued Neural Networks

The approximation of nonlinear mappings using neural networks is useful in various aspects of signal processing, such as in pattern classification, prediction, system modeling and identification. This section reviews the fundamentals of standard real-valued feed-forward architectures.

Cybenko [4] used, for the approximation of a continuous function, $g(\mathbf{x})$, the superposition of weighted functions:

$$g(\mathbf{x}) = \sum_{j=1}^{N} w_j \sigma_j \left(\mathbf{w}_j^{\mathrm{T}} \mathbf{x} + \theta_j\right) \tag{21.19}$$

where $\sigma(\cdot)$ is a continuous discriminatory function like a sigmoid, $w_j \in \Re$ and $\mathbf{x}, \theta_j, \mathbf{w}_j \in \Re^{\boldsymbol{n}}$. Finite sums of the form of Equation (21.19) are dense in $C^0(\boldsymbol{I_n})$, if $|g_k(\mathbf{x}) - y_k(\mathbf{x})| < \varepsilon$ for a given $\varepsilon > 0$ and all $\mathbf{x} \in [0, 1]^{\boldsymbol{n}}$. This is called the *density theorem* and is a fundamental concept in approximation theory and nonlinear system modeling [4,9].

A structure with $k$ outputs $y_k$, having several layers using logistic functions, is known as a *Multilayer Perceptron* (MLP) [10]. The output of any neuron of a hidden layer or of the output layer can be represented in a similar way,

$$o_j = f_j\left(\sum_{i=1}^{N_i} w_{ji} x_{ji} + \theta_j\right) \quad y_k = f_k\left(\sum_{j=1}^{N_j} w_{kj} o_{kj} + \theta_k\right) \tag{21.20}$$

where $f_j(\cdot)$ is logistic and $f_k(\cdot)$ is logistic or linear. Linear functions at the outputs are often used for pattern classification. In some tasks of pattern classification, a hidden layer is necessary, whereas in some tasks of automatic control, two hidden layers may be required. Hornik [9] showed that standard multilayer feed-forward networks are able to accurately approximate any measurable function to a desired degree. Thus, they can be seen as *universal approximators*. In the case of a training failure, we should attribute any error to inadequate learning, an incorrect number of hidden neurons, or a poorly defined deterministic relationship between the input and output patterns.

Poggio and Girosi [11] developed the *Radial Basis Function* (RBF) network, which consists of a superposition of weighted Gaussian functions,

$$y_j(\mathbf{x}) = \sum w_{ji} G_i\left(\mathbf{D}_i\left(\mathbf{x} - \mathbf{t}_i\right)\right) \tag{21.21}$$

where $y_j$ is the $j$-output, $w_{ij} \in \Re$, $G_i$ is a Gaussian function, $\mathbf{D}_i$ an $N \times N$ dilatation diagonal matrix, and $\mathbf{x}, \mathbf{t}_i \in \Re^n$. The vector $\mathbf{t}_i$ is a translation vector. This architecture is supported by the regularization theory.

## 4. Complex MLP and Quaternionic MLP

An MLP is defined to be in the complex domain when its weights, activation function and outputs are complex-valued. The selection of the activation function is not a trivial matter. For example, the extension of the sigmoid function from $\Re$ to $C$,

$$\boldsymbol{f}(z) = \frac{1}{1 + e^{-z}} \tag{21.22}$$

where $\boldsymbol{z} \in C$, is not allowed, because this function is analytic and unbounded [12]; this is also true for the functions $\tanh(\boldsymbol{z})$ and $e^{-z^2}$. We believe these kinds of activation function exhibit problems with convergence in training due to their singularities. The necessary conditions that a complex activation $\boldsymbol{f}(\boldsymbol{z}) = a(x, y) + \mathrm{i}b(x, y)$ has to fulfill are: $\boldsymbol{f}(\boldsymbol{z})$ must be nonlinear in $x$ and $y$, the partial derivatives $a_x, a_y, b_x$ and $b_y$ must exist, $a_x b_y \neq b_x a_y$, and $\boldsymbol{f}(\boldsymbol{z})$ must not be entire. Accordingly, Georgiou and Koutsougeras [12] proposed the formulation:

$$\boldsymbol{f}(\boldsymbol{z}) = \frac{\boldsymbol{z}}{c + \dfrac{1}{r}|\boldsymbol{z}|} \tag{21.23}$$

where $c, r \in \Re^+$. These authors thus extended the traditional real-valued back-propagation learning rule to the complex-valued rule of the *Complex Multilayer Perceptron* (CMLP).

Arena *et al.* [13] introduced the *Quaternionic Multilayer Perceptron* (QMLP), which is an extension of the CMLP. The weights, activation functions and outputs of this net are represented in terms of quaternions [14]. Arena *et al.* chose the following nonanalytic bounded function:

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{q}) &= \boldsymbol{f}(q_0 + q_{1i} + q_{2j} + q_{3k}) \\ &= \left(\frac{1}{1 + e^{-q_0}}\right) + \left(\frac{1}{1 + e^{-q_1}}\right) i + \left(\frac{1}{1 + e^{-q_2}}\right) j + \left(\frac{1}{1 + e^{-q_3}}\right) k \end{aligned} \tag{21.24}$$

where $\boldsymbol{f}(\cdot)$ is now the function for quaternions. These authors proved that the superposition of such functions accurately approximates any continuous quaternionic function defined in the unit polydisc of $C^n$. The extension of the training rule to the CMLP was demonstrated in [13].

## 5. Clifford-valued Feed-forward Neural Networks

Real, complex and quaternionic neural networks can be further generalized within the geometric algebra framework, in which the weights, the activation functions and the outputs are now represented using multivectors. For the real-valued neural networks discussed in Section 3, the vectors are multiplied with the weights, using the scalar product. For geometric neural networks, the scalar product is replaced by the geometric product.

### *5.1 The Activation Function*

The activation function of Equation (21.23), used for the CMLP, was extended by Pearson and Bisset [15] for a type of Clifford MLP by applying different Clifford algebras, including quaternion algebra. We propose here an activation function that will affect each multivector basis element. This function was introduced independently by the authors [16] and is in fact a generalization of the function of Arena *et al.* [13]. The function for an $n$-dimensional multivector $\boldsymbol{m}$ is given by:

$$\begin{aligned}\boldsymbol{f}(\boldsymbol{m}) &= \boldsymbol{f}(m_0 + m_i\sigma_i + m_j\sigma_j + m_k\sigma_k + \cdots + m_{ij}\sigma_i \wedge \sigma_j + \cdots + \\ &\quad + m_{ijk}\sigma_i \wedge \sigma_j \wedge \sigma_k + \cdots + m_n\sigma_1 \wedge \sigma_2 \wedge \cdots \wedge \sigma_n) \\ &= (m_0) + f(m_i)\,\sigma_i + f\left(m_j\right)\sigma_j + f(m_k)\,\sigma_k + \cdots + f\left(m_{ij}\right)\sigma_i \wedge \sigma_j + \cdots + \\ &\quad + f\left(m_{ijk}\right)\sigma_i \wedge \sigma_j \wedge \sigma_k + \cdots + f(m_n)\,\sigma_1 \wedge \sigma_2 \wedge \cdots \wedge \sigma_n \end{aligned} \tag{21.25}$$

where $\boldsymbol{f}(\cdot)$ is written in bold to distinguish it from the notation used for a single-argument function $f(\cdot)$. The values of $f(\cdot)$ can be of the sigmoid or Gaussian type.

### *5.2 The Geometric Neuron*

The *McCulloch–Pitts neuron* uses the scalar product of the input vector and its weight vector [10]. The extension of this model to the *geometric neuron* requires the substitution of the scalar product with the Clifford or geometric product, i.e.

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} + \theta \Rightarrow wx + \theta = w \cdot x + w \wedge x + \theta \tag{21.26}$$

Figure 21.3 shows in detail the McCulloch–Pitts neuron and the geometric neuron. This figure also depicts how the input pattern is formatted in a specific geometric algebra. The geometric neuron outputs a richer kind of pattern. We can illustrate this with an example in $G_{3,0,0}$

$$\begin{aligned} o &= \boldsymbol{f}(wx + \theta) \\ &= \boldsymbol{f}(s_0 + s_1\sigma_1 + s_2\sigma_2 + s_3\sigma_3 + s_4\sigma_1\sigma_2 + s_5\sigma_1\sigma_3 + s_6\sigma_2\sigma_3 + s_7\sigma_1\sigma_2\sigma_3) \\ &= \boldsymbol{f}(s_0) + \boldsymbol{f}(s_1)\sigma_1 + \boldsymbol{f}(s_2)\sigma_2 + \boldsymbol{f}(s_3)\sigma_3 + \boldsymbol{f}(s_4)\sigma_1\sigma_2 + \cdots + \\ &\quad + \boldsymbol{f}(s_5)\sigma_1\sigma_3 + \boldsymbol{f}(s_6)\sigma_2\sigma_3 + \boldsymbol{f}(s_7)\sigma_1\sigma_2\sigma_3) \end{aligned} \tag{21.27}$$

where $\boldsymbol{f}$ is the activation function defined in Equation (21.25), and $s_i \in \Re$. If we use the McCulloch–Pitts neuron in the real-valued neural network, the output is simply the scalar given by:

$$o = f\left(\sum_i^N w_i x_i + \theta\right) \tag{21.28}$$

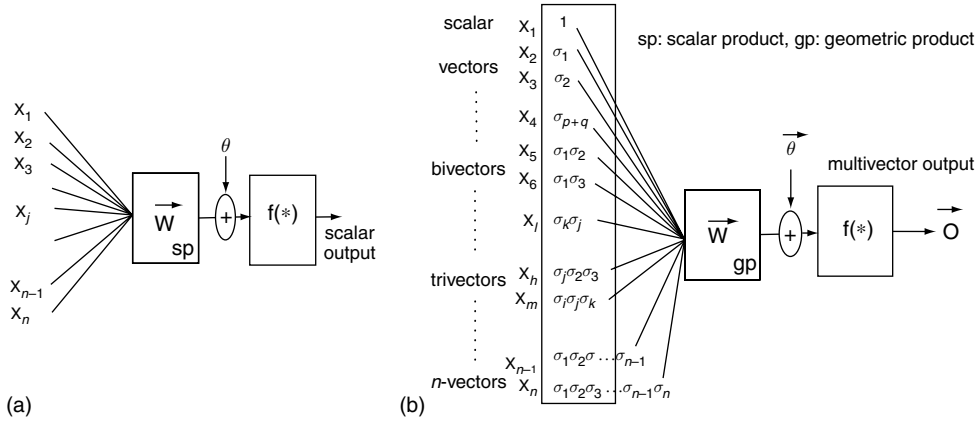



**Figure 21.3** (a) McCulloch–Pitts neuron; and (b) geometric neuron.

The geometric neuron outputs a signal with more geometric information:

$$\boldsymbol{o} = \boldsymbol{f}(\boldsymbol{wx} + \theta) = \boldsymbol{f}(\boldsymbol{w} \cdot \boldsymbol{x} + \boldsymbol{w} \wedge \boldsymbol{x} + \theta) \tag{21.29}$$

It has both a scalar product like the McCulloch–Pitts neuron,

$$\boldsymbol{f}(\boldsymbol{w} \cdot \boldsymbol{x} + \theta) = f(s_0) \equiv \left( \sum_i^N w_i x_i + \theta \right) \tag{21.30}$$

and also the outer product given by:

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{w} \wedge \boldsymbol{x} + \theta - \theta) = {} & f(s_1)\sigma_1 + f(s_2)\sigma_2 + f(s_3)\sigma_3 + f(s_4)\sigma_1\sigma_2 + \cdots + \\ & + f(s_5)\sigma_1\sigma_3 + f(s_6)\sigma_2\sigma_3 + f(s_7)\sigma_1\sigma_2\sigma_3) \end{aligned} \tag{21.31}$$

Note that the outer product gives the scalar cross-products between the individual components of the vector, which are nothing more than the multivector components of points or lines (vectors), planes (bivectors) and volumes (trivectors). This characteristic can be used for the implementation of geometric preprocessing in the extended geometric neural network. To a certain extent, this kind of neural network resembles the higher order neural networks of [17]. However, an extended geometric neural network uses not only a scalar product of higher order, but also all the necessary scalar cross-products for carrying out a *geometric cross-correlation*.

In conclusion, a geometric neuron can be seen as a kind of *geometric correlation operator*, which, in contrast to the McCulloch–Pitts neuron, offers not only points but higher grade multivectors such as planes, volumes, . . . , hypervolumes for interpolation.

## *5.3 Feed-forward Clifford-valued Neural Networks*

Figure 21.4 depicts standard neural network structures for function approximation in the geometric algebra framework. Here, the inner vector product has been extended to the geometric product and the activation functions are according to Equation (21.25).

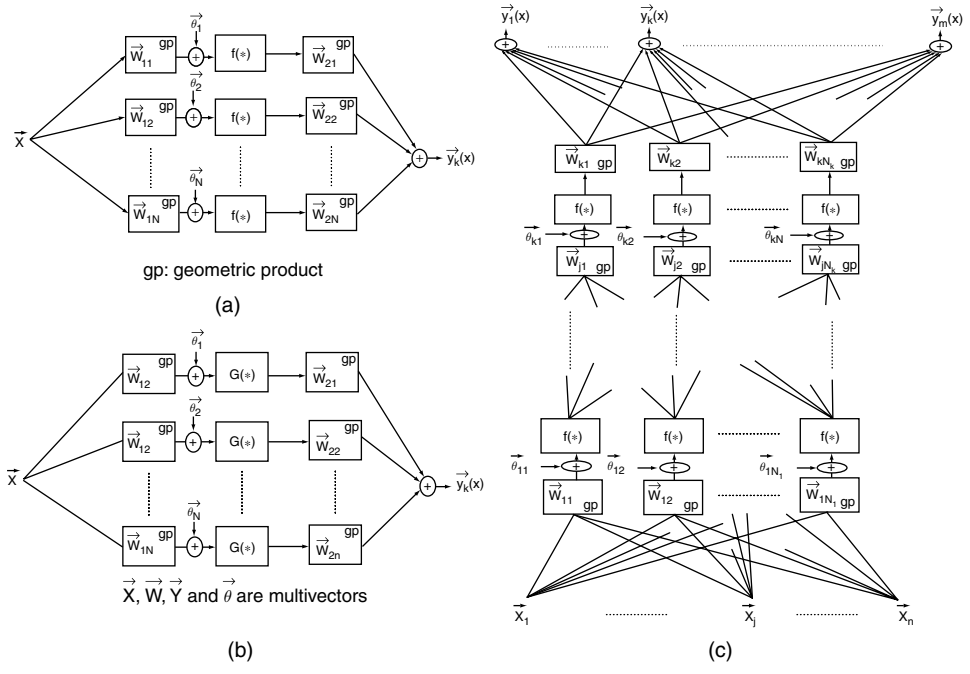



**Figure 21.4** Geometric network structures for approximation. (a) Cybenko's; (b) GRBF network; (c) $GMLP_{p,q,r}$.

Equation (21.19) of Cybenko's model in geometric algebra is:

$$\boldsymbol{y}(\boldsymbol{x}) = \sum_{j=1}^{N} \boldsymbol{w}_j \boldsymbol{f}(\boldsymbol{w}_j \cdot \boldsymbol{x} + \boldsymbol{w}_j \wedge \boldsymbol{x} + \theta_j) \tag{21.32}$$

The extension of the MLP is straightforward. The equations using the geometric product for the outputs of hidden and output layers are given by:

$$\begin{aligned} \boldsymbol{o}_j &= \boldsymbol{f}\left(\sum_{i=1}^{N_i} \boldsymbol{w}_{ji} \cdot \boldsymbol{x}_{ji} + \boldsymbol{w}_{ji} \wedge \boldsymbol{x}_{ji} + \theta_j\right) \\ \boldsymbol{y}_k &= \boldsymbol{f}_k\left(\sum_{j=1}^{N_j} \boldsymbol{w}_{kj} \cdot \boldsymbol{o}_{kj} + \boldsymbol{w}_{kj} \wedge \boldsymbol{o}_{kj} + \theta_k\right) \end{aligned} \tag{21.33}$$

In radial basis function networks, the dilatation operation, given by the diagonal matrix $\mathbf{D}_i$, can be implemented by means of the geometric product with a dilation $\mathbf{D}_i = e^{\alpha \frac{i\tilde{i}}{2}}$ [7], i.e.,

$$\mathbf{D_i}\,(\mathbf{x} - \mathbf{t}_i) \Rightarrow \mathbf{D}_i\,(\mathbf{x} - \mathbf{t}_i)\,\tilde{\mathbf{D}}_i \tag{21.34}$$

$$\boldsymbol{y}_k(\boldsymbol{x}) = \sum_{j=1}^{N} \boldsymbol{w}_{kj} \boldsymbol{G}_j\left(\mathbf{D}_j\left(\mathbf{x}_{ji} - \mathbf{t}_j\right)\tilde{\mathbf{D}}_j\right) \tag{21.35}$$

Note that in the case of the geometric RBF we are also using an activation function according to Equation (21.25).

## 6. Learning Rule

This section demonstrates the multidimensional generalization of the gradient descent learning rule in geometric algebra. This rule can be used for training the Geometric MLP (GMLP) and for tuning the weights of the Geometric RBF (GRBF). Previous learning rules for the real-valued MLP, complex MLP [12], and quaternionic MLP [13] are special cases of this extended rule.

### *6.1 Multidimensional Back-propagation Training Rule*

The *norm of a multivector* $\boldsymbol{x}$ for the learning rule is given by:

$$|\boldsymbol{x}| = (\boldsymbol{x}|\boldsymbol{x})^{\frac{1}{2}} = \left(\sum_{A} [\boldsymbol{x}]_A^2\right)^{\frac{1}{2}} \tag{21.36}$$

The geometric neural network with $n$ inputs and $m$ outputs approximates the target mapping function:

$$y_t : \left(G_{p,q,r}\right)^n \to \left(G_{p,q,r}\right)^m \tag{21.37}$$

where $\left(G_{p,q,r}\right)^n$ is the $n$-dimensional module over the geometric algebra $G_{p,q,r}$ [15]. The error at the output of the net is measured according to the metric:

$$\mathrm{E} = \frac{1}{2}\int_{\boldsymbol{x}\in\boldsymbol{X}} \|y_{\boldsymbol{w}} - y_{\boldsymbol{t}}\|^2 \tag{21.38}$$

where $\boldsymbol{X}$ is some compact subset of the Clifford module $\left(G_{p,q,r}\right)^n$ involving the product topology derived from Equation (21.36) for the norm, and where $y_w$ and $y_t$ are the learned and target mapping functions, respectively. The *back-propagation algorithm* [10] is a procedure for updating the weights and biases. This algorithm is a function of the negative derivative of the error function (Equation (21.38)) with respect to the weights and biases themselves. The computing of this procedure is straightforward, and here we will only give the main results. The updating equation for the multivector weights of any hidden $j$-layer is

$$\boldsymbol{w_{ij}}(t+1) = \eta\left[\left(\sum_{k}^{N_k} \delta_{kj} \otimes \overline{\boldsymbol{w_{kj}}}\right) \circ \boldsymbol{F}'\left(\boldsymbol{net_{ij}}\right)\right] \otimes \overline{\boldsymbol{O_i}} + \alpha \boldsymbol{w_{ij}}(t) \tag{21.39}$$

for any $k$-output with a nonlinear activation function

$$\boldsymbol{w_{jk}}(t+1) = \eta\lfloor(\boldsymbol{y_{kt}} - \boldsymbol{y_{ka}}) \circ \boldsymbol{F}'\left(\boldsymbol{net_{jk}}\right)\rfloor \otimes \overline{\boldsymbol{o}_j} + \alpha \boldsymbol{w_{jk}}(t) \tag{21.40}$$

and for any $k$-output with a linear activation function

$$\boldsymbol{w_{jk}}(t+1) = \eta\,(\boldsymbol{y_{kt}} - \boldsymbol{y_{ka}}) \otimes \overline{\boldsymbol{o}_j} + \alpha \boldsymbol{w_{jk}}(t) \tag{21.41}$$

In the above equations, $\boldsymbol{F}$ is the activation function defined in Equation (21.25), $t$ is the update step, $\eta$ and $\alpha$ are the *learning rate* and the momentum, respectively, $\otimes$ is the Clifford or geometric product, $\circ$ is the scalar product, and $\overline{(\cdot)}$ is the *multivector anti-involution* (reversion or conjugation).

In the case of the non-Euclidean $\boldsymbol{x} \in G_{0,3,0}$, $\overline{(\cdot)}$ corresponds to the simple conjugation. Each neuron now consists of $p+q+r$ units, each for a multivector component. The biases are also multivectors and are absorbed as usual in the sum of the activation signal, here defined as $\boldsymbol{net}_{ij}$. In the learning rules, Equations (21.39)–(21.41), the computation of the geometric product and the anti-involution varies depending on the geometric algebra being used [18]. To illustrate this, the conjugation required in the learning rule for quaternion algebra is $\bar{\mathbf{x}} = x_0 - x_1\sigma_1 - x_2\sigma_2 - x_3\sigma_1\sigma_2$, where $\boldsymbol{x} \in G_{0,2,0}$.

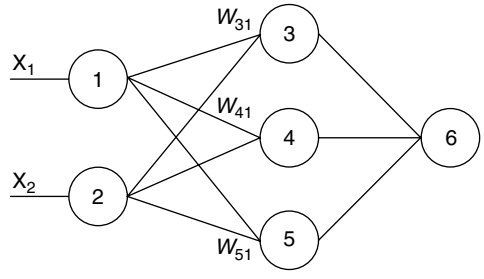



**Figure 21.5** Geometric MLP for training using genetic algorithms. The weights are Clifford numbers.

## *6.2 Geometric Learning Using Genetic Algorithms*

This learning method utilizes a standard genetic algorithm (see [19]). All Clifford weights and biases of the geometric MLP depicted in Figure 21.5 are encoded as a genotype. Note that the weights are Clifford numbers which have a length of $2^n$, according to the total dimension $n$ of the Clifford algebra $G_n$. The used objective function is

$$E(\boldsymbol{w}_{31}, \boldsymbol{w}_{32}, \ldots, \boldsymbol{w}_{13}) = \frac{1}{2} \sum_{i=1}^{n} \|\boldsymbol{o}_i - \hat{\boldsymbol{o}}_i\| \tag{21.42}$$

where $\boldsymbol{w}_{ij}$ stands for the Clifford weights and $\hat{\boldsymbol{o}}_i$ and $\boldsymbol{o}_i$ for the actual and target outputs respectively. As usual, in each iteration the standard genetic operations (mutation, crossover) are carried out and the fitness of the population is evaluated. The procedure carries on and on until little progress is noticed. The qualified genotype guarantees a good performance of the neural net. Interestingly enough, this procedure takes into account the involved Clifford product.

We trained a geometric MLP using the genetic algorithm for the problem of pattern classification. For this case, we utilized as preprocessing the hypercomplex moments. As expected, this kind of training proved not to be trapped in a local minimum. This experiment is discussed in detail in Section 9.

# 7. Support Vector Machines in the Geometric Algebra Framework

The *Support Vector Machine* (SVM) approach of Vladimir N. Vapnik [20] applies optimization methods for learning. Using SVMs we can generate a type of two-layer network and RBF networks, as well as networks with other kernels. Our idea is to generate neural networks by using SVMs in conjunction with geometric algebra, and thereby in the neural processing of multivectors. We will call our approach the *Support Multivector Machine* (SMVM). We shall briefly review SVMs and then explain the SMVM.

## *7.1 Support Vector Machines*

The SVM maps the input space $\Re^d$ into a high-dimensional *feature space* $H$, given by $\Phi : \Re^d \Rightarrow H$, satisfying a kernel $K(\mathbf{x}; \mathbf{x}_i) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, which fulfills *Mercer's condition* [20]. The SVM constructs an optimal hyperplane in the feature space which divides the data into two clusters.

SVMs build the following mapping:

$$f(\mathbf{x}) = \text{sign}\left( \sum_{\text{supportvectors}} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) - b \right) \tag{21.43}$$

The coefficients $\alpha_i$ in the separable case (and analogously in the nonseparable case), are found by maximizing the functional based on Lagrange coefficients:

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j K\left(\mathbf{x}_i, \mathbf{x}_j\right) \tag{21.44}$$

subject to the constraints $\sum_{i=1}^{l} \alpha_i y_i = 0$, where $\alpha_i \geq 0, i = 1, 2, \ldots, l$. This functional coincides with the functional for finding the optimal hyperplane. Examples of SVMs include:

$$K(\mathbf{x}; \mathbf{x}_i) = [(\mathbf{x} \cdot \mathbf{x}_i) + 1]^d \quad (\textit{polynomial learning machines}) \tag{21.45}$$

$$K_y\left(|\mathbf{x} - \mathbf{x}_i|\right) = \exp\{-y|\mathbf{x} - \mathbf{x}_i|^2\} \quad (\textit{radial basis function machines}) \tag{21.46}$$

$$K(\mathbf{x}, \mathbf{x}_i) = S\left(v\left(\mathbf{x} \cdot \mathbf{x}_i\right) + c\right) \quad (\textit{two-layer neural networks}) \tag{21.47}$$

## 7.2 Support Multivector Machines

A *Support Multivector Machine* (SMVM) maps the multivector input space $\Re^{2n}$ of $G_{p,q,r}(n = p+q+r)$ into a high-dimensional feature space $\Re^{n_f}\left(n_f \geq 2^n\right)$

$$\Phi : \Re^{2^n} \Rightarrow \Re^{\boldsymbol{n}_f} \tag{21.48}$$

The SMVM constructs an optimal separating hyperplane in the multivector *feature space* by using, in the nonlinear case, the kernels:

$$\sum_{m=1}^{N\text{graded spaces}} K_m\left(\boldsymbol{x}_{mi}, \boldsymbol{x}_{mj}\right) = \sum_{m=1}^{N\text{graded spaces}} \Phi\left(x_{mi}\right) \cdot \Phi\left(\boldsymbol{x}_{mj}\right) \tag{21.49}$$

which fulfill Mercer's conditions. SMVMs for multivectors $\boldsymbol{x}, \boldsymbol{y}$ of a geometric algebra $G_{p,q,r}$ are implemented by the multivector mapping:

$$\begin{aligned} \{y_1, y_2, \ldots, y_m\} &= \sum_{\boldsymbol{m}=1}^{N\text{graded spaces}} f_m\left(\boldsymbol{x}_{mi}\right) \\ &= \sum_{m=1}^{N\text{graded spaces}} \operatorname{sign}\left(\sum_{\text{support vectors}} y_{mi} \alpha_{mi} K\left(\boldsymbol{x}_{mi}, \boldsymbol{x}_m\right) - b_m\right) \end{aligned} \tag{21.50}$$

where $m$ denotes the grade of the spaces and $y_m$ the magnitude of the $m$-component ($m$-blade) of the multivector $\boldsymbol{y}$. The coefficients $\alpha_{mi}$ in the separable case (and analogously in the nonseparable case), are found by maximizing the functional based on Lagrange coefficients:

$$\begin{aligned} W\left(\alpha_i^m\right) &= W\left(\sum_{m=1}^{N\text{graded spaces}} \sum_{i}^{l} \alpha_{mi}\right) = \\ &\sum_{m=1}^{N\text{graded spaces}} \sum_{i}^{l} \alpha_{mi} - \frac{1}{2} \sum_{m=1}^{N\text{graded spaces}} \sum_{i,j}^{l} \alpha_{mi} \alpha_{mj} y_{mi} y_{mj} K\left(x_{mi}, x_{mj}\right) \end{aligned} \tag{21.51}$$

subject to the constraint

$$\sum_{m=1}^{N\text{graded spaces}} \sum_{i=1}^{l} \alpha_{mi} y_{mi} = 0 \tag{21.52}$$

where $\alpha_{mi} \geq 0$, for $m = 1, \ldots, N$ graded spaces and $i = 1, 2, \ldots, l$. This functional coincides with the functional for finding the optimal separating hyperplane for the multivector feature space.

## 7.3 Generating SMVMs with Different Kernels

The application of the kernel of Equation (21.45) results in a multivector classifier based on a multivector polynomial of degree $p$. An SMVM with kernels according to Equation (21.46), constitutes a *Gaussian radial basis function multivector* classifier. The reader can see that by using the SMVM approach we can straightforwardly generate multivector RBF networks, avoiding the complications of training that are encountered in the use of classical RBF networks. The number of the centers ($N_s = m \times 2^n$, $m2^n$*-dimensional multivectors*), the centers themselves ($s_i, i = 1, \ldots, N_s$), the weights ($\alpha_i, i = 1, \ldots, N_s$) and the threshold $b$ are all produced automatically by the SMVM training. This is an important contribution to the field, because in earlier work [16,21], we simply used extensions of the real-valued networks' RBFs for the so-called geometric RBF networks with classical training methods. We believe that the use of RBF–SMVM presents a promising neurocomputing tool for various applications of data coded in geometric algebra.

In the case of two-layer networks (Equation (21.47)), the first layer is composed of $N_{s1}$ sets of weights, with each set consisting of the dimension of the data $d = m \times 2^n$ and $m2^n$-dimensional multivectors, whereas the second layer is composed of $N_{s2}$ weights (the $\alpha_i$). In this way, according to Cybenko's theorem [4], an evaluation simply requires the consideration of the weighted sum of sigmoid functions, evaluated on inner products of test data with the multivector support vectors. Using our approach, we can see that the architecture ($N_{s2}$ weights) is found automatically by the SMVM.

## 7.4 Design of Kernels Involving the Clifford Geometric Product for Nonlinear Support Multivector Machines

The previous subsection shows that the kernels use multivectors as arguments. These kernels treat each of the multivectors of an $n$-dimensional geometric algebra simply as vectors of dimension $2^n$. This section presents the design of kernels which utilize the Clifford geometric product of multivectors. For that purpose we will use nonlinear mappings which map multivectors into a higher dimensional geometric algebra. Bear in mind that in any geometric algebra $G_n$ we can use an appropriate involution equivalent to the generalized inner product of the Hilbert space, consequently, the entries of the Gramm matrix are scalars and this matrix will be positive definite. For the case of complex numbers or quaternions, which can be found in Euclidean geometric algebras, the involution is given by the Clifford product of a multivector with the complex conjugate of a second multivector, i.e. for $\boldsymbol{x}'_{mi} = \Phi(\boldsymbol{x}_{mi}), \boldsymbol{x}'_{mj} = \Phi(\boldsymbol{x}_{mj}) \in G_{0,2,0}$, the involution is $\boldsymbol{x}'_{mi}\tilde{\boldsymbol{x}}'_{mj}$.

Using a combination of linear mappings denoted by $\phi_i(\cdot)$ we can design a new nonlinear mapping $\Phi(\cdot)$ which will transform the input multivector into a new one involving high-order products of the components of the original multivector. Let us consider a vector $\boldsymbol{x} \in \Re^n$, first we embed it in a higher dimensional linear space $\Re^{2^n}$, which has its associated geometric algebra $G_n$, then the vector will be transformed into a multivector by means of $\Phi(\cdot)$. This process has mapped the $n$-dimensional vector into a $2^n$-dimensional linear space. In its associated geometric algebra the mapped multivectors are $\Phi(\boldsymbol{x}_{mi}) = \boldsymbol{x}'_{mi}, \Phi(\boldsymbol{x}_{mj}) = \boldsymbol{x}'_{mj} \in G_n$. Now, in $G_n$ we can find an equivalent to the generalized inner product of the Hilbert space by considering the element of lowest grade of the geometric product, thus our kernel will be computed as follows:

$$\begin{aligned} K(\boldsymbol{x}_{mi}, \boldsymbol{x}_{mj}) &= \left\langle \Phi(\boldsymbol{x}_{mi})\Phi(\boldsymbol{x}_{mj}) \right\rangle_0 \\ &= \Phi(\boldsymbol{x}_{mi}) \cdot \Phi(\boldsymbol{x}_{mj}) = \boldsymbol{x}'_{mi} \cdot \boldsymbol{x}'_{mj} \end{aligned} \tag{21.53}$$

As a result we can easily compute the Gramm matrix, which requires scalar entries. Note that this kernel fulfills Mercer's condition, because the operation $\boldsymbol{x}'_{mi} \cdot \boldsymbol{x}'_{mj}$ corresponds to the basic kernel which uses simply the inner product.

$\Phi(\cdot)$ is computed by computing Clifford geometric products between different linear mappings $\phi_i$ as follows:

$$\Phi(\cdot) = \phi(\cdot)_1 \phi(\cdot)_2 \ldots \phi(\cdot)_p = \prod_i^m \phi(\cdot)_i \tag{21.54}$$

where $\prod_i^m$ stands for successive Clifford products. Depending on the kind of $\phi_i$ and the grade of $p$ we can get high nonlinear mappings. We present now two examples. Given $\boldsymbol{x} = x\sigma_1 + y\sigma_2$, we select the 2D Euclidean geometric algebra $G_{2,0,0}$ and compute $\Phi(\cdot)$ and the kernel as follows:

$$\begin{aligned} \phi_1(\boldsymbol{x}) &= x\sigma_1 + y\sigma_1\sigma_2 \\ \phi_2(\boldsymbol{x}) &= x\sigma_1 + y\sigma_2 \\ \Phi(\boldsymbol{x}) &= \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x}) \\ &= x^2 - y^2\sigma_1 - xy\sigma_2 - xy\sigma_1\sigma_2 \\ K(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \langle \Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j) \rangle_0 \end{aligned} \tag{21.55}$$

Note that $\Phi(\cdot)$ maps nonlinearly a 2D vector to a 4D multivector.

Given $\boldsymbol{x} = x\sigma_1 + y\sigma_2$, we select the 3D Euclidean geometric algebra $G_{3,0,0}$ and compute $\Phi(\cdot)$ and the kernel as follows:

$$\begin{aligned} \phi_1(\boldsymbol{x}) &= x\sigma_1 + y\sigma_3 \\ \phi_2(\boldsymbol{x}) &= x^2 - y^2\sigma_1 - xy\sigma_2 - xy\sigma_1\sigma_2 \\ \Phi(\boldsymbol{x}) &= \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x}) \\ &= -xy^2 + x^3\sigma_1 - x^2y\sigma_2 - x^2y\sigma_3 - xy^2\sigma_2\sigma_3 + \cdots \\ K(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \langle \Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j) \rangle_0 \end{aligned} \tag{21.56}$$

Note that $\Phi(\cdot)$ maps nonlinearly a 2D vector to an 8D multivector. These examples are illustrated in the next section.

## 7.5 Design of Kernels Involving the Conformal Neuron

In subsection 2.5, we outlined the conformal geometry $G_{n+1,1}$, which includes spheres. This is a potential geometric entity which can be used for encapsulating clusters. In this section, we present a new way to compress data in order to facilitate the learning of support multivector machines.

First we use a simple self-organizing network for the preprocessing of the clusters. Each one of them is modeled in the conformal space using spheres, see Figure 21.6(c). Note the interesting point: our data have been reduced to a small bunch of spheres, which in conformal geometry are represented simply as multivectors $\mathbf{s} \in G_{n+1,1}$ of $(n+2)$th dimension. Thereafter we supply these spheres to an SMVM for obtaining the optimal hyperplane, see Figure 21.6(d). In the case of nonlinear separable problems, we can apply the SMVM with Gaussian kernels.

You can see now the difference between the McCulloch–Pitts neuron in Figure 21.6(a) and the conformal neuron in Figure 21.6(e), the conformal neuron gets as input geometric entities of the conformal geometric algebra. The separating hyperplane in Figure 21.6(d) denoted by $\boldsymbol{w}$ (one element of the version space) can be found by an SMVM. However, we cannot guarantee that the point Figure 21.6(c) of the version space is a Tshebysheff or Bayes point. The design of an SMVM which will yield a hyperplane which approaches the Bayes point is one of our current research concerns.

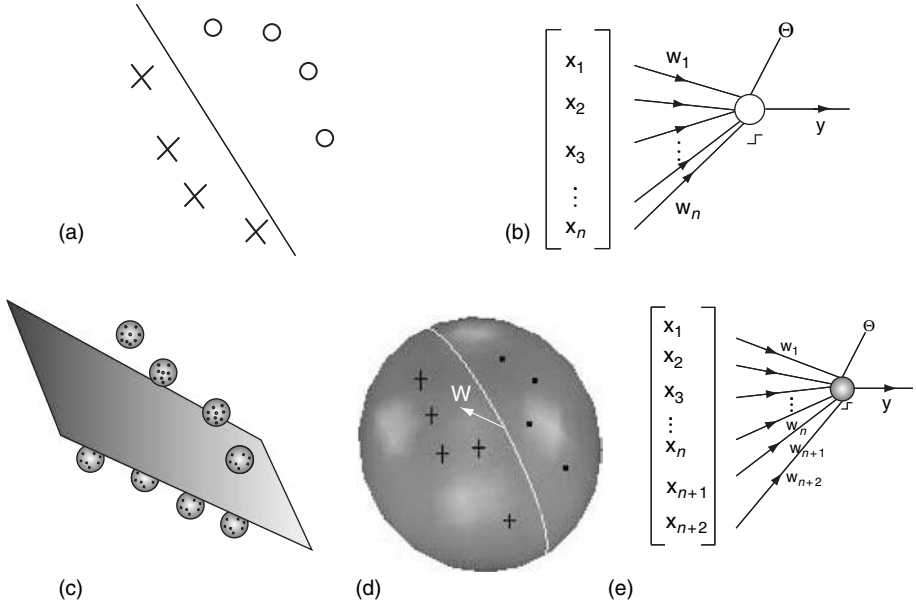



**Figure 21.6** (a) A linear separable case; (b) the perceptron separates nonoptimally the two classes; (c) preprocessing by clustering clouds of points using spheres; (d) a conformal neuron separates on a hypersphere the two classes of multivector (the spheres of (c)); (f) the conformal neuron with $2+n$ dimensional input. The separating hyperplane used by this neuron corresponds to the circle on the sphere in(d).

## 8. Clifford Moments for 2D Pattern Classification

In this section we give a brief introduction to Clifford moments.

The real moments of 2D patterns are computed using the well-known expression:

$$m_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^i y^j f(x,y)\,\mathrm{d}x\mathrm{d}y \tag{21.57}$$

where $i, j = 0, 1, 2, \ldots$ and their central moments:

$$\mu_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\overline{x})^i (y-\overline{y})^j f(x,y)\,\mathrm{d}x\mathrm{d}y \tag{21.58}$$

where $\overline{x} = \dfrac{m_{01}}{m_{00}}$ and $\overline{y} = \dfrac{m_{01}}{m_{00}}$. Note that the central moments are invariant under a Euclidean transformation. Similarly, the complex moments are given by:

$$C_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x+\mathrm{i}y)^i (x-\mathrm{i}y)^j f(x,y)\,\mathrm{d}x\mathrm{d}y \tag{21.59}$$

which can also be written in terms of $z = x + \mathrm{i}y$ as follows:

$$C_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (z)^i(\overline{z})^j f(x, y)\,\mathrm{d}x\mathrm{d}y \tag{21.60}$$

In the Clifford algebra we can generalize the standard moments to the hypercomplex moments as follows:

$$Cl_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (\boldsymbol{x})^i(\overline{\boldsymbol{x}})^i f(x, y)\,\mathrm{d}x\mathrm{d}y \tag{21.61}$$

where $\boldsymbol{x} \in G_{p,q,r}$ and $i, j = 0, 1, 2, \ldots$ Note that we are still using the same 2D manifold $f(x, y)$ but weighted with powers of multivectors $\boldsymbol{x}$. This kind of generalization is completely new. We claim that it is very limited in using simply the norm of invariants [22]; conversely, using the Clifford moments defined by Equation (21.61) we can have a much richer characterization of a 2D pattern. The next section illustrates the power of the use of Clifford moments.

In order to illustrate the computing of the Clifford moments as a method of preprocessing data, we take 2D regular shapes, see Figure 21.7. These were obtained by computer simulation. Note that it will not be possible to use real-valued moments for describing such shapes, because the real moments capture poorly the shape changes under different views of the objects. Only using quaternion-valued moments can we observe phase changes in the curves of the moments (see Figure 21.8).

As an illustration using different views of the patterns, we compute their moments $Cl_{21}Cl_{31}$ in the geometric algebra of the quaternions $G_{2,0,0}$. These moments are expressed as quaternions and are depicted in Figure 21.8. This kind of moment captures different symmetries, so there is no need to use all other moments. As a result, with this kind of preprocessing, we can keep low the input dimension of the neural network.

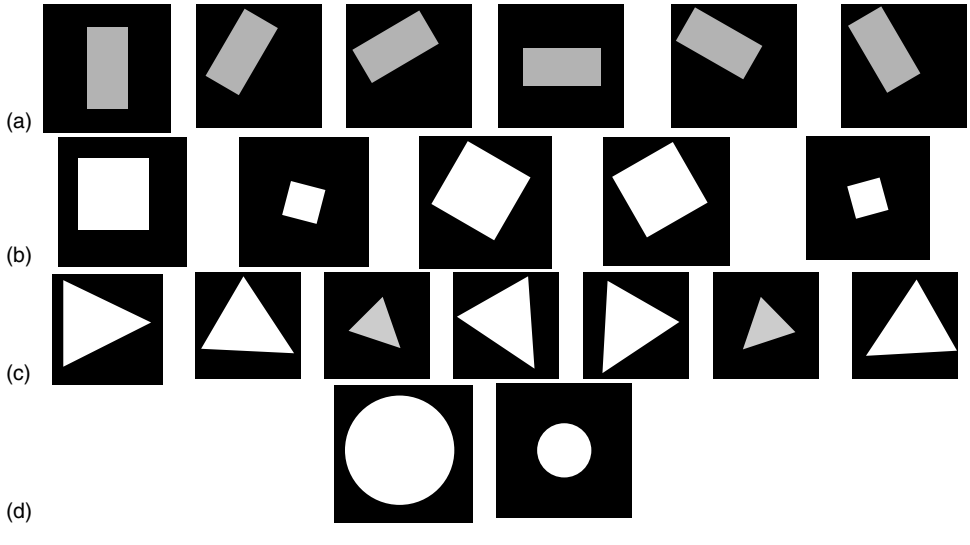



**Figure 21.7** Some views of 2D patterns for the classification problem. (a) Rectangle; (b) square; (c) triangle; (d) circle.

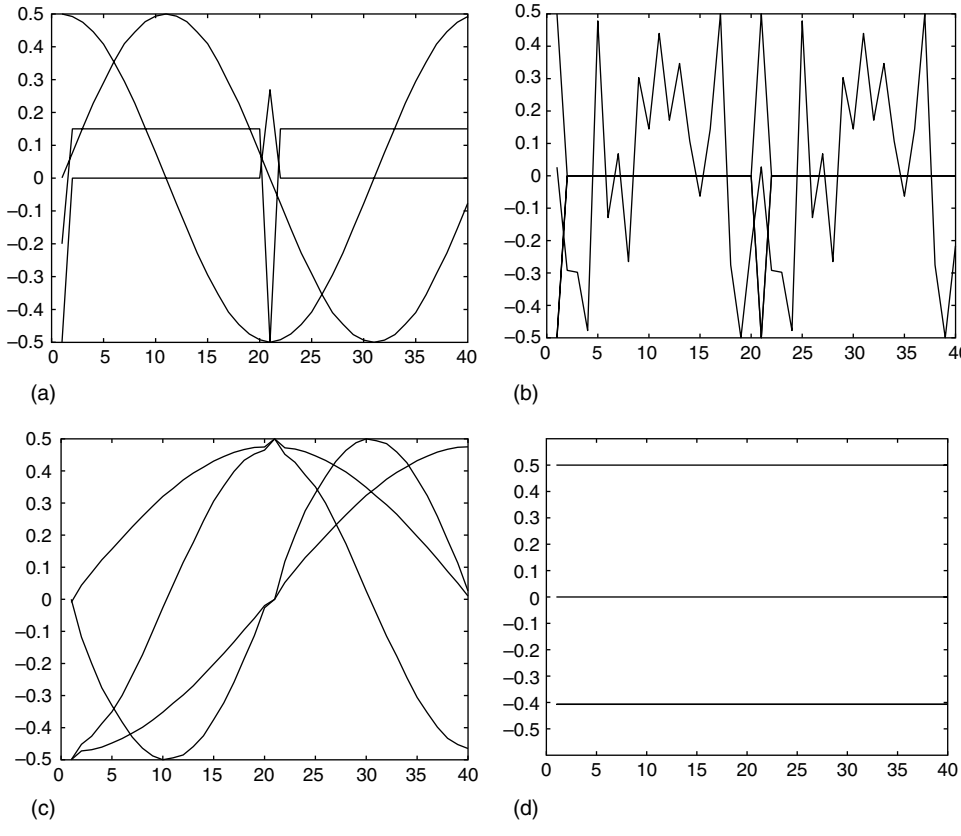



**Figure 21.8** Curves of the quaternionic moments of (a) a rectangle; (b) a square; (c) a triangle; and (d) a circle.

## 9. Experimental Analysis

This section presents tests and applications of the Clifford MLP and SMVM. We explore the use of Gaussian kernels, kernels generated via automorphisms and one example of an SMVM which uses clustering hyperspheres as preprocessing. In order to show the effectiveness of Clifford moments as a preprocessing method, we consider a case of 2D shape classification.

### *9.1 Test of the Clifford valued MLP for the XOR Problem*

In this section, the geometric MLPs were trained using the generalized gradient descendent. The power of using bivectors for learning was confirmed with the test using the XOR function. Figure 21.9(a) shows that the quaternion-valued geometric nets $\mathrm{GMLP}_{0,2,0}$ and $\mathrm{GMLP}_{2,0,0}$ have a faster convergence rate than either the MLP or the P-QMLP – the quaternionic multilayer perceptron of Pearson [15], which uses the activation function given by Equation (21.23).

Figure 21.9(a) shows the MLP with two-and four-dimensional input vectors. An epoch means one training set comprised of four training pars. Since the MLP(4), working also in 4D, cannot outperform the GMLPs, it can be claimed that the better performance of the geometric neural network is due not

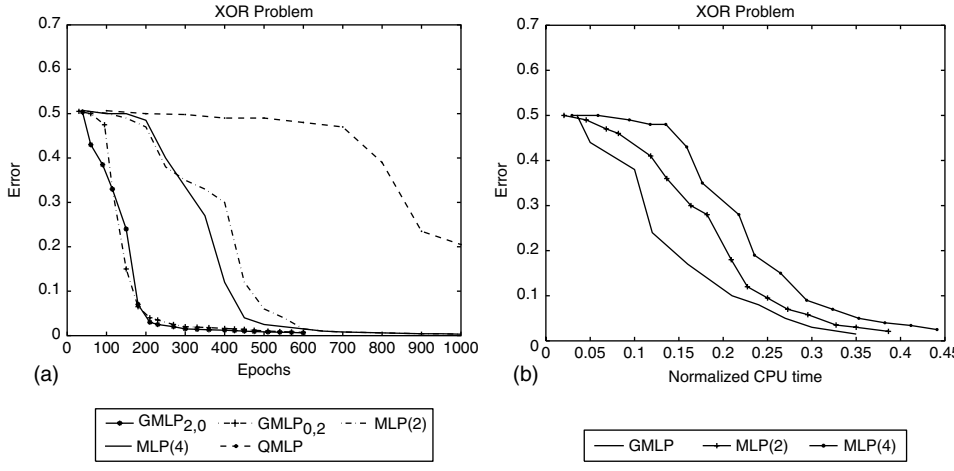



**Figure 21.9** (a) Learning XOR using the 2-input MLP denoted in the figure as MLP(2), 4-input MLP, denoted as MLP(4), quaternion-valued geometric nets $GMLP_{0,2,0}$ and $GMLP_{2,0,0}$ and P-QMLP; (b) comparison of the geometric neural network $GMLP_{0,2,0}$, the 2-input MLP, denoted MLP(2), and the 4-input MLP, denoted MLP(4).

to the higher dimensional quaternionic inputs, but rather to the algebraic advantages of the geometric neurons of the net.

Figure 21.9(b) shows the error curves referring to the involved normalized computational time. The demanding computer resources of the MLP with four-dimensional input vectors (MLP(4)) is almost similar to the geometric neural network, however, its performance is lower than the geometric network. The learning performance of the MLP with two inputs (MLP(2)) is still not as good as that of the geometric neural net $\text{GMLP}_{0,2,0}$.

The reader should see that the learning of geometric neural nets is improved due to the Clifford product computed in a higher dimensional space. It is worth noting in Figure 21.9(b) that the MLP(4), even working in a four dimensional space, cannot perform better than the geometric neural net.

## *9.2 Classification of 2D Patterns in Real Images*

Now we will show an application of Clifford moments and geometric MLP using real images of the objects presented in Figure 21.10. The binarized images used to generate the moments

**Figure 21.10** Images of the real objects.

$Cl_{21}$, $Cl_{31}$ are depicted in Figure 21.11. The resulting curves of the Clifford moments are presented in Figure 21.12.

Using binarized views of the objects unseen before by the neural network, we test the geometric MLP. The results in Table 21.1 show that the procedure performs well.

**Figure 21.11** Binarized images of the objects.

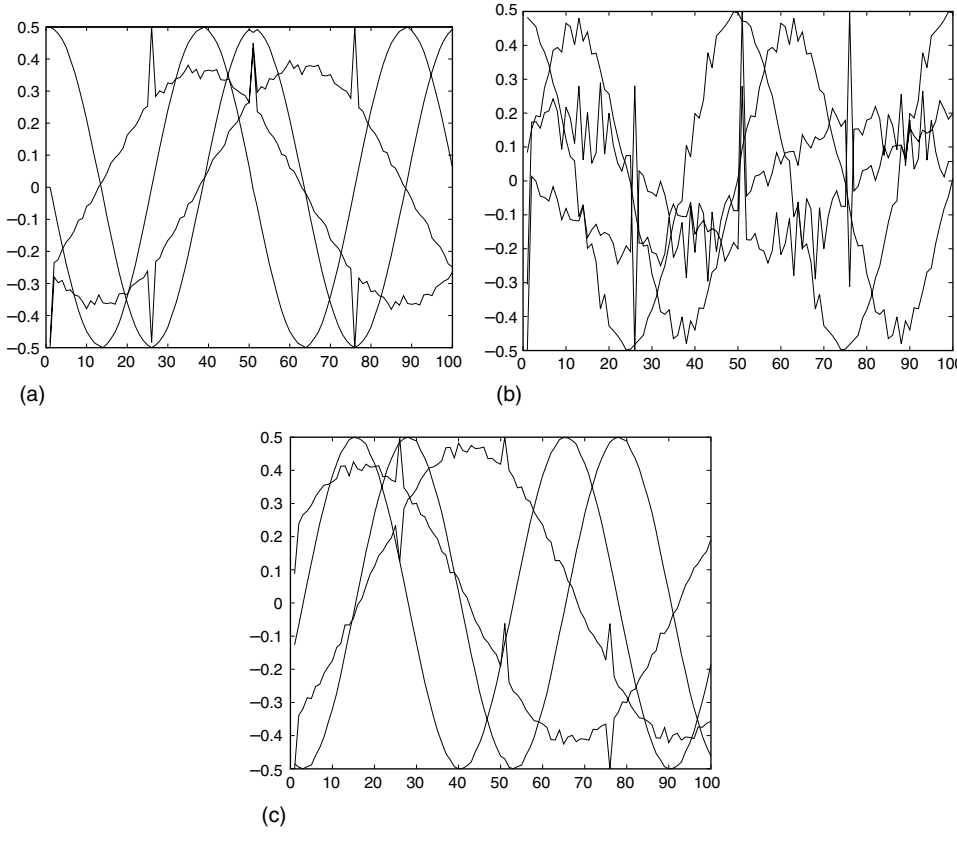


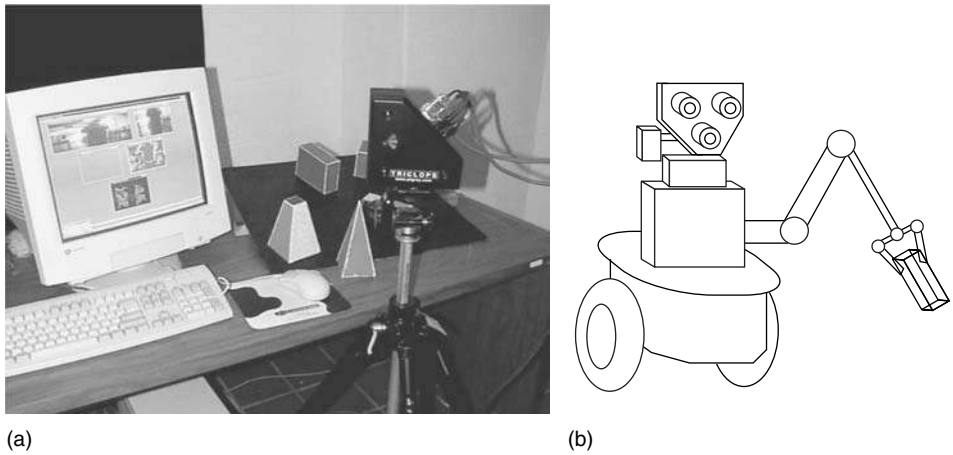

**Figure 21.12** Curves of the quaternionic moments of (a) a cylinder; (b) a cube and (c) a truncated pyramid.

**Table 21.1** Results using real images.

| | Classified | Misclassified |
|---|---|---|
| Cylinder | 85 % | 15 % |
| Cube | 86 % | 14 % |
| Pyramid | 83 % | 17 % |

## *9.3 Estimation of 3D Pose*

This experiment shows the application of SMVMs for the estimation of the 3D pose of rigid objects. Figure 21.13(b) shows the equipment used. Images of the object were captured using a trinocular head. This is a calibrated three-camera system which computes the 3D depth using a triangulation algorithm [23,24]. We extracted 3D relevant features using the 2D coordinates of corners and lines of the central view of the trinocular camera array. For this we applied edge and corner detectors and the Hough transform (see Figure 21.14(b)). After we had found the relevant 3D points (see Figure 21.14(a)), we applied a regression operation to gain the 3D lines in order to be note to compute intersecting points (see Figure 21.14(b)). This helps enormously to correct distorted 3D points or to compute the 3D coordinates of occluded points. Since lines and planes are more robust against noise than points, we used them to describe the object shape in terms of motor algebra representations of 3D lines and planes (see Figure 21.14(b)). In order to estimate the 3D pose of the object, we trained an SMVM using the multivector representation of the object as training input and we chose its 3D pose (in terms of two points lying on a 3D line) as output, which in turn also told us where the object should be held by a robot gripper. We trained an SMVM using polynomial kernels with data of each object extracted from trinocular views taken every 12 degrees. In recall the SMVM was able to estimate the holding line for an arbitrary position of the object, as Figure 21.14(b) shows. The underlying idea of the experiment using platonic objects was to test whether the SMVM could handle this kind of problem. In future, we will apply a similar strategy for estimating the pose of real objects in order to orient a robot gripper.




(a) (b)

**Figure 21.13** (a) Visually guided robot manipulation; (b) trinocular head for 3D object recognition.

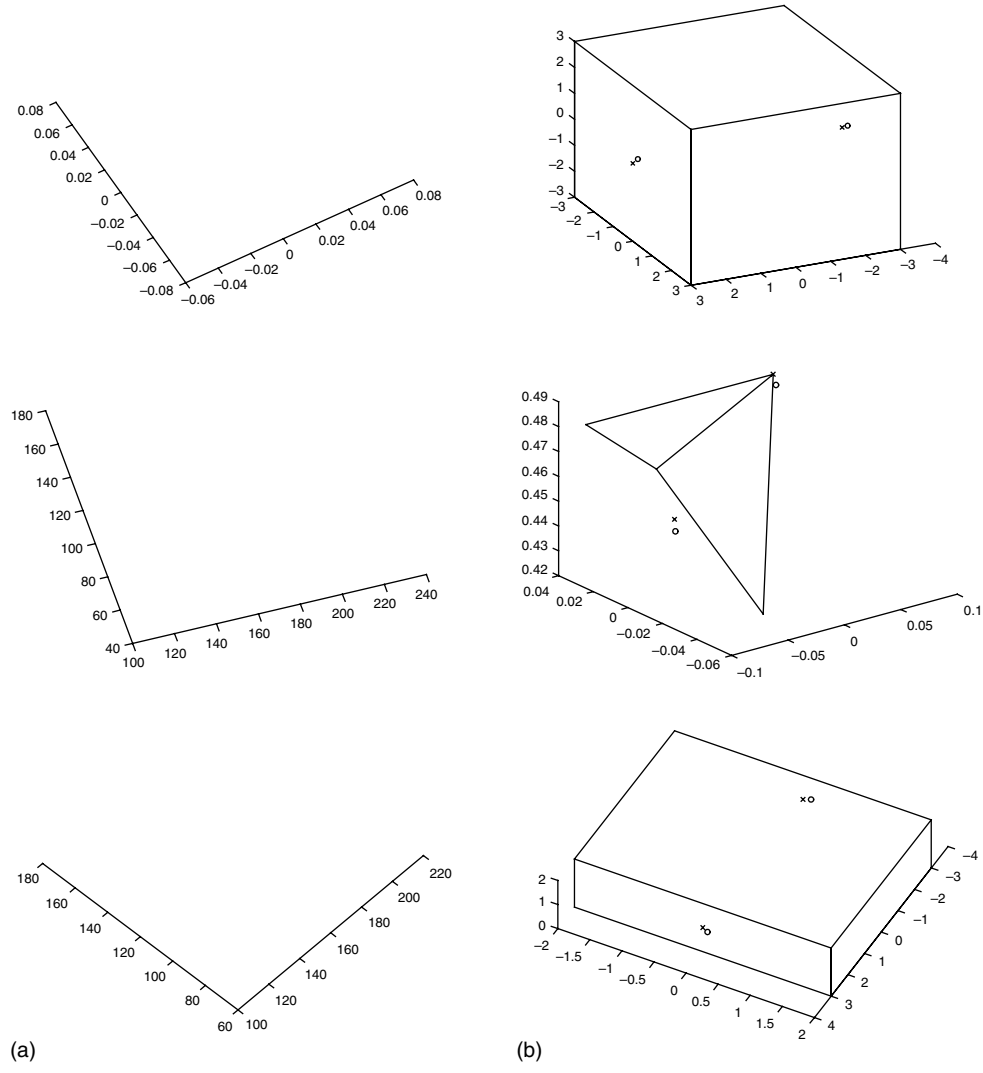



**Figure 21.14** Estimation of the 3D pose of objects. (a) The 3D points of each object; (b) completed object shape and estimated grasp points using SMVM, circle ground true and cross estimated.

## *9.4 Performance of SMVMs Using Kernels Involving the Clifford Product*

The next tests showed the performance of the kernels involving the Clifford product. These kernels were implemented using the functions $\Phi(\cdot)$ given by Equations (21.55) and (21.56). The pseudocode of the used algorithm is given next. The algorithm stopped if the iteration $t$ had reached its maximum value $t_{\max}$ or the margin $\gamma \approx 1$, for the latter, the algorithm should have found the optimal $\alpha$ s. In the algorithm **GM** $[i][j][0]$ denotes the array for the Gramm matrix using the blade of grade zero

(scalar part) of the Clifford product between $\boldsymbol{x}'_{mi} = \Phi(\boldsymbol{x}_{mi})$ and $\boldsymbol{x}'_{mj} = \Phi(\boldsymbol{x}_{mj})$, this is equivalent to computing the generalized inner product of the multivectors.

```
Initialize α_i = 0
For t = 1 to t_max
if t = 0
  b_0 = 0.1
else
    if t = 1
      b_1 = −0.1
    else
      b_t = b_{t−1} − w_{t−1} ((b_{t−1} − b_{t−2}) / (w_{t−1} − w_{t−2}))
for i = 1 to N
% for multivector and label x'_i = Φ(x_mi, x_i)
    z_i = Σ α_j y_j GM[i][j][0]
    Δα_i^t = η(1 − z_i y_i − b^t y_i)
if (α_i^t + Δα_i^t) ≤ 0 then α_i^t = 0
    else α_i^t ← α_i^t + Δα_i^t
end
w^t = Σ_j Δα_i^t y_j
compute margin γ
if γ ≈ 1 then break
end.
```

Figure 21.15(a) to (c) show a 2D nonseparable case in a 4D space using the kernel given by Equation (21.55). We can appreciate that our method works very well.

Figures 21.15(d)–(e) shows two nonlinear separable cases in an 8D space using in both of them, the kernel given by Equation (21.56). Figure 21.15(f) shows a 3D nonseparable case again using the kernel given by Equation (21.56), which works in an 8D space. We have shown that the kernels involving the Clifford product find optimal hyperplanes.

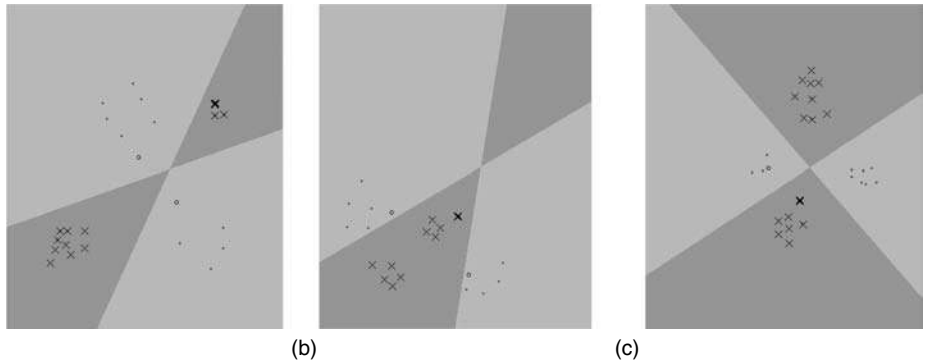
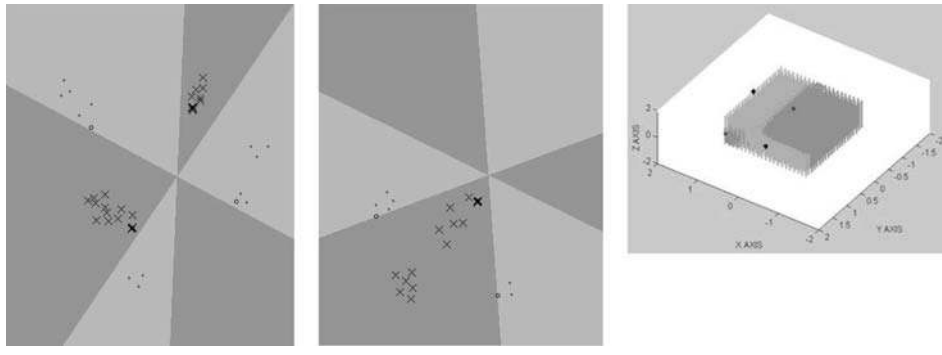



**Figure 21.15** (a),(b),(c) Nonlinear 2D separable classification using the kernel of Equation (21.55), which works in 4D; (d),(e),(f) nonlinear separable classification cases using the kernel of Equation (21.56), which works in 8D. The first two experiments are 2D cases and the third a 3D case.

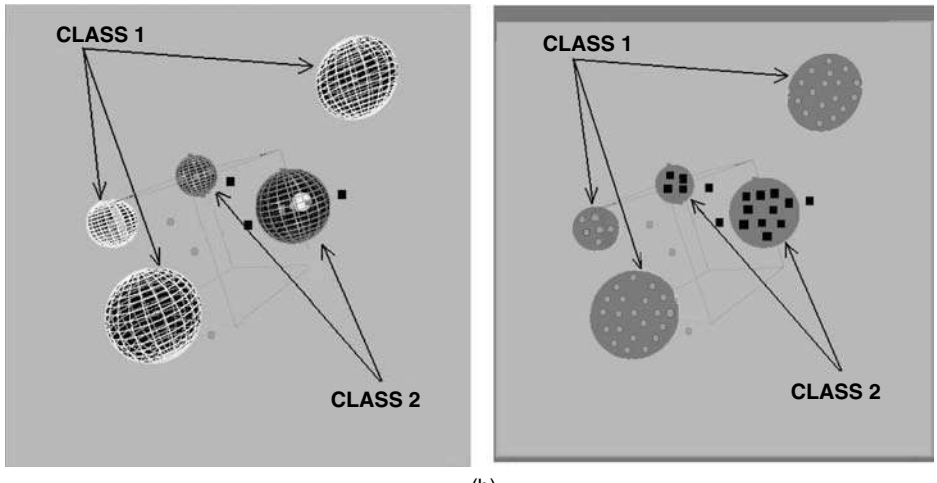



(d) (e) (f)

**Figure 21.15** (*continued*)

## *9.5 An SMVM Using Clustering Hyperspheres*

The last experiment clustered 3D data using the method explained in Section 7.5. Figure 21.16 shows the spheres that were found via a self-organizing neural network. They were coded as spheres in the conformal geometric algebra $G_{4,1}$ and only these were given to the SMVM for finding the optimal nonlinear hyperplane. Note that after training, all the vectors inside the spheres are classified correctly. Even points near to both sides of the nonlinear optimal hyperplane are classified correctly. The coding of the spheres, points and points near to the optimal hyperplane was as follows: we used the conformal geometric algebra $G_{4,1}$ and vectors were represented using the basis $\sigma_1\sigma_2\sigma_3 e_+ e_-$. Table 21.2 and Figure 21.16 show the coding and classification results. Note the successful classification of the points near to the decision border.




(a) (b)

**Figure 21.16** Nonlinear separable classification cases using the conformal kernel. (a) Spheres; (b) the patterns inside the spheres.

**Table 21.2** Classification of spheres, points in and out of the spheres.

| Object | Expected | Obtained |
|---|---|---|
| s1 = [1, 2, 0, 1.5, 2.5] | 1 | 1 |
| s2 = [0, 0, 0, 1.5, 1.5] | 1 | 1 |
| s3 = [−1.5, −1.5, −1.5, 2.155, 3.155] | −1 | −1 |
| s4 = [−3, −1, 1, 4.755, 5.755] | −1 | −1 |
| s5 = [3, 3, 2.5, 11.22, 12.22] | −1 | −1 |
| sphere inside sphere s1 | | |
| s6 = [1, 2.5, 3.09375, 4.09375] | −1 | −1 |
| points inside S1 | | |
| ps11 = [1, 2, 1, 1.5, 3.5] | 1 | 1 |
| ps12 = [1, 2.5, 0, 3.125, 4.125] | 1 | 1 |
| ps13 = [1, 1.5, 0, 1.125, 2.125] | 1 | 1 |
| ps14 = [1.5, 2, 0, 2.625, 3.625] | 1 | 1 |
| ps15 = [0.5, 2.2, 0, 2.045, 3.045] | 1 | 1 |
| points inside S2 | | |
| ps21 = [0, 0, 1.5, 0.625, 1.625] | 1 | 1 |
| ps22 = [0, 0.5, 1.5, 0.75, 1.75] | 1 | 1 |
| ps23 = [0, −0.5, 1.5, 0.75, 1.75] | 1 | 1 |
| ps24 = [0.5, 0, 1.5, 0.75, 1.75] | 1 | 1 |
| points inside S3 | | |
| ps31 = [−1.5, −1.5, −1.5, 2.875, 3.875] | −1 | −1 |
| ps32 = [−2, −2.1, −2.2, 6.125, 7.125] | −1 | −1 |
| ps33 = [−1, −1.5, −1.2, 1.845, 2.845] | −1 | −1 |
| ps34 = [−1.5, −1.5, −0.5, 1.875, 2.875] | −1 | −1 |
| points inside S4 | | |
| ps41 = [−3, −1, 1, 5, 6] | −1 | −1 |
| ps42 = [−2.4, −1.2, 1.3, 3.945, 4.945] | −1 | −1 |
| ps43 = [−3, −0.5, 0.5, 4.25, 5.25] | −1 | −1 |
| ps44 = [−3, −0.4, 1, 4.58, 5.58] | −1 | −1 |
| points inside S5 | | |
| ps51 = [3, 3, 2.5, 11.625, 12.625] | −1 | −1 |
| ps52 = [3.7, 3, 2.5, 13.97, 14.97] | −1 | −1 |
| ps53 = [3, 3.5, 2.5, 13.25, 14.25] | −1 | −1 |
| ps54 = [3, 3, 3.2, 13.62, 14.62] | −1 | −1 |
| outside the spheres and near to the optimal hyperplane | | |
| po1 = [2.5, 2, 0, 4.625, 5.625] | 1 | 1 |
| po2 = [1, 0.5, 0, 0.125, 1.125] | 1 | 1 |
| po3 = [0, −0.9, 1.5, 1.03, 2.03] | 1 | 1 |
| po4 = [−0.3, −0.3, −1.5, 0.715, 1.715] | −1 | −1 |
| po5 = [−0.3, −0.3, −2.5, 2.715, 3.715] | −1 | −1 |
| po6 = [−2.1, −0.3, 1, 2.25, 3.25] | −1 | −1 |

## 10. Conclusions

This chapter has shown the analysis and design of feed-forward neural networks and support vector machines using the coordinate-free system of Clifford, or geometric, algebra. It has been shown that real-, complex- and quaternion-valued neural networks are simply particular cases of geometric algebra multidimensional neural networks. We designed kernels for support multivector machines, which involve the Clifford product. The conformal neuron was used for clustering data, this idea is very useful for alleviating the complexity and computational demand in classification problems. In neural computing, preprocessing is of foremost importance, in this regard, we introduced a novel method of geometric preprocessing utilizing hypercomplex or Clifford moments. This method was applied, together with geometric MLPs, for tasks of 2D pattern classification.

The design of feed-forward neural networks and SVMs in the geometric algebra framework allows us to expand their sphere of applicability for multidimensional learning. The experimental analysis shows the potential of such geometric neural networks for a variety of real applications using multidimensional representations.

## References

[1] Pellionisz, A. and Llinas, R. "Tensorial approach to the geometry of brain function: cerebellar coordination via a metric tensor," *Neuroscience*, **5**, pp. 1125–1136, 1980.

[2] Pellionisz, A. and Llinas, R. "Tensor network theory of the metaorganization of functional geometries in the central nervous system", *Neuroscience*, **16**(2), pp. 245–273, 1985.

[3] Hertz, J., Krogh, A. and Palmer, R.G. *Introduction to the Theory of Neural Computation*, Addison-Wesley, 1991.

[4] Cybenko, G. "Approximation by superposition of a sigmoidal function," *Mathematics of Control, Signals and Systems*, **2**, pp. 303–314, 1989.

[5] Hestenes, D. *Space–Time Algebra*, Gordon and Breach, 1966.

[6] Bayro-Corrochano, E. *Geometric Computing for Perception-Action Systems*, Springer Verlag, New York, 2001.

[7] Hestenes, D. and Sobczyk, G. *Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics*, D. Reidel, Dordrecht, 1984.

[8] Li, H., Hestenes, D. and Rockwood A. "Generalized homogeneous coordinates for computational geometry," in Sommer, G. (Ed.), *Geometric Computing with Clifford Algebra*, pp. 27–59, Springer Verlag, 2001.

[9] Hornik, K. "Multilayer feedforward networks are universal approximators," *Neural Networks*, **2**, pp. 359–366, 1989.

[10] Rumelhart, D.E. and McClelland, J.L. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 2 Vols, MIT Press, Cambridge, 1986.

[11] Poggio, T. and Girosi, F. "Networks for approximation and learning," *IEEE Proceedings*, **78**(9), pp. 1481–1497, 1990.

[12] Georgiou, G. M. and Koutsougeras, C. "Complex domain backpropagation," *IEEE Transactions on Circuits and Systems*, **330**, pp. 334, 1992.

[13] Arena, P., Caponetto, R., Fortuna, L., Muscato, G. and Xibilia, M.G. "Quaternionic multilayer perceptrons for chaotic time series prediction," *IEICE Transactions on Fundamentals*, **E79-A**(10), pp. 1–6, 1996.

[14] Hamilton, W.R. *Lectures on Quaternions*, Hodges and Smith, Dublin, 1853.

[15] Pearson, J.K. and Bisset, D.L. "Back Propagation in a Clifford Algebra," in Aleksander, I. and Taylor, J. (Ed), *Artificial Neural Networks* 2, pp. 413–416, Elsevier Science Publishers, 1992.

[16] Bayro-Corrochano, E., Buchholz, S. and Sommer, G. "Selforganizing Clifford neural network," *IEEE ICNN'96* Washington, DC, pp. 120–125, 1996.

[17] Perantonis, S.J. and Lisboa, P.J.G. "Translation, rotation, and scale invariant pattern recognition by high-order neural networks and moment classifiers", *IEEE Transactions on Neural Networks*, **3**(2), pp 241–251, 1992.

[18] Porteous, I.R. *Clifford Algebras and the Classical Groups*, Cambridge University Press, Cambridge, 1995.

[19] Holland, J. *Adaptation in Neural and Artificial Systems*, University of Michigan Press. Ann Arbor, 1975.

[20] Vapnik, V. *Statistical Learning Theory*, John Wiley & Sons, Inc., New York, 1998.

[21] Bayro-Corrochano, E. “Clifford Selforganizing Neural Network, Clifford Wavelet Network”. *Proceedings of 14th IASTED International Conference on Applied Informatics*, Innsbruck, Austria, pp. 271–274, 1996.
[22] Reiss, T.H. *Recognizing Planar Objects Using Invariant Image Features*, Springer Verlag, Berlin, 1993.
[23] Tsai, R.Y. “An efficient and accurate camera calibration technique for 3D machine vision,” in *Proceedings of International Conference on Computer Vision and Pattern Recognition*, Miami Beach, Florida, USA, pp. 364–374, 1986.
[24] Klette, R., Schlüns, K. and Koschan, A. *Computer Vision. Three-Dimensional Data from Images*, Springer-Verlag, Singapore, 1998.

# 22

# Intelligent Recognition: Components of the Short-time Fourier Transform vs. Conventional Approaches

**Leonid Gelman**
**Mike Sanderson**
**Chris Thompson**
Department of Process and Systems Engineering, Cranfield University, Cranfield, MK43 0AL, UK

**Paul Anuzis**
Rolls-Royce plc, Derby, UK

*A new feature representation approach is generalized and used for Gaussian recognition. The generalized approach consists of using two new recognition features – the real and imaginary Fourier components – taking into account the covariance between features. The generalization approach improves the recognition effectiveness. An advanced time–frequency technique, the short-time Fourier transform, is considered. The covariance and the correlation coefficient between the proposed features are obtained for the first time for arbitrary stationary signals. The recognition effectiveness between the generalized approach and the Hartley, cosine and Power Spectral Density (PSD) approaches is compared. It is shown that the Hartley, cosine and PSD approaches are not optimal. Use of the generalized approach provides an essential increase in effectiveness in comparison with the Hartley, cosine and PSD approaches. Application of the generalized approach is considered for vibration diagnostics of object damping and fatigue. The application results agree with the theoretical results.*

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

## 1. Introduction

A new generic approach for feature representation has been proposed [1,2] for those cases in which one- or multidimensional Fourier transforms are used for computer-aided intelligent pattern recognition. The approach consists of using, simultaneously, two new recognition features: the real and imaginary components of the Fourier transform.

This approach is in contrast to intelligent recognition applications, including Fourier optics applications, where the Power Spectral Density (PSD) or the phase spectrum are used [3–33]. It has been shown that the new approach is more generic than the PSD and phase spectrum approaches and provides better recognition effectiveness than the power spectral density approach.

However, the approach under discussion here was proposed and investigated [1,2] only for a particular case, without taking into account the covariance between new features. Taking covariance between recognition features into account may improve the intelligent recognition effectiveness [5,34–36].

Therefore, to generalize the proposed approach, we need to estimate and take into account the covariance between the features, i.e. the Fourier components. The covariance between the real and imaginary components of the general Fourier transform, e.g. the *short-time* Fourier transform, which is an advanced time–frequency technique, has been little studied. Jenkins and Watts [37] have obtained this covariance only for the classical (not the short-time) Fourier transform and Gaussian white noise.

In recent years, the Hartley and cosine transforms [35,38] have gained importance in pattern recognition, image processing and applications. The Hartley transform is similar to the Fourier transform, but is computationally much faster [35] than even the fast Fourier transform.

The purposes of this chapter are to:

- estimate the covariance and correlation coefficient between the new features for the *short-time* Fourier transform of *arbitrary* stationary signals, generalize the approach [1,2] and estimate the effect of generalization;
- compare the recognition effectiveness of the generalized approach with the following conventional approaches (for the common case of nonzero covariance between the proposed features):
  - — the Hartley transform approach;
  - — the cosine transform approach;
  - — the PSD approach;
- employ the proposed approach in damping and fatigue diagnostics.

## 2. Theoretical Analysis

The approach [1,2] consists of using, simultaneously, the real $X_R$ and imaginary $X_I$ components of the Fourier transform at the frequency $\omega_x$ as parameters (diagnostic features):

$$X_R(\omega_x, T) = \int_0^T x(t) \cos \omega_x t \mathrm{d}t \tag{22.1}$$

$$X_I(\omega_x, T) = \int_0^T x(t) \sin \omega_x t \mathrm{d}t \tag{22.2}$$

where $T$ is the time length of the short-time Fourier transform.

We consider the general case of the Fourier transform, the short-time Fourier transform (i. e. $T \neq \infty$), which is the advanced time–frequency signal processing technique.

The covariance between the features in Equations (22.1) and (22.2) for arbitrary stationary signals, employing only signal stationarity, can be presented as:

$$K_{\mathrm{RI}} = m(X_{\mathrm{R}} X_{\mathrm{I}}) = \int_0^T \int_0^T K_x(t_2 - t_1) \cos \omega t_1 \sin \omega t_2 \mathrm{d}t_1 \mathrm{d}t_2 \tag{22.3}$$

where $m$ is the mean operator and $K_x(t_2 - t_1)$ is the autocovariance function of the signal $x(t)$.

Using a new variable $\tau = t_2 - t_1$, reversing the orders of integration between $\tau$ and $t_1$ and carrying out the $t_1$ integration, after transformations we find the final equation:

$$K_{\mathrm{RI}} = \frac{\sigma_x^2 \int_0^T r_x(\tau)[\cos \omega\tau - \cos \omega(2T - \tau)]\mathrm{d}\tau}{2\omega} \tag{22.4}$$

where $r_x(\tau)$ is the normalized autocovariance function of the signal.

Employing Equation (22.4) and standard deviations $\sigma_{\mathrm{R}}$ and $\sigma_{\mathrm{I}}$ respectively of the components $X_{\mathrm{R}}$ and $X_{\mathrm{I}}$ from [39], the correlation coefficient can be presented, after transformations, as:

$$r_{\mathrm{RI}} = \frac{0.5 \int_0^T r_x(\tau)[\cos \omega\tau - \cos \omega(2T - \tau)]\mathrm{d}\tau}{\sqrt{\left\{\int_0^T r_x(\tau)[\omega(T - \tau)\cos \omega\tau - \sin \omega\tau]\mathrm{d}\tau\right\}\left\{\int_0^T r_x(\tau)[\omega(T - \tau)\cos \omega\tau + \sin \omega\tau]\mathrm{d}\tau\right\}}} \tag{22.5}$$

Equations (22.4) and (22.5) are new and generic. They are obtained for *arbitrary stationary* signals.

We find from Equations (22.4) and (22.5) that, generally, for the *short-time* Fourier transform, the covariance and correlation coefficient between the features in Equations (22.1) and (22.2) are *nonzero*. Therefore, we generalize the approach [1,2]. The generalized approach consists of using, simultaneously, two new recognition features *taking into account the covariance between these features*.

Now we employ the generalized approach for the following two-class recognition of stationary Gaussian narrowband centered signals $x(t)$:

- hypothesis $H_0$: signal variance is $\sigma_{x0}^2$; normalized autocovariance is $r_x$;
- hypothesis $H_1$: signal variance is $\sigma_{x1}^2$; normalized autocovariance is $r_x$.

The recognition information is contained in the short-time Fourier transform at the frequency $\omega_x$.

The recognition feature based on the short-time Hartley transform can be written in the form [35]:

$$X_{\mathrm{H}}(\omega_x, t_1) = \int_0^{t_1} x(t)(\cos \omega_x t + \sin \omega_x t)\,\mathrm{d}t = X_{\mathrm{R}} + X_{\mathrm{I}} \tag{22.6}$$

The recognition feature based on the short-time cosine transform can be written in the form [38]:

$$X_{\mathrm{C}}(\omega_x, t_1) = \int_0^{t_1} x(t) \cos \omega_x t \mathrm{d}t = X_{\mathrm{R}} \tag{22.7}$$

We notice that these approaches are particular cases of the proposed approach.

Likelihood ratios of the features in Equations (22.1), (22.2), (22.6) and (22.7) are defined respectively as:

$$S_{\mathrm{o}}(X_{\mathrm{R}}, X_{\mathrm{I}}) = \ln \frac{W(X_{\mathrm{R}},\, X_{\mathrm{I}}|\, H_1)}{W(X_{\mathrm{R}},\, X_{\mathrm{I}}|\, H_0)} \tag{22.8}$$

$$S_{\mathrm{H}}(X_{\mathrm{H}}) = \ln \frac{W(X_{\mathrm{H}}|H_1)}{W(X_{\mathrm{H}}|H_0)} \tag{22.9}$$

$$S_{\mathrm{C}}(X_{\mathrm{C}}) = \ln \frac{W(X_{\mathrm{C}}|H_1)}{W(X_{\mathrm{C}}|H_0)} \tag{22.10}$$

where $W\left(X_{\mathrm{R}},\, X_{\mathrm{I}}|\, H_j\right)$, $W\left(X_{\mathrm{H}}|\, H_j\right)$ and $W\left(X_{\mathrm{C}}|\, H_j\right)$ are the conditional probability density functions (pdfs) of the features in Equations (22.1), (22.2), (22.6) and (22.7) respectively, for hypothesis $H_j$.

Obviously, the proposed features, the cosine and Hartley-based features are Gaussian variables.

Using the bivariate Gaussian conditional pdf of the features in Equations (22.1) and (22.2), we obtain the feature likelihood ratio Equation (22.8) in the form:

$$S_{\mathrm{o}} = AX_{\mathrm{R}}^2 + BX_{\mathrm{I}}^2 + CX_{\mathrm{R}}X_{\mathrm{I}} + D \tag{22.11}$$

where $A = \dfrac{\sigma_{\mathrm{R1}}^2 - \sigma_{\mathrm{R0}}^2}{2(1-r^2)\sigma_{\mathrm{R0}}^2\sigma_{\mathrm{R1}}^2}$, $B = \dfrac{\sigma_{\mathrm{I1}}^2 - \sigma_{\mathrm{I0}}^2}{2(1-r^2)\sigma_{\mathrm{I0}}^2\sigma_{\mathrm{I1}}^2}$, $C = \dfrac{r(\sigma_{\mathrm{R0}}\sigma_{\mathrm{I0}} - \sigma_{\mathrm{R1}}\sigma_{\mathrm{I1}})}{(1-r^2)\sigma_{\mathrm{R0}}\sigma_{\mathrm{R1}}\sigma_{\mathrm{I0}}\sigma_{\mathrm{I1}}}$, $D = \ln \dfrac{\sigma_{\mathrm{R0}}\sigma_{\mathrm{I0}}}{\sigma_{\mathrm{R1}}\sigma_{\mathrm{I1}}}$ $\sigma_{\mathrm{R}j}$ and $\sigma_{\mathrm{I}j}$ are standard deviations of the components $X_{\mathrm{R}}$ and $X_{\mathrm{I}}$ respectively for hypothesis $H_j$ and $r$ is the correlation coefficient between features in Equations (22.1) and (22.2). This correlation coefficient is identical for hypothesis $H_j$, $j = 0, 1$, due to the identical normalized autocovariance functions of the signals for both hypotheses. Employing [39] we obtain: $\dfrac{\sigma_{\mathrm{R1}}}{\sigma_{\mathrm{R0}}} = \dfrac{\sigma_{\mathrm{I1}}}{\sigma_{\mathrm{I0}}} = \dfrac{\sigma_{x1}}{\sigma_{x0}}$.

Using Equation (22.6) and the one-dimensional Gaussian pdf, we find the likelihood ratio of Equation (22.9) in the form:

$$S_{\mathrm{H}} = DX_{\mathrm{H}}^2 + E = D(X_{\mathrm{R}}^2 + X_{\mathrm{I}}^2 + 2X_{\mathrm{R}}X_{\mathrm{I}}) + E \tag{22.12}$$

where

$$D = \frac{1}{2}\left(\frac{1}{\sigma_{\mathrm{R0}}^2 + \sigma_{\mathrm{I0}}^2 + 2r\sigma_{\mathrm{R0}}\sigma_{\mathrm{I0}}} - \frac{1}{\sigma_{\mathrm{R1}}^2 + \sigma_{\mathrm{I1}}^2 + 2r\sigma_{\mathrm{R1}}\sigma_{\mathrm{I1}}}\right),$$

$$E = \frac{1}{2}\ln \frac{\sigma_{\mathrm{R0}}^2 + \sigma_{\mathrm{I0}}^2 + 2r\sigma_{\mathrm{R0}}\sigma_{\mathrm{I0}}}{\sigma_{\mathrm{R1}}^2 + \sigma_{\mathrm{I1}}^2 + 2r\sigma_{\mathrm{R1}}\sigma_{\mathrm{I1}}}$$

Using Equation (22.7) and the one-dimensional Gaussian pdf, we find the likelihood ratio of Equation (22.10) in the form:

$$S_{\mathrm{C}} = EX_{\mathrm{R}}^2 + J \tag{22.13}$$

where $E = \dfrac{1}{2}\left(\dfrac{1}{\sigma_{\mathrm{R0}}^2} - \dfrac{1}{\sigma_{\mathrm{R1}}^2}\right)$, $J = \ln \dfrac{\sigma_{\mathrm{R0}}}{\sigma_{\mathrm{R1}}}$.

One can see the difference between the likelihood ratios in Equation (22.11) and Equations (22.12) and (22.13). We find that the likelihood ratios in Equations (22.12) and (22.13) are not even particular cases of the likelihood ratio in Equation (22.11).

From Equation (22.11) we also note that, generally, the likelihood ratio of the proposed features is not the PSD, which can be presented in the form (except for a constant multiplier, which is nonessential for recognition):

$$S_{\mathrm{PSD}} = X_{\mathrm{R}}^2 + X_{\mathrm{I}}^2 \tag{22.14}$$

Therefore, the use of the Hartley, cosine and PSD approaches is not optimal for the recognition under consideration.

We find from Equations (22.11) and (22.14) that if simultaneously: the features in Equations (22.1) and (22.2) are uncorrelated, i.e. $r = 0$, and the standard deviations of the features $X_R$ and $X_I$ are equal for hypothesis $H_j$, i.e. $\sigma_{Rj} = \sigma_{Ij} = \sigma_j$, the likelihood ratio in Equation (22.11) is the PSD (except for a constant multiplier and addendum, which are nonessential for recognition). Thus, in this case, the use of the PSD is optimal.

Now we estimate and compare the recognition effectiveness of the proposed approach and the Hartley, cosine and PSD approaches. We use an information criterion of recognition effectiveness, Fisher's criterion [40]. The following Fisher's criteria are employed for comparison:

$$F_o = \frac{[m(S_o/H_1) - m(S_o/H_0)]^2}{\sigma^2(S_o/H_1) + \sigma^2(S_o/H_0)} \tag{22.15}$$

$$F_H = \frac{[m(S_H/H_1) - m(S_H/H_0)]^2}{\sigma^2(S_H/H_1) + \sigma^2(S_H/H_0)} \tag{22.16}$$

$$F_C = \frac{[m(S_C/H_1) - m(S_C/H_0)]^2}{\sigma^2(S_C/H_1) + \sigma^2(S_C/H_0)} \tag{22.17}$$

$$F_{PSD} = \frac{[m(S_{PSD}/H_1) - m(S_{PSD}/H_0)]^2}{\sigma^2(S_{PSD}/H_1) + \sigma^2(S_{PSD}/H_0)} \tag{22.18}$$

where $m$ and $\sigma^2$ are the mean and variance operators respectively.

Using Equations (22.1)–(22.3), and (22.6)–(22.9), after transformations, we obtain:

$$F_o = 1 - \frac{2}{b + \frac{1}{b}} \tag{22.19}$$

$$G_H = G_C = 2 \tag{22.20}$$

$$G_{PSD} = \frac{2(a^2 + 2r^2a + 1)}{(a+1)^2} \tag{22.21}$$

where $G_H = \frac{F_o}{F_H}$, $G_C = \frac{F_o}{F_C}$, $G_{PSD} = \frac{F_o}{F_{PSD}}$; $G_H, G_C$ and $G_{PSD}$ are the effectiveness gains provided by the proposed approach in comparison with the Hartley, cosine and PSD approaches respectively; $b = \frac{\sigma_{x1}^2}{\sigma_{x0}^2}$, parameter $b$ characterizes the difference of the signal variances, parameter $a$ characterizes the difference between variances of the features in Equations (22.1) and (22.2). For the identical normalized autocovariance function of the signals for hypothesis $H_j$ we obtain from [39] the following equation: $a = \frac{\sigma_{R0}}{\sigma_{I0}} = \frac{\sigma_{R1}}{\sigma_{I1}}$.

We find from Equations (22.19) and (22.20) that the recognition effectiveness of the proposed approach, as well as the recognition effectiveness of the both Hartley and cosine approaches, depends only upon the difference in the signal variances. The recognition effectiveness of the proposed approach does not depend on the correlation coefficient between the features in Equations (22.11) and (22.2) and the difference in the features' variances, because the generalized approach takes into account the correlation coefficient between the features and the difference in the features' variances.

It can be seen from Equation (22.20) that use of the proposed approach provides essential constant recognition effectiveness gain in comparison with the Hartley and cosine approaches for arbitrary values of correlation coefficient between the features in Equations (22.1) and (22.2), signal variances

and feature variances. This indicates that the features in Equations (22.1) and (22.2) have essentially better recognition effectiveness than the Hartley and cosine-based features.

We find from Equations (22.19) and (22.21) that the recognition effectiveness of the PSD approach depends upon the difference in the signal variances and also upon the correlation coefficient between the features and parameter $a$; the recognition effectiveness of the PSD approach decreases as the correlation coefficient departs from zero and as parameter $a$ departs from unity, because the PSD approach does not take into account the correlation coefficient between the short-time Fourier components in Equations (22.1) and (22.2) and the variance difference between these components.

It can be seen from Equations (22.19) and (22.21) that the use of the generalized approach provides a recognition effectiveness gain $G_{\mathrm{PSD}}$ in comparison with the PSD approach for arbitrary values of the correlation coefficient between the features and parameter $a$ (except if, simultaneously, $r = 0$ and $a = 1$). The effectiveness gain depends upon nondimensional characteristics (i.e. parameter $a$ and correlation coefficient), and not on the dimensional statistical characteristics of the features (i.e. conditional feature variances).

When the correlation coefficient between the features equals zero (i.e. $r = 0$), Equation (22.21) coincides with the appropriate expression in [1]. The dependencies between the gain $G_{\mathrm{PSD}}$ and parameters $r$ and $a$ are shown in Figures 22.1(a) and (b).

We find from Equation (22.21) and Figure 22.1 that:

1. The effectiveness gain in Equation (22.21) increases as the correlation coefficient $r$ departs from zero for arbitrary values of the parameter $a$; for $a = 1$ (i.e. when the variances of the features in Equations (22.1) and (22.2) are equal), the gain can be presented as $G_{\mathrm{PSD}} = r^2 + 1$.
2. The effectiveness gain in Equation (22.21) increases as the parameter $a$ departs from unity for arbitrary values of the correlation coefficient.

Now we estimate the effect of the approach generalization, i.e. the effect of accounting for the feature covariance, on the recognition effectiveness of features in Equations (22.1) and (22.2) while estimating the likelihood ratio.

We use the relative effectiveness criterion:

$$f = \frac{F_{\mathrm{o}}}{F_1} \tag{22.22}$$

where

$$F_1 = \frac{[m(S_1/H_1) - m(S_1/H_{\mathrm{o}})]^2}{\sigma^2(S_1/H_1) + \sigma^2(S_1/H_{\mathrm{o}})} \tag{22.23}$$

$$S_1 = \ln \frac{W(X_{\mathrm{R}}/H_1)W(X_{\mathrm{I}}/H_1)}{W(X_{\mathrm{R}}/H_0)W(X_{\mathrm{I}}/H_0)}, \; S_1 = A_1 X_{\mathrm{R}}^2 + B_1 X_{\mathrm{I}}^2 + D$$

$$A_1 = A(r = 0), \; B_1 = B(r = 0)$$

$S_1$ is the likelihood ratio without taking into account feature covariance, $F_1$ is the Fisher criterion of the likelihood ratio $S_1$.

By using Equations (22.1), (22.2), (22.11), (22.19), (22.22) and (22.23) we obtain, after transformations:

$$f = r^2 + 1 \tag{22.24}$$

From Equations (22.22) and (22.24) we can conclude that the approach generalization improves the recognition effectiveness. The relative effectiveness criterion $f$ increases as the correlation coefficient departs from zero.

Now we consider and discuss how the recognition is accomplished in a noisy environment. We consider the above recognition on a background of additive stationary Gaussian narrowband centered uncorrelated interference $y(t)$. The normalized autocovariance function of the interference is identical to the normalized autocovariance function of the signals.

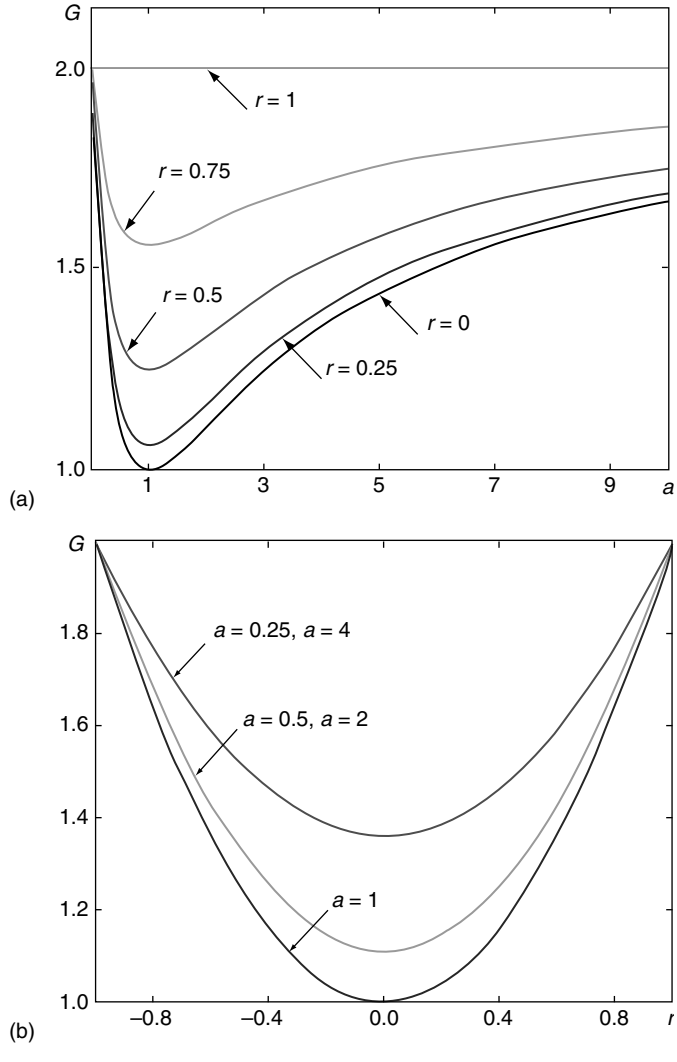



**Figure 22.1** Variation of the effectiveness gain with (a) correlation of coefficient r between features in Equations (22.1) and (22.2); (b) parameter a, which characterizes the difference between variances of features in Equations (22.1) and (22.2).

We utilize the real and imaginary components of the Fourier transform as recognition features:

$$X_{\mathrm{R}y}(\omega_x, t_1) = \int_0^{t_1} [x(t) + y(t)] \cos \omega_x t \mathrm{d}t \tag{22.25}$$

$$X_{\mathrm{I}y}(\omega_x, t_1) = \int_0^{t_1} [x(t) + y(t)] \sin \omega_x t \mathrm{d}t \tag{22.26}$$

The Hartley and cosine-based features can be written in the form:

$$X_{\mathrm{H}y}(\omega_x, t_1) = \int_0^{t_1} [x(t) + y(t)](\cos \omega_x t + \sin \omega_x t)\, \mathrm{d}t = X_{\mathrm{R}y} + X_{\mathrm{I}y} \tag{22.27}$$

$$X_{\mathrm{C}y}(\omega_x, t_1) = \int_0^{t_1} [x(t) + y(t)] \cos \omega_x t\, \mathrm{d}t = X_{\mathrm{R}y} \tag{22.28}$$

Using the above methodology, we find, after transformations, the final results:

$$F_{\mathrm{o}y} = 1 - \frac{2}{c + \dfrac{1}{c}} \tag{22.29}$$

where

$$G_{\mathrm{H}y} = G_{\mathrm{C}y} = 2 \tag{22.30}$$

$$G_{\mathrm{PSD}y} = \frac{2(a^2 + 2r^2 a + 1)}{(a+1)^2} \tag{22.31}$$

$$c = \frac{bq_0^2 + 1}{q_0^2 + 1} \tag{22.32}$$

$G_{\mathrm{H}y} = \dfrac{F_{\mathrm{o}y}}{F_{\mathrm{H}y}}$, $G_{\mathrm{C}y} = \dfrac{F_{\mathrm{o}y}}{F_{\mathrm{C}y}}$, $G_{\mathrm{PSD}y} = \dfrac{F_{\mathrm{o}y}}{F_{\mathrm{PSD}y}}$, $F_{\mathrm{o}y}$, $F_{\mathrm{H}y}$ and $F_{\mathrm{C}y}$ are the Fisher criteria for the likelihood ratios of features in Equations (22.25), (22.26), (22.27) and (22.28) respectively; $G_{\mathrm{H}y,} G_{\mathrm{C}y}$ and $G_{\mathrm{PSD}y}$ are the effectiveness gains provided by the proposed approach in comparison with the Hartley approach,

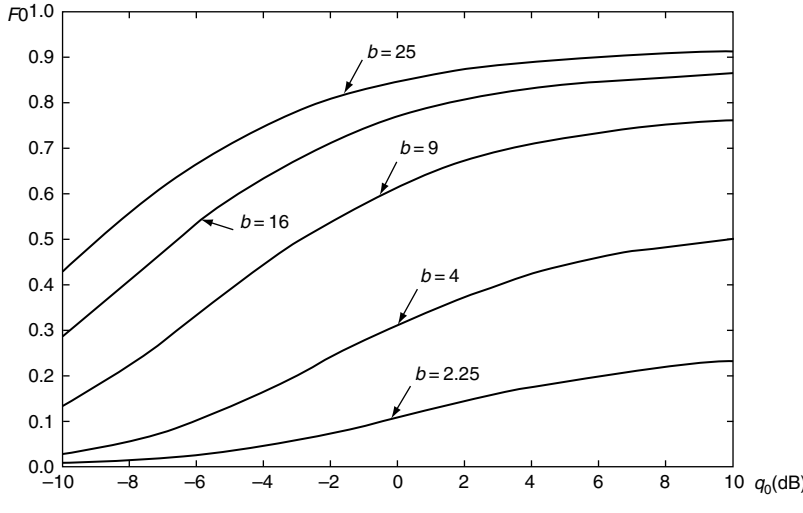



**Figure 22.2** Variation of the Fisher criterion Equation (22.29) for the proposed features with the signal/noise ratio.

cosine approach and PSD approach respectively in the considered noisy environment; $q_0 = \dfrac{\sigma_{x0}}{\sigma_y}$; $q_0$ is the signal/noise ratio for the hypothesis $H_0$; $\sigma_y^2$ is the interference variance.

When the level of interference is relatively low, e.g. at $q_0 \geq 10$ dB, the criterion of Equation (22.29) coincides with that in Equation (22.19). The dependency between the Fisher criterion Equation (22.29) and the signal/noise ratio $q_0$ is shown in Figure 22.2.

We find from Equations (22.29)–(22.32) and Figure 22.2 that the recognition effectiveness of the proposed approach, as well as the recognition effectiveness of the Hartley approach, depends only on the difference of the signal variances and the signal/noise ratio.

It can be seen from Equations (22.29)–(22.32) and Figure 22.2 that the effectiveness of the proposed approach and the Hartley, cosine and PSD approaches decreases with decrements of the signal/noise ratio $q_0$ (e.g. increments of the noise variance) for arbitrary values of the parameter $b$; however, the use of the proposed approach in the considered noisy environment provides the same recognition effectiveness gain (see Equations (22.30) and (22.31) as in the case without a noisy environment.

## 3. Application

We apply the generalized approach to the intelligent recognition of object damping and fatigue. We consider the two-class diagnostics of the object damping ratio $\zeta_j$ for hypothesis $H_j$, using the forced oscillation diagnostic method [2,41]. The method, which consists of exciting tested objects into resonant oscillations and recognition, is based on the Fourier transform of the vibration resonant oscillations. The basis of the method is the fact that damping ratio changes will modify the parameters of the vibration resonant oscillations.

The differential equation of motion for the tested object – a single degree of freedom linear oscillator under white Gaussian noise stationary excitation – is described as:

$$\ddot{x} + 2\zeta_j \omega_n \dot{x} + \omega_n x = A(t)\cos\varphi(t) \tag{22.33}$$

where $x$ is the object displacement, $\zeta_j = \dfrac{c_j}{2\sqrt{km}}$; $m, c_j, k$ are the object mass, damping and stiffness respectively, $\omega_n = \sqrt{k/m}$, $\omega_n$ is the circular natural frequency, $A(t) = \dfrac{A_i(t)}{m}$, $A(t)$ and $A_i(t)$ are the normalized and unnormalized random Rayleigh envelopes of the Gaussian excitation, $\varphi(t)$ is the random phase, which is uniformly distributed in the range $(0, 2\pi)$.

From Equation (22.33), we obtain that the vibration resonant oscillations are stationary Gaussian signals with different variances for hypothesis $H_j$ and identical normalized autocovariance functions. Therefore, the diagnostic under consideration is the two-class intelligent recognition of the stationary Gaussian signals with different variances for hypothesis $H_j$ and identical normalized autocovariance functions. The recognition information is contained in the short-time Fourier transform of the resonant oscillations at the resonant frequency. We employ the following recognition (diagnostic) features:

- the real and imaginary components of the short-time Fourier transform of the resonant oscillations at the resonant frequency, taking into account the covariance between features;
- the PSD of the resonant oscillations at the resonant frequency.

We undertake computer-aided simulation using Simulink and the Monte-Carlo procedure. The *simulation parameters* are: $\zeta_0 = 0.095$, $\zeta_1 = 0.1$ for hypotheses $H_0$ and $H_1$; the duration of the steady-state resonant oscillations is $T = 0.625$ s; the circular natural frequency is $\omega_n = 2\pi\ 20$ rad/s; the sampling frequency is 128 Hz, and the value $b = 1.3 \cdot 10^3$ is used for the parameter of the pdf of the Rayleigh envelope. The number of randomly simulated samples is 5000 for every hypothesis.

The estimate of the correlation coefficient between features in Equations (22.1) and (22.2) is nonzero: $\hat{r}_{\mathrm{RI}} = 0.12$; the estimate of the parameter $a$ is 0.29; the estimate of the effectiveness gain is $\hat{G}_{\mathrm{PSD}} = 1.24$.

Using Equation (22.5), $r_x(\tau) = \delta(\tau)$ for white noise and $N = 1.25$, we obtain that the theoretical correlation coefficient between features in Equations (22.1) and (22.2) is also nonzero: $r_{\text{RI}} = 0.13$, where $\delta(\tau)$ is the Dirac function, $N$ is the number of periods related to the resonant frequency on the signal duration $T$. Using Equation (22.21), we find the theoretical effectiveness gain $G_{\text{PSD}} = 1.31$. One can see that the simulation results match the theoretical results.

We consider the experimental fatigue crack diagnostics of objects using the forced oscillation method. The nonlinear equations of a cracked object motion under white Gaussian noise stationary excitation can be written as follows [2,41]:

$$\begin{cases} \ddot{x} + 2\zeta_{\text{S}}\omega_{\text{S}}\dot{x} + \omega_{\text{S}}x = A(t)\cos\varphi, & x \geq 0 \\ \ddot{x} + 2\zeta_{\text{C}}\omega_{\text{C}}\dot{x} + \omega_{\text{C}}x = A(t)\cos\varphi, & x < 0 \end{cases}$$

where $\zeta_{\text{S}} = \dfrac{c}{2\sqrt{k_{\text{S}}m}}$, $\zeta_{\text{C}} = \dfrac{c}{2\sqrt{k_{\text{C}}m}}$, $\omega_{\text{S}} = \sqrt{\dfrac{k_{\text{S}}}{m}}$, $\omega_{\text{C}} = \sqrt{\dfrac{k_{\text{C}}}{m}}$, $k_{\text{S}}$ and $k_{\text{C}}$ are the stiffnesses at tension and compression, $\zeta_{\text{S}}$ and $\zeta_{\text{C}}$ are the damping ratios at tension and compression.

At compression, the crack is closed and the object behaves like a continuum; therefore, the stiffness is the same as that of the object without a crack, i.e. $k_{\text{C}} = k$. At tension, the crack is opened and the object is discontinuous; therefore, the stiffness decreases with the quantity $\Delta k = k_{\text{C}} - k_{\text{S}}$, $k^* = \dfrac{\Delta k}{k}$, $k^*$ is the stiffness ratio. A relative crack size characterizes the stiffness ratio [2,41,42]. The basis for using this method lies in the fact that the level of the object nonlinearity changes with the crack size [2,41,42].

We consider the two-class diagnostics of the object stiffness ratio: $k^* = k_j^*$ for class $H_j$, $j = 0, 1$. This consideration is generic because it is independent of the correlation between the stiffness ratio and relative crack size.

We employ the following recognition (diagnostic) features:

- the real and imaginary components of the short-time Fourier transform of the higher harmonic of the object resonant oscillations, taking into account the covariance between features;
- the PSD of the higher harmonic of the object resonant oscillations.

Experimental investigation was undertaken with uncracked and cracked turbine blades from an aircraft engine. The flexural resonant blade vibrations were generated using a shaker. Acoustics radiated from the blades were received using a microphone located near the blades at a distance of 1 m. The duration of the steady state blade resonant oscillations was $t_1 = 2.3$ s; the sampling frequency was 43 478 Hz, the leakage parameter was 0.4 and the higher harmonic number was 2.

We used for comparison the effectiveness gain $A$, the ratio of the 95 % upper confidence limit $P_{\text{PSD}}$ of the total error probability for the PSD-based feature and the proposed features, $P_{\text{NEW}}$. The obtained gain estimate was $\hat{A} = 1.7$.

Thus, the use of the proposed generalized approach provides an effectiveness gain in comparison with the PSD approach for the application under consideration.

## 4. Conclusions

1. Generalization of the feature representation approach [3,4] has been proposed and investigated. The generalized approach consists of simultaneously using two new recognition features – the real and imaginary components of the Fourier transform – *taking into account covariance between these features*. It was shown that the generalization (i.e. accounting for the covariance between these features) improves the recognition effectiveness. The relative effectiveness criterion increases as the correlation coefficient departs from zero.

2. The covariance and the correlation coefficient between the proposed features, i.e. *short-time* Fourier components, were obtained for the first time for *arbitrary* stationary signals. The obtained generic expressions take into account the significant parameters: the signal normalized autocovariance function, signal variance, signal duration and Fourier transform frequency.
3. Recognition of the Gaussian signals was considered using the generalized approach. A comparison of the recognition effectiveness of the generalized approach and the Hartley, cosine and PSD approaches was carried out.
4. Comparing the generalized approach to the Hartley approach shows that:

   — the recognition effectiveness of the proposed approach, as well as the recognition effectiveness of the Hartley approach, depends only on the difference of the signal variances and does not depend on the correlation coefficient between the new features and the features' variances;
   — the Hartley approach is not an optimal feature representation approach and does not represent even a particular case of the proposed approach;
   — the use of the proposed approach provides an essential constant effectiveness gain in comparison with the Hartley approach for arbitrary values of the correlation coefficient between these features and signal variances.

5. Comparing the generalized approach to the cosine approach shows that:

   — recognition effectiveness of the proposed approach, as well as recognition effectiveness of the cosine approach, depends only on the difference of the signal variances and does not depend on the features' variances;
   — the cosine approach is not an optimal feature representation approach and does not represent even a particular case of the proposed approach;
   — the use of the proposed approach provides an essential constant effectiveness gain in comparison with the cosine approach for arbitrary values of the correlation coefficient between features and signal variances.

6. Comparing the generalized approach to the PSD approach shows that:

   — the PSD approach generally is not an optimal feature representation approach and represents only a particular case of the generalized approach;
   — the use of the PSD approach is optimal only if simultaneously: the correlation coefficient between Fourier components is equal to zero and the standard deviations of the Fourier components are equal;
   — the use of the generalized approach provides an effectiveness gain in comparison with the PSD approach for arbitrary values of the correlation coefficient between new features and the difference between feature variances (except for the above-mentioned case);
   — the effectiveness gain increases as the correlation coefficient departs from zero and as the parameter that characterizes the difference between variances of the features departs from unity.

7. Comparing the generalized approach to the Hartley, cosine and PSD approaches in a noisy environment shows that the recognition effectiveness of the proposed approach and the Hartley, cosine and PSD approaches decreases with decrements of the signal/noise ratio (e.g. increments of the noise variance) for arbitrary values of the signal difference. However, the use of the proposed approach provides the same essential effectiveness gain in comparison with the Hartley, cosine and PSD approaches as in the case without a noisy environment.
8. Application of the generalized approach was considered for vibration diagnostics of object damping and fatigue. The simulation and experimental results agree with the theoretical results.

Thus, we recommend considering simultaneous usage of the Fourier components, taking into account covariance between these components, as the most generic feature representation approach.

## Acknowledgment

The authors are very grateful to Mr Petrunin for assistance with experimental validation.

## References

[1] Gelman, L. and Braun, S. "The optimal usage of the Fourier transform for pattern recognition," *Mechanical Systems and Signal Processing*, **15**(3), pp. 641–645, 2001.
[2] Gelman, L. and Petrunin, I. "New generic optimal approach for vibroacoustical diagnostics and prognostics," *Proceedings of the National Symposium on Acoustics*, India, **2**, pp. 10–21, 2001.
[3] Alam, M. and Thompson, B (Eds) *Selected Papers on Optical Pattern Recognition Using Joint Transform Correlation*, SPIE International Society for Optical Engineering, 1999.
[4] Arsenault, H., Szoplik, T. and Macukow, B. *Optical Data Processing*, Academic Press, 1989.
[5] Burdin, V., Ghorbel, F. and deBougrenet de la Tocnaye, J. "A three-dimensional primitive extraction of long bones obtained from high-dimensional Fourier descriptors," *Pattern Recognition Letters*, **13**, pp. 213–217, 1992.
[6] Duffieux, P. *Fourier Transform and its Applications to Optics*, John Wiley & Sons, Inc., New York, 1983.
[7] Fukushima, S., Soma, T., Hayashi, K. and Akasaka, Y. "Approaches to the computerized diagnosis of stomach radiograms," *Proceedings of the Third World Conference on Medical Informatics*, Holland, pp. 769–773, 1980.
[8] Gaskill, J. *Linear Systems, Fourier Transforms and Optics*, John Wiley & Sons, Inc., New York, 1978.
[9] Goodman, J. *Introduction to Fourier Optics*, McGraw-Hill Higher Education, 1996.
[10] Granlund, G. "Fourier preprocessing for hand print character recognition," *IEEE Transactions on Computers*, **C-21** (2), pp. 195–201, 1972.
[11] Kauppinen, H., Seppanen, T. and Pietikainen, M. "An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17** (2), pp. 201–206, 1995.
[12] Liang, J. and Clarson, V. "A new approach to classification of brainwaves," *Pattern Recognition*, **22**, pp. 767–774, 1989.
[13] Linfoot, E. *Fourier Methods in Optical Image Evaluation*, Butterworth–Heinemann, 1966.
[14] Moharir, P. *Pattern-Recognition Transforms*, John Wiley & Sons, Inc., New York, 1993.
[15] Oirrak, A., Daoudi, M. and Aboutajdine, D. "Estimation of general 2D affine motion using Fourier descriptors," *Pattern Recognition*, **35**, pp. 223–228, 2002.
[16] Oppenheim, A. and Lim, J. "The importance of phase in signals," *Proceedings of the IEEE*, **69**, pp. 529–541, 1981.
[17] Ozaktas, H., Kutay, M. A. and Zalevsky, Z. *Fractional Fourier Transform: With Applications in Optics and Signal Processing*, John Wiley & Sons, Ltd, Chichester, 2001.
[18] Persoon, E. and Fu, K. "Shape discrimination using Fourier descriptors," *IEEE Transactions on Systems, Man and Cybernetics*, **SMC-7** (3), pp. 170–179, 1977.
[19] Persoon, E. and Fu, K. "Shape discrimination using Fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8** (8), pp. 388–397, 1986.
[20] Pinkowski, B. "Principal component analysis of speech spectrogram images," *Pattern Recognition*, **30**, pp. 777–787, 1997.
[21] Pinkowski, B. and Finnegan-Green, J. "Computer imaging features for classifying semivowels in speech spectrograms," *Journal of the Acoustical Society of America*, **99**, pp. 2496–2497, 1996.
[22] Pinkowski, B. "Computer imaging strategies for sound spectrograms," *Proceedings of the International Conference on DSP Applications and Technology*, DSP Associates, pp. 1107–1111, 1995.
[23] Pinkowski, B. "Robust Fourier descriptors for characterizing amplitude modulated waveform shapes," *Journal of the Acoustical Society of America*, **95**, pp. 3419–3423, 1994.
[24] Pinkowski, B. "Multiscale Fourier descriptors for classifying semivowels in spectrograms," *Pattern Recognition*, **26**, pp. 1593–1602, 1993.
[25] Poppelbaum, W., Faiman, M., Casasent, D. and Sabd, D. "On-line Fourier transform of video images," *Proceedings of the IEEE*, **56** (10), pp. 1744–1746, 1968.

[26] Price, C., Snyder, W. and Rajala, S. "Computer tracking of moving objects using a Fourier domain filter based on a model of the human visual system," *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing*, Dallas, USA, pp. 561–564, 1981.

[27] Reeves, A., Prokop, R., Andrews, S. and Kuhl, F. "Three-dimensional shape analysis using moments and Fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**, pp. 937–943, 1988.

[28] Reynolds, G., Thompson, B. and DeVelis, J. *The New Physical Optics Notebook: Tutorials in Fourier Optics*, SPIE International Society for Optical Engineering, 1989.

[29] Shridhar, M. and Badreldin, A. "High accuracy character recognition algorithm using Fourier and topological descriptors," *Pattern Recognition*, **17**, pp. 515–524, 1984.

[30] Wallace, T. and Mitchell, O. "Analysis of three-dimensional movement using Fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2**, pp. 583–588, 1980.

[31] Wilson, R. *Fourier Series and Optical Transform Techniques in Contemporary Optics: An Introduction*, John Wiley & Sons, Inc., New York, 1995.

[32] Wu, M. and Sheu, T. "Representation of 3D surfaces by two-variable Fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20** (8), pp. 858–863, 1998.

[33] Zahn, C. and Roskies, R. "Fourier descriptors for plane closed curves," *IEEE Transactions on Computers*, **C-21** (3), pp. 269–281, 1972.

[34] Bilmes, J.A. "Maximum mutual information based reduction strategies for cross-correlation based joint distributional modeling," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*. Seattle, USA, pp. 469–472, 1998.

[35] Bracewell, R. N. "Assessing the Hartley Transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, **38**, pp. 2174–2176, 1990.

[36] Devijver, P.A. and Kittler, J. *Pattern Recognition: A Statistical Approach*, Prentice Hall, 1982.

[37] Jenkins, G. M. and Watts, D. G. *Spectral Analysis and its Applications*, Holden-Day, 1968.

[38] Hsu, Y. S., Prum, S., Kagel, J. and Andrews, H. "Pattern recognition experiments in the Mandala/Cosine Domain," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5** (5), pp. 512–520, 1983.

[39] Gelman, L. and Sadovaya, V. "Optimization of the resolving power of a spectrum analyzer when detecting narrowband signals," *Telecommunications and Radio Engineering*, **35** (11), pp. 94–96, 1980.

[40] Young, T.Y. and Fu, K.S. *Handbook of Pattern Recognition and Image Processing*, Academic Press, Inc., 1986.

[41] Dimarogonas, A. "Vibration of cracked structures: a state of the art review," *Engineering Fracture Mechanics*, **55** (5), pp. 831–857, 1996.

[42] Gelman, L. and Gorpinich, S. "Non-Linear vibroacoustical free oscillation method for crack detection and evaluation," *Mechanical Systems and Signal Processing*, **14**(3), pp. 343–351, 2000.

# 23

# Conceptual Data Classification: Application for Knowledge Extraction

**Ahmed Hasnah**
**Ali Jaoua**
**Jihad Jaam**
Department of Computer Science, University of Qatar P.O.Box 2713, Doha, Qatar

*Formal Concept Analysis (FCA) offers a strong background for data classification. FCA is increasingly applied in conceptual clustering, data analysis, information retrieval and knowledge discovery. In this chapter, we present an approximate algorithm for the coverage of a binary context by a minimal number of optimal concepts. We notice that optimal concepts seem to help in discovering the main data features. For that reason, they have been used several times to successfully extract knowledge from data in the context of supervised learning. The proposed algorithm has also been used for several applications, such as for discovering entities from an instance of a relational database, simplification of software architecture or automatic text summarization. Experimentation on several cases proved that optimal concepts exhibit the main invariant data structures.*

## 1. Introduction

Is it not normal to expect that human intelligence organizes data in a uniform and universal way? The reason is that our natural and biological thinking structure is mostly invariant. The purpose of such a thinking process is to understand and learn from data how to recognize similar objects or situations, and create new objects in an incremental way. As a matter of fact, by analogy, most 'intelligent' information retrieval methods need to realize data classification, and minimization of its representation in memory, in an incremental way. Classification means pattern generation and recognition. Formal Concept Analysis (FCA) offers a simple, original, uniform and mathematically well-defined method for data clustering. A pattern is associated with a formal concept (i.e. a set of objects sharing the maximum number of properties). In this chapter, we minimize information representation by selecting

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

only 'optimal concepts'. We assume that data may be converted to a binary relation as a subset of the product of a set of objects and a set of properties. This hypothesis does not represent a strong constraint, because we can see that most numerical data may be mapped to a binary relation, with some approximation. In this chapter, we defend the idea that coverage of a binary context with a minimal number of conceptual clusters offers a base for optimal pattern generation and recognition [1]. We defend the idea that while we are thinking, we incrementally optimize the context space storage. We have generally different possible concepts for data coverage. Which one is the best? In recent years, we applied the idea of minimal conceptual coverage of a binary relation to supervised learning and it gave us defendable results with respect to other known methods in terms of error rate [2–7]. We also applied it for automatic entity extraction from a database. As a last important application, we used it for software restructuring by minimizing its complexity. Because of the huge amount of data contained in most existing documents and databases, it becomes important to find a priority order for concept selections to enable users to find pertinent information first. For that reason, we also exploited these patterns (i.e. concepts) for text summarization combined with a method for assessing word similarity [8]. In this chapter, we give all the steps of these conceptual methods and illustrate them with significant results.

In the second section, we present the mathematical foundation of conceptual analysis. In the third section, we give a polynomial approximate algorithm for the NP-complete problem of binary context coverage with a minimal number of optimal concepts. In the fourth section, we explain how to apply the idea of the optimal concept (also called the optimal rectangle) for knowledge extraction. We give a synthesis discussion about the following applications: supervised learning, entity extraction from an instance of a database, minimization of the complexity of software and discovering the main groups of users communicating through one or different servers.

## 2. Mathematical Foundations

Among the mathematical theories found recently with important applications in computer science, lattice theory has a specific place for data organization, information engineering, data mining and reasoning. It may be considered as the mathematical tool that unifies data and knowledge, also information retrieval and reasoning [9–13]. In this section, we define a binary context, a formal concept and the lattice of concepts associated with the binary context.

### *2.1 Definition of a Binary Context*

A binary context (or binary relation) is a subset of the product of two sets O (set of objects) and P (set of properties).

**Example 1** [10]:

$$O = \{\text{Leech, Bream, Frog, Dog, Spike-weed, Reed, Bean, Maize}\}$$

and

$$P = \{a, b, c, d, e, f, g, h, i\}$$

where O is a set of some living things, and P the set of the following properties: a = needs water; b = lives in water; c = lives on land; d = needs chlorophyll to produce food; e = is two seed leaves; f = one seed leaf; g = Can move around; h = has limbs; i = suckles its offspring. A binary context R may be defined by Table 23.1.

**Table 23.1** An example of a binary context R.

| | | a | b | c | d | e | f | g | h | i |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Leech | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 2 | Bream | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 3 | Frog | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | Dog | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 5 | Spike-weed | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 6 | Reed | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 7 | Bean | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | Maize | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

Let $f$ be a function from the powerset of the set of objects O (i.e. $2^0$) to the powerset of the set of properties P (i.e. $2^P$), such that:

$$f(\mathrm{A}) = \{m | \forall g \in \mathrm{A} \Rightarrow (g, m) \in \mathrm{R}\} \tag{23.1}$$

$f(\mathrm{A})$ is the set of all properties shared by all objects of A (subset of O) with respect to the context R. Let $g$ be a function from $2^P$ to $2^O$, such that:

$$g(\mathrm{B}) = \{g | \forall m \in \mathrm{B} \Rightarrow (g, m) \in \mathrm{R}\} \tag{23.2}$$

$g(\mathrm{B})$ is the set of objects sharing all the properties B (subset of P) with respect to the binary context R. We also define closure$(\mathrm{A}) = g(f(\mathrm{A})) = \mathrm{A}'$, and closure$(\mathrm{B}) = f(g(\mathrm{B})) = \mathrm{B}'$.

The meaning of $\mathrm{A}'$ is that a set of objects A shares the same set of properties $f(\mathrm{A})$ with other objects $(\mathrm{A}' - \mathrm{A})$, relative to the context R. $\mathrm{A}'$ is the maximal set of objects sharing the same properties as objects A. In Example 1, if A = {Leech, Bream, Frog, Spike-weed} then $\mathrm{A}'$ = {Leech, Bream, Frog, Spike-weed, Reed}. This means that the shared properties a and b of living things in A, are also shared by a reed, the only element in $\mathrm{A}' - \mathrm{A}$. The meaning of $\mathrm{B}'$ is that if an object $x$ of the context R verifies properties B, then $x$ also verifies some number of additional properties $(\mathrm{B}' - \mathrm{B})$. $\mathrm{B}'$ is the maximal set of properties shared by all objects verifying properties B. In Example 1, if B = {a, h}, then $\mathrm{B}'$ = {a, h, g}. This means that any animal that needs water (a) and has limbs (h), can move around (g). For each subset B, we may create an association rule $\mathrm{B} \rightarrow \mathrm{B}' - \mathrm{B}$. The number of these rules depends on the binary context R. In [10], we find different algorithms for extracting the minimal set of such association rules.

## 2.2 Definition of a Formal Concept

A formal concept of a binary context is the pair (A,B), such that $f(\mathrm{A}) = \mathrm{B}$ and $g(\mathrm{B}) = \mathrm{A}$. We call A the extent and B the intent of the concept (A,B). If $(\mathrm{A}_1, \mathrm{B}_1)$ and $(\mathrm{A}_2, \mathrm{B}_2)$ are two concepts, $(\mathrm{A}_1, \mathrm{B}_1)$ is called a subconcept of $(\mathrm{A}_2, \mathrm{B}_2)$, provided that $\mathrm{A}_1 \subseteq \mathrm{A}_2, \mathrm{B}_2 \subseteq \mathrm{B}_1$. In this case, $(\mathrm{A}_2, \mathrm{B}_2)$ is a superconcept of $(\mathrm{A}_1, \mathrm{B}_1)$ and it is written $(\mathrm{A}_1, \mathrm{B}_1) < (\mathrm{A}_2, \mathrm{B}_2)$. The relation '<' is called the hierarchical order relation of the concepts. The set of all concepts of (G, M, I) ordered in this way is called the concept lattice of the Context (G, M, I). Formal Concept Analysis (FCA) is used for deriving conceptual structures from data. These structures can be graphically represented as conceptual hierarchies, allowing the analysis of complex structures and the discovery of dependencies within the data. Formal concept analysis is based on the philosophical understanding that a concept is constituted by two parts: its extent, which consists of all objects belonging to the concept, and its intent, which

**Table 23.2** Formal context K.

| | A | B | C | D |
|---|---|---|---|---|
| G1 | 1 | 1 | 0 | 0 |
| G2 | 1 | 1 | 0 | 0 |
| G3 | 0 | 1 | 1 | 0 |
| G4 | 0 | 1 | 1 | 1 |
| G5 | 0 | 0 | 1 | 1 |

comprises all attributes shared by those objects. One of the main objectives of this method is to visualize the data in the form of concept lattices.

Let K be the formal context presented in Table 23.2. Then Figure 23.1 represents the structured set of concepts.

A formal concept has been defined and introduced by different scientific communities in the world, starting in 1948 with Riguet [14] under the name of maximal rectangle. In graph theory, a maximal bipartite graph was exploited in the thesis of Le Than in 1986 [15] in a database to introduce a new kind of dependencies called 'Iso-dependencies', independently, Jaoua *et al.* introduced difunctional dependencies as the most suitable name for iso-dependencies in a database [16]. The most recent mathematical studies about formal concept analysis have been done by Ganter and Wille. More details may be found in [9–11].

What is remarkable is that concepts are increasingly used in several areas in real-life applications: text analysis, machine learning, databases, data mining, software decomposition, reasoning and pattern recognition. A complete conjunctive query and its associated answer in a database is no more nor less than a concept (i.e. an element in a lattice of concepts). Its generality and simplicity is very attractive. We almost may find a bridge between any computer science application and concepts. Combined with other methods for mapping any kind of data into a binary context, it gives an elegant base for data mining.

## *2.3 Galois Connection*

A Galois connection is a conceptual learning structure used to extract new knowledge from an existing context (database). The context is represented by a binary relation. We can decompose the context into

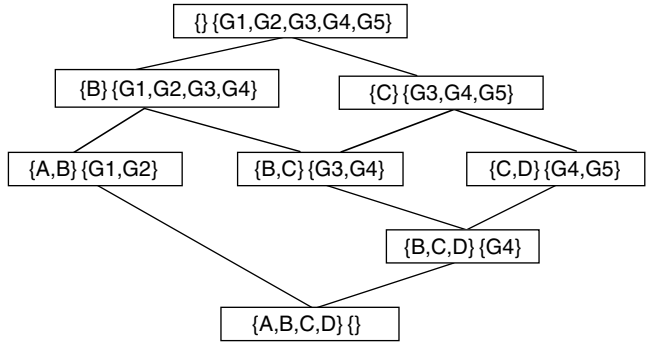



**Figure 23.1** Concept lattice of the context K.

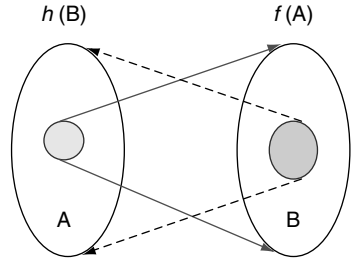



**Figure 23.2** A Galois connection.

a set of concepts and we can build a hierarchy of concepts also known as a '*Galois Lattice*'. A pair ($f$ (A), $h$ (B)) of maps is called a Galois connection *if and only if* $A \subseteq h(B) \Leftrightarrow B \subseteq f(A)$, as we can see in Figure 23.2. It is also known that $f(A) = fhf(A)$ and $h(B) = hfh(B)$.

## *2.4 Optimal Concept or Rectangle*

A binary relation can be decomposed into a minimal set of optimal rectangles (or optimal concepts).

### 2.4.1 Definition 1: Rectangle

Let R be a binary relation defined from E to F. A *rectangle* of R is a pair of sets (A,B) such that $A \subseteq E, B \subseteq F$ and $A \times B \subseteq R$. A is the *domain* of the rectangle and B is the *co-domain* or *range* [11]. A rectangle is maximal if and only if it is a concept. A binary relation can be represented by different sets of rectangles. In order to save storage space, the gain function W(R) defined in Section 2.4.2 is important for the selection of a minimal significant set of maximal rectangles representing the relation. In the next section, we use interchangeably the word 'concept' and 'maximal rectangle'. We also introduce the definition of an optimal rectangle (i.e. optimal concept). Our conviction is that intelligence does not keep in mind the whole lattice structure of concepts, but only a few concepts covering all the data context. As a matter of fact, this thinking process is efficient because it optimizes the quantity of data kept in the main memory. Here, we propose a method for such coverage. But in the future, we may propose a variety of other methods perhaps more efficient.

### 2.4.2 Definition 2: Gain in Storage Space (Economy)

The *gain* in storage space W (R) of binary relation R is given by:

$$W(R) = (r/dc)(r - (d + c)) \tag{23.3}$$

where $r$ is the cardinality of R (i.e. the number of pairs in binary relation R), $d$ is the cardinality of the domain of R and $c$ is the cardinality of the range of R.

Note that the quantity $(r/dc)$ provides a measure of the density of the relation R. The quantity $(r - (d + c))$ is a measure of the economy of information.

### 2.4.3 Definition 3: Optimal Concept

A rectangle $RE \subseteq R$, containing an element $(x, y)$ of a relation R is called *optimal* if it produces a maximal gain W(RE($x$, $y$)) with respect to other concepts containing $(x, y)$. Figure 23.3(a) presents an

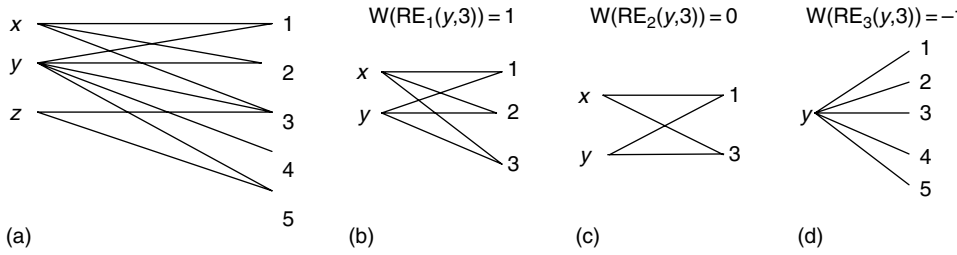



**Figure 23.3** Optimal rectangle in binary relation R. (a) Relation R; (b) $\mathrm{RE}_1(y, 3)$; (c) $\mathrm{RE}_2(y, 3)$; (d) $\mathrm{RE}_3(y, 3)$.

example of a relation R, and Figures 23.3(b), (c) and (d) represent three different rectangles containing element $(y, 3)$. The corresponding gains are 1, 0 , $-1$. Therefore, the optimal rectangle containing the pair $(y, 3)$ of R is the rectangle of Figure 23.3(b). It is easy to prove that it must always be a concept (maximal rectangle).

$$\mathrm{W}(\mathrm{RE1}(y, 3)) = (6/6)(6-(2+3)) = 1. \quad \text{Here, } r = 6, d = 2 \text{ and } c = 3$$
$$\mathrm{W}(\mathrm{RE2}(y, 3)) = (4/4)(4-(2+2)) = 0. \quad \text{Here, } r = 4, d = 2 \text{ and } c = 2.$$
$$\mathrm{W}(\mathrm{RE3}(y, 3)) = (5/5)(5-(1+5)) = -1. \quad \text{Here, } r = 5, d = 1 \text{ and } c = 5.$$

We can notice that function W applied to a concept RE is always greater than $-1$. The minimal value $-1$ is reached if $r = 1$ or $d = 1$. W is equal to 0 only when $r = d = 2$.

### 2.4.4 Elementary Relation (noted PR)

If R is a finite binary relation (i.e. a subset of $\mathrm{E} \times \mathrm{F}$, where E is a set of objects and F a set of properties) and $(a, b) \in \mathrm{R}$, then the union of rectangles containing $(a, b)$ is the elementary relation PR (i.e. subset of R) given by:

$$\mathrm{PR} = \Phi_{\mathrm{R}}(a, b) = \mathrm{I}(b.\mathrm{R}^{-1}) \circ \mathrm{R} \circ \mathrm{I}(a.\mathrm{R}) \tag{23.4}$$

where:

I is the identity relation.
$\mathrm{R}^{-1}$is the inverse relation of R (i.e the set of inverse pairs of R).
'o' refers to the relative product operator, where :

$$\mathrm{R} \circ \mathrm{R}' = \{(x, y) | \exists z : (x, z) \in \mathrm{R} \wedge (z, y) \in \mathrm{R}'\} \tag{23.5}$$

Let $\mathrm{A} \subseteq \mathrm{E}$, then we define $\mathrm{I}(\mathrm{A}) = \{(a, a) | a \in \mathrm{A}\}$.

PR is the subrelation of R, prerestricted by the antecedents of $b$ (i.e. $b.\mathrm{R}^{-1}$), and postrestricted by the set of images of $a$ (i.e. $a.\mathrm{R}$).

In the next section, we use such elementary relations PR to find a coverage of a relation by some 'minimal' number of optimal concepts. The problem is NP-complete. For that reason, we only propose an approximate solution based on a greedy method using the gain function W. Later, this algorithm has been adapted to become incremental and concurrent.

## 3. An Approximate Algorithm for Minimal Coverage of a Binary Context

The search for a set of optimal rectangles that provides a coverage of a given relation R can be made through the following steps [15]:

### Example 2

Let R be a finite binary relation between two sets as illustrated in Figure 23.4.

1. Divide the relation R into disjoint elementary relations $PR_1, PR_2, \ldots, PR_m$.
2. For each elementary relation $PR_i$, search the optimal rectangle which includes an element of $PR_i$.

If $PR_i$ is a rectangle, then it is an optimal rectangle containing $(a, b)$, else check if PR contains other elements $(X, Y)$ in the form $(a, Y)$ or $(X, b)$ by trying all the images of $a$ and all the antecedents of $b$.

The elementary relation of (1,7) as shown in Figure 23.5 is:

$$\mathrm{PR}(1,7) = \Phi_{\mathrm{R}}(1,7) = \mathrm{I}(7.\mathrm{R}^{-1}) \circ \mathrm{R} \circ \mathrm{I}(1.\mathrm{R}) \tag{23.6}$$

So we search in an iterative way the optimal rectangles of PR(1,7) which successively contain the elements (1,8),(1,9),(1,11),(2,7) and (3,7).

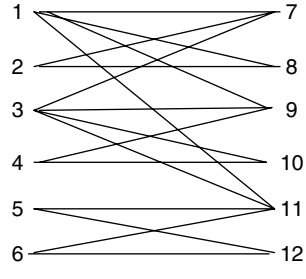



**Figure 23.4** Binary relation R for Example 2.

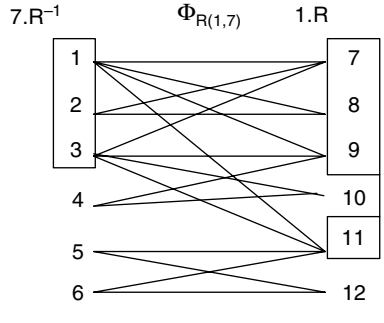



**Figure 23.5** The elementary relation of (1,7).

### *First Iteration*

From the five elementary relations of the above-mentioned elements, select the first that gives a maximal gain:

1. $PR'_{1,8} = \Phi_{PR1,7}(1, 8); W(PR'_{1,8}) = 0$
2. $PR'_{1,9} = \Phi_{PR1,7}(1, 9); W(PR'_{1,9}) = 7/8 \blacktriangleright$ selected
3. $PR'_{1,11} = \Phi_{PR1,7}(1, 11); W(PR'_{1,11}) = 7/8$
4. $PR'_{2,7} = \Phi_{PR1,7}(2, 7); W(PR'_{2,7}) = 0$
5. $PR'_{3,7} = \Phi_{PR1,7}(3, 7); W(PR'_{3,7}) = 7/9$

The selected elementary relation $PR'_{1,9}$ (Figure 23.6) is not a rectangle, so the algorithm continues on the already selected elements, i.e.(1,7) and (1,9).

### *Second Iteration*

Search now the optimal rectangles of $PR'_{1,9}$ that successively contain elements (1,8), (1,11) and (3,9). This step provides three elementary relations:

1. $PR''_{1,8} = \Phi_{PR'_{1,9}}(1, 8); W(PR''_{1,8}) = -1$
2. $PR''_{1,11} = \Phi_{PR'_{1,9}}(1, 11); W(PR''_{1,11}) = 7/8$
3. $PR''_{3,9} = \Phi_{PR'_{1,9}}(3, 9); W(PR''_{3,9}) = 1 \blacktriangleright$ selected

$PR''_{3,9}$ is a rectangle (Figure 23.7), so it is an optimal one that contains element (1,7) of R.

Figure 23.8, illustrates the iterations used to search for the optimal rectangle. In bold you can see the selected elementary relation at each level of the search tree. Each level is associated with an iteration in the proposed algorithm. The proposed algorithm is polynomial. When we find an optimal rectangle, we continue to search for a next optimal one containing another pair not already selected. Here, if we select the pair (6,12), we find at the first iteration the concept: $PR_{6,12} = \{5, 6\} \times \{11, 12\}$. Then, if we select the pair (4,10), we obtain the concept: $PR_{4,10} = \{3, 4\} \times \{9, 10\}$. Finally, if we select the pair (2,8), we obtain the concept: $PR_{2,8} = \{2, 1\} \times \{7, 8\}$. The selected coverage is composed of: $\{PR''_{3,9}, PR_{6,12}, PR_{4,10}, PR_{2,8}.\}$.

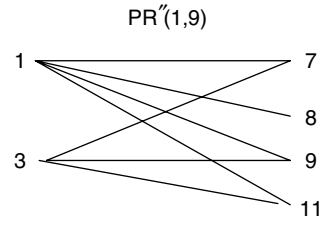



**Figure 23.6** The selected elementary relation $PR'_{1,9}$.

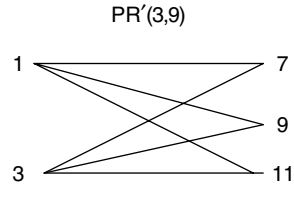



**Figure 23.7** The optimal rectangle $PR'_{3,9}$.

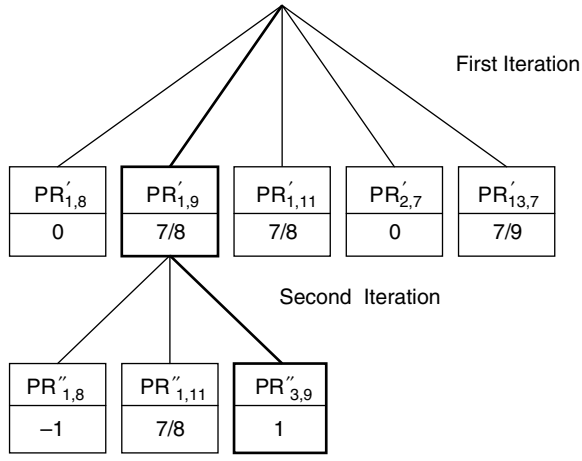



**Figure 23.8** Iterations for searching for the optimal rectangle contained in R.

```
Optimal_Rectangle (Relation R, int& s', int& w')
Problem: Determine the optimal rectangle of a binary relation R
Inputs: A binary relation R[][]
Outputs: The pair (s', w') containing an optimal rectangle in R.
Begin
Let R [m][n] be the binary relation of m objects and n properties,
Emax = 0;//   The maximum searched gain in R (W(R)) initialized to 0
For s = 0 to n − 1
   For w = 0 to m − 1
      If R[s, w]! = 0
         Then PR = I(R.w)oRoI(s.R); // calculating the elementary relation of (s,w)
            E= economy (PR);
               If E > Emax
                  Then {Emax = E;
                        Highest = PR;   // Highest is the concept of maximal gain
                        s' = s;
                        w' = w; }
               End if
      End if
   End for
End for
If Highest is not rectangle                    // r! = cd
   Then Optimal_Rectangle (Highest, s', w') //Recursive call to function
                                    //Optimal Rectangle starting from
                                 //relation Highest corresponding to the
                           //next level in the search tree
Else return the pair(s', w')
End if
End.
```

**Figure 23.9** Algorithm calculating an optimal rectangle in a binary relation R.

In Figure 23.9, we illustrate an algorithm for extracting an optimal rectangle from a binary relation (function optimal rectangle). In Figure 23.10, we illustrate a function to calculate the gain of a rectangle.

In the following section, we see that optimal concepts are used to discover the main patterns in data and that they may be used in several situations to extract a different kind of knowledge.

***Problem:*** *Determine the economy of a binary relation*
***Inputs:*** *A binary relation* R
***Outputs:*** *The economy*
***Begin***
*Let* R *[m][n] be the binary relation of m objects and n properties*
*Let r be the number of pairs in* R.
*Let c be the cardinality of domains of* R.
*Let d be the cardinality of the co-domain of* R.
*Return* $(r/(c^*d))^*(r-(c+d))$
*End.*

**Figure 23.10** Economy of a binary relation calculation.

## 4. Conceptual Knowledge Extraction from Data

Data are inherently and internally composed of related elements. We are generally able to map several kinds of data into a binary relation linking these elements to their properties or to each other. As a first example, assume that you want to analyze a group of computers. You first discover the general pattern corresponding to the maximum number of properties shared by the maximum number of computers. You then discover other subgroups of computers sharing another subset of properties, etc . . . Assume now that you receive a million web/URLs from the Internet. If you associate, for each web page, a list of keywords indexing it, then a user can identify the main features of this huge amount of web references by first extracting optimal concepts. This classification of web pages should help users in the browsing process to converge to the required documents in a shorter time. As a third example, before deciding to read a book or document, it is useful to read its abstract. For that purpose, we could decompose the document into sentences, then create a binary relation linking each sentence to nonempty words belonging to it. An optimal concept linking the maximum number of sentences to the maximum number of words (or similar words) may be used as a base for a summary extraction from the document. As a last application, assume that you have an instance of a table in a relational database model, how can we discover the entities of the database? In Section 4.1, we will explain the main ideas of supervised learning using optimal concepts. In Section 4.2, we give a method to explain how we discover entities from an instance of a database. In Section 4.3, we explain how we can find the simplest software architecture with a minimal complexity. Finally, in Section 4.4, we show that we can find the most important communicating groups in a network using optimal concepts.

### *4.1 Supervised Learning by Associating Rules to Optimal Concepts*

Assume that we start from a relation describing several objects (such as patients in a hospital, or students, or customers in a bank). Here, data elements are objects or properties. Each object is supposed to belong to a specific class. For example, for the table of patients, the class attribute may be associated with the kind of disease the patient has (heart, skin, etc.). The purpose of supervised learning is to predict the class of a new object, by using the knowledge extracted from the existing database about already classified objects. In the proposed method, we first build a binary relation corresponding to the relation between the set of objects and the set of attributes, as shown in Table 23.3. Using the algorithm described in the previous section (Figures 23.9 and 23.10), we extract the minimal coverage of optimal concepts of the binary relation R. For each concept, we create an association rule with some degree of certainty [17].

**Table 23.3** Relation R.

| | P1 | P2 | P3 | P4 | P5 | P6 | Class |
|---|---|---|---|---|---|---|---|
| O1 | 1 | 0 | 0 | 1 | 0 | 0 | C1 |
| O2 | 0 | 1 | 1 | 0 | 1 | 1 | C2 |
| O3 | 1 | 0 | 0 | 1 | 0 | 0 | C1 |
| O4 | 0 | 1 | 1 | 0 | 1 | 1 | C2 |
| O5 | 1 | 0 | 0 | 1 | 0 | 0 | C1 |

From this relation R between objects $O1, \ldots, O5$ and properties $P1, \ldots, P6$, by the algorithm of the previous section, we find the following two concepts:

$$R = \{O1, O3, O5\} \times \{P1, P4\} \cup \{O2, O4\} \times \{P2, P3, P5, P6\}$$

Because all objects contained in the first concept are in the class C1, we extract the first association rule:

$$P1 \text{ AND } P2 \text{ THEN CLASS} = C1 \text{ WITH CERTAINTY DEGREE} = 1$$

By the same means, from the second concept, we can extract the second rule:

$$P2 \text{ AND } P3 \text{ AND } P5 \text{ AND } P6 \text{ THEN CLASS} = C2 \text{ WITH CERTAINTY}$$
$$\text{DEGREE} = 1$$

From the minimal coverage, we can, in this way, extract a minimal number of rules associated with each concept. When we have to decide about the class of an object with respect to relation R, we can use these association rules to give the 'best' approximation to the predicted class of this object. However, a concept may contain objects belonging to different classes. In that case, assume that a concept $\{O2, O4\} \times \{P1, P2, P3\}$ contains only 57 % of objects belonging to class C1, then we create the rule:

$$\text{IF } P1 \text{ AND } P2 \text{ AND } P3 \text{ THEN CLASS} = C1 \text{ WITH CERTAINTY DEGREE} = 0.57$$

So, we select the class corresponding to the majority of objects in the concept. When we have to decide about the class of an object, we can deduce different alternatives, but we only take the best one, corresponding to the class that we can deduce with the highest certainty degree. Experiences realized on several public databases (such as heart disease and flower tables) have proved that the proposed approach is defendable with respect to other known methods. The learning time corresponding to rule generation is polynomial and lower than the other methods. By using an incremental approach, we have been able to improve time efficiency by only updating a few concepts in each database update.

## *4.2 Automatic Entity Extraction from an Instance of a Relational Database*

Assume that you start with the instance of a database in Table 23.4. This table links three different entities: suppliers, projects and parts. We assume that initially, we ignore these entities and would like to discover these entities automatically. We can notice that an entity is defined as a subset of attributes. In this example, we have the following attributes: {S#, SNAME, STATUS, SCITY, P#, PNAME, COLOR, WEIGHT, PCITY, J#, JNAME, JCITY, QTY }.

**Table 23.4** An instance SPJ of a relational database.

| S# | SNAME | STATUS | SCITY | P# | PNAME | COLOR | WEIGHT | PCITY | J# | JNAME | JCITY | QTY |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S1 | Smith | 20 | London | p1 | Nut | Red | 12 | London | J1 | Sorter | Paris | 200 |
| S1 | Smith | 20 | London | p1 | Nut | Red | 12 | London | J4 | Console | Athens | 700 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J1 | Sorter | Paris | 400 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J2 | Punch | Rome | 200 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J3 | Reader | Athens | 200 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J4 | Console | Athens | 500 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J5 | Collator | London | 600 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J6 | Terminal | Oslo | 400 |
| S2 | Durand | 10 | Paris | p3 | Screw | Blue | 17 | Rome | J7 | Tape | London | 800 |
| S2 | Durand | 10 | Paris | p5 | Cam | Blue | 12 | Paris | J2 | Punch | Rome | 100 |
| S3 | Dupont | 30 | Paris | p3 | Screw | Blue | 17 | Rome | J1 | Reader | Paris | 200 |
| S3 | Dupont | 30 | Paris | p4 | Screw | Red | 14 | London | J2 | Tape | Rome | 500 |
| S4 | Clark | 20 | London | p6 | Cog | Red | 19 | London | J3 | Console | Athens | 300 |
| S4 | Clark | 20 | London | p6 | Cog | Red | 19 | London | J7 | Console | London | 300 |
| S5 | Kurt | 30 | Athens | p1 | Nut | Red | 12 | London | J4 | Punch | Athens | 1000 |
| S5 | Kurt | 30 | Athens | p2 | Bolt | Green | 17 | Paris | J4 | Console | Athens | 100 |
| S5 | Kurt | 30 | Athens | p2 | Bolt | Green | 17 | Paris | J2 | Collator | Rome | 200 |
| S5 | Kurt | 30 | Athens | p3 | Screw | Blue | 17 | Rome | J4 | Console | Athens | 1200 |
| S5 | Kurt | 30 | Athens | p5 | Cam | Blue | 12 | Paris | J5 | Tape | London | 500 |
| S5 | Kurt | 30 | Athens | p5 | Cam | Blue | 12 | Paris | J4 | Console | Athens | 400 |
| S5 | Kurt | 30 | Athens | p5 | Cam | Blue | 12 | Paris | J7 | Punch | London | 100 |
| S5 | Kurt | 30 | Athens | p4 | Cam | Red | 14 | London | J4 | Console | Athens | 800 |
| S5 | Kurt | 30 | Athens | p6 | Cam | Red | 19 | London | J2 | Punch | Rome | 200 |
| S5 | Kurt | 30 | Athens | p6 | Cam | Red | 19 | London | J4 | Console | Athens | 500 |

The entity extraction algorithm [18–20] is composed of the following steps:

1. We define elementary data as a pair of (attribute, value) written as 'attribute.value'. The elementary data set is: {S#.S1, S#.S2, S#.S3, S#.S4, S#.S5, SNAME.Clark, SNAME.Kurt, SNAME.Dupont, SNAME.Smith, SNAME.Durand, STATUS.10, STATUS.20, STATUS.30, SCITY.Paris, SCITY.Athens, SCITY.London, P#.p1, P#.p2, P#.p3, P#.p4, P#.p5, P#.p6, etc …}.
2. We then create a binary relation R relating elementary data using the following definition. We say that a pair of items $(X.x, Y.y)$ belongs to binary relation R if and only if $x$ is the value for attribute $X$, and $y$ is the value for attribute $Y$, for the same row in the instance SPJ in Table 23.4. We then extract the first optimal concept $RE_{opt}$ as shown in Figure 23.11.
3. From $RE_{opt}$ we may extract two disjoint sets of attributes: set $S_{left}$, which contains the attributes which appear in $dom(RE_{opt})$, and set $S_{right}$, which contains the attributes which appear in $cod(RE_{opt})$.

$$S_{left} = \{S\#, SNAME, STATUS, SCITY, P\#, PNAME, COLOR, WEIGHT, PCITY\};$$

$$S_{right} = \{J\#, JNAME, JCITY, QUANTITY\}$$

{S#.S2, SNAME.Durand, STATUS.10, SCITY.Paris, PNAME.Screw, COLOR.Blue, JCITY.Rome, WEIGHT.17} X {J#.J1; JNAME.Sorter; JCITY.Paris, QTY.200, J#.J4, JNAME.Console, JCITY.Athens; QTY.500, J#.J2, JNAME.Punch, JCITY.Rome; J#.J3; JNAME.Reader, QTY.200, J#.J5, Collator.10, JCITY.London, QTY.600, J#.J6, JNAME.Terminal, JCITY.Oslo; J#.J7, JNAME.Tape, QTY.800};

**Figure 23.11** Optimal rectangle $RE_{opt}$ of R.

As a matter of fact, $S_{left}$ represents the two entitites Supplier and Parts, and $S_{right}$ represents the two entities Project and Quantity. Furthermore, $S_{left}$ and $S_{right}$ are also disjoint.

4. By making the projection of SPJ on the attributes of $S_{left}$, we obtain $SPJS_{left}$. Similarly, the projection of SPJ on the attributes of $S_{right}$ gives $SPJS_{right}$. Then we apply steps 1 to 3 successively on $SPJS_{left}$ and $SPJS_{right}$. The decomposition of $SPJS_{left}$ gives exactly the two predicted entities:

$$\text{Supplier} = \{\text{S\#, SNAME, STATUS, SCITY}\} \text{ and}$$

$$\text{Parts} = \{\text{P\#, PNAME, COLOR, WEIGHT, PCITY}\}$$

The decomposition of $SPJS_{right}$ gives exactly the two other predicted entities:

$$\text{Project} = \{\text{J\#, JNAME, JCITY}\} \text{and} \{\text{Quantity}\}.$$

As a matter of fact, Quantity cannot be associated with any other entity. Consequently, there is no reason to associate it with Parts, Project or Supplier.

We have done similar experiments with several instances of SPJ, with different sizes, and we have always obtained exact results. We have made other experiments with two to four attributes, and we have also obtained good results. These results may be considered excellent, since the number of all possible combinations of subsets of attributes is higher than $2^{13}$.

### *4.3 Software Architecture Development*

The question here is to derive a software with the simplest architecture possible: i.e. with minimal interactions between its components. In functional programming, we may consider a function as elementary data and include a pair of functions $(x, y)$ if and only if function $x$ calls function $y$. Then, using the algorithm of Section 3, we can restructure the software by clustering the functions associated with optimal concepts into the same subsystem [21,22]. The method could be reiterated to an upper level, using a uniform algorithm. We can also relate data to functions, then change automatically the program into an object-oriented one. When the program is already written as an object-oriented one, we can use communication between classes to create a binary relation, then using the algorithm in Section 3, we can create superclasses to obtain a better software structuring with a minimal communication between the subsystems that are identified with the derived superclasses.

### *4.4 Automatic User Classification in the Network*

Assume that an administrator of a server would like to classify the users into groups of users that communicate the most. In this case, it becomes obvious that we can use an incremental version of the algorithm in Section 3. First, we can create a binary relation between different users if they have communicated at least some number of messages per month. Second, we extract the minimum number of concepts. Each concept will identify a group of users. This incremental classification might be used for several purposes.

## 5. Conclusion

In this chapter, one can find a uniform and incremental algorithm for data organization. This method is based on formal concept analysis. Among the exponential number of concepts existing between elementary data, we only select optimal ones. We explained here how we can use these optimal concepts to extract association rules, or entities, from an instance of a database, or to find the best

architecture of a software, or to discover the main communicating groups in a network. We can also use these algorithms to generate a summary from a text, using the optimal concept in the text as the main part which is used to generate a summary [23–27]. Each concept represents a cluster of sentences sharing common indexing words. From each concept, the system extracts different parts of the text. The quality of the summaries seems to be quite acceptable. In the future, we will be able to integrate the system to abstract huge documents. This system may be integrated into search engines to filter only web pages belonging to the optimal concept associated with the binary relations linking web pages to their indexing words. It would also be interesting to use the system to filter successive concepts from search engines to only deliver web pages belonging to the main concepts from the global web pages extracted from the usual search engines. We think that we may improve the quality of the optimal concept selected by using better heuristics. One important aspect of these algorithms is that they may find concepts in an incremental way. So, even if the initial concept extraction is expensive in terms of time, updating these concepts is not time consuming.

## References

[1] Khcherif, R., Gammoudi, M. N. and Jaoua, A. "Using Difunctional Relations in Information Organization," *Journal of Information Sciences*, **125**, pp. 153–166, 2000.

[2] Jaoua, A. and Elloumi, S. "Galois Connection, Formal Concepts and Galois Lattice in Real Relations: Application in a Real Classifier," *The Journal of Systems and Software*, **60**, pp. 149–163, 2002.

[3] Mineau, G. W. and Godin, R. "Automatic Structuring of Knowledge Bases by Conceptual Clustering," *IEEE Transactions On Knowledge and Data Engineering*, **7**(5), pp. 824–829, 1995.

[4] Ben Yahia, S., Arour, K., Slimani, A. and Jaoua, A. "Discovery of Compact Rules in Relational Databases," *Information Journal*, **3**(4), pp. 497–511, 2000.

[5] Ben Yahia, S. and Jaoua, A. "Discovering Knowledge from Fuzzy Concept Lattice," in Kandel, A. Last, M. and Bunke, H. (Eds), *Data Mining and Computational Intelligence, Studies in Fuzziness and Soft Computing*, **68**, pp. 167–190, Physica Verlag, Heidelberg, 2001.

[6] Al-Rashdi, A., Al-Muraikhi, H., Al-Subaiey, M., Al-Ghanim, N. and Al-Misaifri, S. *Knowledge Extraction and Reduction System (K.E.R.S.)*, Senior project, Computer Science Department, University of Qatar, June 2001.

[7] Maddouri, M., Elloumi, S. and Jaoua, A. "An Incremental Learning System for Imprecise and Uncertain Knowledge Discovery," *Information Science Journal*, **109**, pp. 149–164, 1998.

[8] Alsuwaiyel, M. H. *Algorithms, Design Techniques and Analysis*, Word Scientific, 1999.

[9] Davey, B. A. and Priestley, H. A. *Introduction to Lattices and Order*, Cambridge Mathematical Textbooks, 1990.

[10] Ganter, B. and Wille, R. *Formal Concept Analysis*, Springer Verlag, 1999.

[11] Schmidt, G. and Ströhlein, S. *Relations and Graphs*, Springer Verlag, 1989.

[12] Jaoua, A., Bsaies, K. and Consmtini, W. "May Reasoning be Reduced to an Information Retrieval Problem," *International Seminar on Relational Methods in Computer Science*, Quebec, Canada, 1999.

[13] Jaoua, A., Boudriga, N., Durieux, J. L. and Mili, A. "Regularity of Relations: A Measure of Uniformity," *Theoretical Computer Science*, **79**, pp. 323–339, 1991.

[14] Riguet, J. "Relations binaires, fermetures et correspondences de Galois," *Bulletin de la societé Mathematique de France*, pp. 114–155, 1948.

[15] Belkhiter, N., Bourhfir, C., Gammoudi, M. M., Jaoua, A., Le Thanh, N. and Reguig, M. "Decomposition Rectangulaire Optimale d'une Relation Binaire: Application aux Bases de Donnees Documentaires," *Canadian Journal:INFOR*, **32**(1), pp. 33–54, 1994.

[16] Jaoua, A., Belkhiter, N., Desharnais, J. and Moukam, T. "Properties of Difunctional Dependencies in Relational Database," *Canadian Journal INFOR*, **30**(1), pp. 297–315, 1992.

[17] Maddouri, M., Elloumi, S. and Jaoua, A. "An Incremental Learning for Imprecise and Uncertain Knowledge Discovery," *Journal of Information Sciences*, **109**, pp. 149–164, 1998.

[18] ElMasri, R. and Navathe, S. B. *Fundamentals of Database Systems*, third edition, Addison-Wesley, 2000.

[19] Mcleod, R., *Management Information Systems: A Study of Computer Based Information Systems*, seventh edition, Simon and Schuster, 2000.

[20] Jaoua, A., Ounalli, H. and Belkhiter, N. "Automatic Entity Extraction From an *N*-ary Relation: Towards a General Law for Information Decomposition," *International Journal of Information Science*, **87**(1–3), pp. 153–169, 1995.
[21] Bélaïd Ajroud, H., Jaoua, A. and Kaabi, S. "Classes extraction from procedural programs," *Information Sciences*, **140**(3–4) pp. 283–294, 2002.
[22] Belaid, H. and Jaoua, A. "Abstraction of objects by conceptual clustering," *Journal of Information Sciences*, **109**, pp. 79–94, 1998.
[23] *A Text Mining System DIREC: Discovering Relationships between Keywords by Filtering, Extracting and Clustering*, http://lpt.unimb.si/jckbse2002/aplication/Apl_Login/upcamera/32-jckbse.pdf.
[24] *Semi-Automatic Indexing of Multilingual Documents and Optimal Rectangle*, http://arxiv.org/PS_cache/cs/pdf/9902/9902022.pdf.
[25] *Natural Language Techniques and Text Mining Applications*, http://citeseer.nj.nec.com/rajman97text.html.
[26] Tutorial: *Text Analyst*, http://www.megaputer.com/products/ta/tutorial/textanalyst_tutorial_1.html.
[27] Mosaid, T., Hassan, F., Saleh, H. and Abdullah, F. *Conceptual Text Mining: Application for Text Summarization*, Senior Project, University of Qatar, January 2004.

# 24

# Cryptographic Communications With Chaotic Semiconductor Lasers

**Andrés Iglesias**
Department of Applied Mathematics and Computational Sciences, University of Cantabria, Avda. de los Castros, s/n, E-39005, Santander, Spain

*The world of digital communications has received much attention in the last few years. Extraordinary advances in both laser and semiconductor technologies have favored the development of new communication systems. However, it was the emergence of the Internet and the Web that created the largest impact on the telecommunication networks. Their open architecture has provided companies and users with promising opportunities for both e-commerce and e-services (e-banking, e-trading, e-training and others). However, these systems have also proved to be quite vulnerable and, consequently, new challenges must be faced. A crucial problem in all communication technologies is security. In this work, two different chaotic schemes for cryptographic communications with semiconductor lasers are described. Both approaches consist of an optical fiber communication network in which the transmitter and the receiver are both semiconductor lasers subjected to phase-conjugate feedback. The laser parameters are carefully chosen in such a way that the lasers exhibit a chaotic behavior, which is used to mask the message from the transmitter to the receiver. Thus, the laser parameters serve as the encryption key. In the first scheme, chaotic masking, the message is added to the chaotic output of the transmitter and then sent to the receiver, which synchronizes only with the chaotic component of the received signal. The message is recovered by a simple subtraction of the synchronized signal from the transmitted one. In the second scheme, chaotic switching, the information is binary and switches the transmitted signal between two different attractors associated with different chaotic receivers. Potential applications of these schemes, as well as their extraordinary advantages in comparison to other cryptographic schemes, are also discussed.*

*Computer-Aided Intelligent Recognition Techniques and Applications* Edited by M. Sarfraz

# 1. Introduction

In the last few years we have witnessed an extraordinary worldwide growth of digital communications. Extraordinary advances in both laser and semiconductor technologies have favored the appearance of a number of very powerful communication systems (Internet, PDAs, Web TV, high-speed industrial networks, high-bandwidth optical fibers, etc.) and many others are still under development. However, it was the emergence of the Internet and the Web that created the largest impact on the telecommunication networks. The current expanding e-commerce economy, including the business-to-consumer and business-to-business sectors, as well as e-banking, e-trading, e-training and others, will require a network infrastructure capable of supporting the growing number of Internet transactions at an increasing rate. As a consequence, this will create an ongoing demand for more powerful servers, networks and increasing bandwidths. Laser technology based on specialized semiconductor equipment will play a key role in this promising future development of digital communications. In particular, most of the current development of digital communications is motivated by laser technology, in which optical fibers are used for data transmission [1]. By means of optical fiber, our computers can effectively support real-time video, multimedia, Internet, etc. and enormous amounts of information can easily be transmitted. In this new technology, the role of *semiconductor lasers* (or *diode lasers*) is analogous to the role of transistors in electronics. An optical telecommunication network can be seen as a network of semiconductor lasers connected by optical fibers. These semiconductor lasers are needed for the generation of signals, for amplification after many kilometers, and for retrieval of the information at the end point.

On the other hand, the users of this new technology demand effective protection of their information. While companies of all sizes increasingly are utilizing the Internet and Web technologies to lower costs and increase efficiencies and revenues, the open architecture of the Internet and Web opens organizations and users up to an increasing array of security threats. The need for security, evidenced since mid 2003 by increasing series of attacks on networks and systems all over the world, introduces new challenging problems that must be addressed.

In order to achieve a higher security and efficiency level, we must ensure first that only authorized users will gain access to the company resources and systems. For example, in many communication systems (army, police, medical services, etc.) it is very important to establish the clear and unquestionable identity of the communicating parties (authentication). Current security on the Internet involves brute-force firewalls that deny all unsolicited traffic. However, sophisticated Web applications require more powerful methods of protection against attacks. This is also a key issue in current banking services, since the most usual authentication tools, such as magnetic cards and personal identification numbers currently used to access Automatic Teller Machines (ATMs), do not provide a sufficient degree of security and are probably a source of unauthorized operations. In addition, it is important to prevent the sender from later denying that he/she sent a message (nonrepudiation). Because of some advantages (ease of deployment and lower costs), these features are often checked by software. However, the highest level of reliability for (among others) authentication and nonrepudiation involves *hardware* that must be associated with authorized users and that is not easy to duplicate. As a consequence, a number of different approaches based on hardware have recently arisen. In addition to the classical cryptographic models, we can quote those based on biometric technology, such as fingerprint recognition, face recognition, iris recognition, hand shape recognition, signature recognition, voice recognition, etc. (see, for example, [2] to get a quick insight about some recent advances on this topic). The core of this new paradigm for security is to use human body characteristics to identify and authenticate users. Unlike ID cards, passwords or keycards (which can be forgotten, lost or shared), these new techniques try to take advantage of ‘what users are’, as opposed to ‘what users have’. However, biometric mechanisms are still prone to errors, as they fail to provide effective and reliable recognition. Additional shortcomings are the huge database required for storage of biometric templates and the fact that they cannot eliminate the use of stolen or copied valid signatures.

To overcome these limitations, different cryptographic schemes have been applied to hide information from unauthorized parties during the communication process [3,4]. The basic elements of these schemes are: a *sender* (or *transmitter*), a *receiver* and a *message* to be sent from the transmitter to the receiver. It is assumed that any communication between sender and receiver may be read or intercepted by a hostile person, the *attacker*. The primary objective of cryptography is to encode the message in such a way that the attacker cannot understand it. Furthermore, the most recent cryptographic models incorporate additional methods for many other tasks (the interested reader is referred to [4–7] for a gentle introduction to cryptography with many algorithms, C/C++ codes, pseudocode descriptions of a large number of algorithms and hardware aspects of cryptography), such as:

- *access control*: implies protection against the unauthorized use of the communication channel;
- *authentication*: provides methods to authenticate the identity of the sender;
- *confidentiality*: protects against disclosure to unauthorized parties;
- *integrity*: protects against the unauthorized alteration of the message;
- *nonrepudiation*: prevents the sender from later denying that he/she sent a particular message on a particular date and time;
- *availability*: implies that the authorized users have access to the communication system when they need it.

Of course, some of the previous features can be combined. For example, user authentication is often used for access control purposes, nonrepudiation is combined with user authentication, etc. To provide the users with the previous features, a number of different methods have been developed [5–7]. Among them, the possibility of encoding messages within a chaotic carrier has received considerable attention in the last few years [8–19]. In this scheme, both the transmitter and the receiver are (identical) chaotic systems. The chaotic output of the transmitter is used as a carrier in which the message is encoded (masked). The amplitude of the message is much smaller than the typical fluctuations of the chaotic carrier, so that it is very difficult to isolate the message from the chaotic carrier. Decoding is based on the fact that coupled chaotic systems are able to synchronize their output under certain conditions [18]. To decode the message, the transmitted signal is coupled to the chaotic receiver, which is similar to the transmitter. The receiver synchronizes with the chaotic carrier itself, so that the message can be recovered by subtracting the receiver output from the transmitted signal.

This scheme has been applied to secure communications with electronic circuits [11–13,16] and lasers [10,15]. Unfortunately, many of these models exhibit shortcomings that dramatically restrict their application to secure communications. The main one is that, as shown in [20–22], messages masked by low-dimensional chaotic processes, once intercepted, are sometimes readily extracted, even though the channel noise is rather high [23]. This fact explains why this kind of scheme has not been extensively developed for commercial purposes yet.

Until a few years ago, the previous limitation was considered to be overcome by employing either high-dimensional chaotic systems [24] or high-frequency devices like lasers. However, some recent results have reported extraction of messages with very high dimensions and high chaoticity [25], thus limiting the applicability of these systems. By contrast, high-frequency systems are still seen as optimal candidates for chaotic cryptography.

In this context, the present chapter describes two different schemes (chaotic masking and chaotic switching) based on chaos for cryptographic communications with semiconductor lasers. Both approaches consist of an optical fiber communication network in which the transmitter and the receiver are both (identical) semiconductor lasers subjected to phase-conjugate feedback. The laser parameters are carefully chosen in such a way that the lasers exhibit a chaotic behavior, which is used to mask the message. Thus, the laser parameters serve as the *encryption key*. In the first scheme, chaotic masking, the message is added to the chaotic output of the transmitter and then sent to the receiver, which synchronizes only with the chaotic component of the received signal. The message is recovered by a simple subtraction of the synchronized signal from the transmitted one. In the second scheme,

chaotic switching, the information is binary and switches the transmitted signal between two different attractors associated with different chaotic receivers. These new systems provide all the features listed above, they are neither hard to make nor very expensive, they are portable, they support much more information than any other communication system and finally, they are impossible to duplicate with current technology. Each of these statements will be adequately reasoned in the following sections.

The structure of the chapter is as follows: Section 2 describes the semiconductor lasers subjected to phase-conjugate feedback. The steps needed for the implementation of a chaotic cryptographic communication system based on this kind of laser are also discussed in this section. Then, Section 3 describes two different schemes (chaotic masking and chaotic switching) for applying such a communication system to cryptographic communications. The performance of these methods is analyzed by means of some illustrative and interesting examples. Finally, Section 4 closes with the main conclusions of this work.

## 2. Semiconductor Lasers with Optical Feedback

Lasers have become one of the most remarkable technologies of recent decades. They are indispensable in a variety of fields: the entertainment industry (laser games, light shows), dentistry and medicine (microsurgery), manufacturing (for example, machining and welding), remote sensing, etc. However, the most outstanding current application of lasers appears in telecommunications, where optical fibers are used for data transmission [1,26]. A communication network based on optical fibers has several orders of magnitude more information-carrying capacity than a copper-wire-based net (for instance, ten million telephone conversations are considered to be feasible for lasers with short-range connections ($\leq 50$ km), while a standard copper-wire connection can only carry one hundred telephone connections over 200 meters, after which it needs amplification). For a cryptographic communication system based on lasers to be developed, we need to work through four main steps that will be carefully analyzed in the following paragraphs:

- choice of the laser;
- determination of the laser equations and parameters;
- choice of an accessible parameter for chaoticity;
- synchronization of the chaotic transmitter and receiver systems.

### *2.1 Step 1: Choice of the Laser*

In this section we discuss the choice of the kind of laser and its components. This process is somewhat similar to choosing the hardware components of a computer. Here, our decision is mostly determined by the requirements of communication and security.

All lasers basically consist of a medium capable of amplification of light and a cavity providing an adequate mechanism for frequency selection [27]. In the case of semiconductor lasers, various types of semiconductor materials may be used. These materials emit, under suitable conditions, light with wavelengths ranging from the visible to the infrared ($\approx 0.55$ to $\approx 2.0\,\mu$m) [1]. This range is especially attractive for optical communication purposes, because silica fibers are virtually transparent for 1.3 and 1.55 $\mu$m, providing a very low distortion of optical signals when transmitted over long distances. For these reasons, semiconductor lasers have played a key role in the replacement of the 'old' copper-wire-based networks by optical fiber networks. This conversion makes it possible to transmit efficiently and cheaply enormous amounts of information all over the world.

In addition, diode lasers are relatively easy to make and are very reliable, exhibiting a very small size: a typical helium–neon laser (such as those in supermarket scanners) has a length of a few tens of centimeters, whereas the typical length of a semiconductor laser is about 0.25 millimeters. Finally, they are very efficient: their efficiency ranges from 10% to 60%, whereas gas lasers have a typical

efficiency of 1%. All these facts explain why these lasers are so popular as light sources in optical communication networks.

Of course, semiconductor lasers also have their disadvantages: they are very sensitive to external perturbations and can easily be destabilized, for example, when the laser is current modulated or subjected to an external signal. This external field can be light from another laser or light source, or light obtained by reflection from a mirror. These situations are usually referred to as *optical injection* and *optical feedback*, respectively [28].

The optical feedback comes from an external mirror positioned at a distance $\frac{c.\tau}{2}$ (where c represents the velocity of light in a vacuum), thereby creating an external cavity with round-trip $\tau$. Depending on the type of mirror (conventional or phase-conjugate), two different types of optical feedback can be considered (see Figure 24.1):

- *Conventional Optical Feedback* (COF): the plane wave is reflected at an ordinary mirror; only the component of the wave vector that is perpendicular to the mirror is reversed (Figure 24.1(a)).
- *Phase-Conjugate Feedback* (PCF): a beam is reflected by a phase-conjugate mirror; the complete wave vector is reversed and the beam retraces its original path (Figure 24.1(b)).

The effect of optical feedback on these systems has been intensively studied since the late 1970s from both the experimental and theoretical points of view [29–34]. For instance, it has been proven that diode lasers with optical feedback can be stabilized in some cases [35,36], making them useful for communications. In addition, PCF mirrors exhibit interesting advantages with respect to conventional mirrors. The most important one is related to their distortion-undoing property. Any phase distortion between source and phase-conjugator is fixed on the way back by the fact that light beams retrace exactly their path to their respective sources. This property implies that PCF mirrors are a cheaper alternative to the conventional ones, which had to be controlled.

There is, lastly, an additional major advantage of PCF lasers: perfect synchronization (zero error) is only possible for diode lasers with a phase-conjugate mirror. Motivated by these attractive advantages, in this chapter we will focus on semiconductor lasers subjected to PCF.

## 2.2 Step 2: Determination of the Laser Equations and Parameters

To understand the synchronization process described later, we need to introduce the rate equations of the laser. They describe the behavior of the electric field $E(t)$ and the inversion $N(t)$ (the total

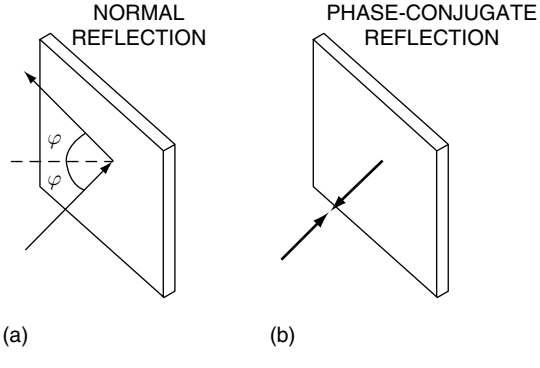



**Figure 24.1** Two cases of optical feedback in semiconductor lasers. (a) Normal reflection; (b) phase-conjugate reflection.

number of electron-hole pairs in the cavity) in a single-mode laser. Theoretical investigations of optical feedback on semiconductor lasers are usually based on the Lang–Kobayashi equations [37], which contain all the dominant effects observed experimentally. These rate equations are:

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = \frac{1}{2}(1-i\alpha)\left[G(t) - \frac{1}{\tau_n}\right]E(t) + \kappa E^*(t-\tau) + \sqrt{2\beta N(t)}\xi(t) \tag{24.1}$$

and

$$\frac{\mathrm{d}N(t)}{\mathrm{d}t} = \frac{I}{q} - \frac{N(t)}{\tau_e} - G(t)\,|E(t)|^2 \tag{24.2}$$

where

$$G(t) = \frac{g\,(N(t)-N_0)}{\left(1+s\,|E(t)|^2\right)} \tag{24.3}$$

The terms of the right-hand-side of Equation (24.1) can be interpreted as follows:

- the first term describes the behavior of the electric field for the *solitary* laser (the semiconductor laser isolated from the outside world);
- the term $\kappa E^*(t-\tau)$ is associated with the optical feedback (the superscript $^*$ of the electric field stands for the complex conjugate). It is accounted for by two parameters: the delay time $\tau$, and the feedback rate $\kappa$;
- the last term $\sqrt{2\beta N(t)}\xi(t)$ is included here to account for the spontaneous emission noise that naturally arises in real applications. Such a noise is modeled by a complex Gaussian white noise term $\xi(t)$ of zero mean, $<\xi(t)>=0$, and correlation $<\xi_i(t)\xi_j(t')>=\delta_{ij}(t-t')$. The parameter $\beta$ stands for the spontaneous emission rate.

The rest of the parameters, their symbols and the values used in this chapter are listed in Table 24.1. We remark that it is not possible to create, *a priori*, lasers with the same parameters as those listed in Table 24.1. What is really done in practical settings is to construct a new laser, then to determine its parameters (which are only approximately chosen by the manufacturer) and finally to analyze its

**Table 24.1** Parameter values of the semiconductor laser with phase-conjugate feedback.

| Parameter | Symbol | Value |
|---|---|---|
| Linewidth enhancement factor | $\alpha$ | 5 |
| Photon lifetime | $\tau_n$ | 2 ns |
| Feedback coefficient | $\kappa$ | $0.0238\,\mathrm{ps}^{-1}$ |
| External cavity round-trip time | $\tau$ | 200 ps |
| External cavity length | $L_{\mathrm{ext}}$ | 1 cm |
| Spontaneous emission rate | $\beta$ | $1.5\times10^{-9}\,\mathrm{ps}^{-1}$ |
| Bias current | $I$ | 44 mA |
| Electron charge | $q$ | $1.602\times10^{-19}\,\mathrm{C}$ |
| Carrier lifetime | $\tau_e$ | 2 ns |
| Gain parameter | $g$ | $1.5\times10^{-8}$ |
| Nonlinear gain parameter | $s$ | $5\times10^{-7}$ |
| Transparency carrier number | $N_0$ | $1.5\times10^{-8}$ |

dynamical behavior when varying an accessible parameter (for example, the distance between the laser and the mirror, $L_{\text{ext}}$). Other laser and mirror parameters cannot be chosen but only determined after construction. Especially interesting is the case of the mirror, because it is extracted from the central part of a 'wafer' whose dynamical behavior changes as a function of the points of the piece. Thus, usually only a few (typically two or three) small mirrors with the same parameters can be obtained from the same piece. These mirrors can be used for the transmitter and the receiver lasers, so the assumption that both systems are identical is feasible in practice. After finishing this process, no other mirror with the same parameters can be made. This is a very important issue, because very strong authentication requires hardware components that can be in the possession of only a few people at a time. This situation differs radically from electronic circuits, which can easily be duplicated with current engineering.

## 2.3 Step 3: Choice of Some Accessible Parameter for Chaoticity

Lasers have attracted considerable attention because of their interesting nonlinear behavior. In 1975, Haken [38] showed that in some cases, the behavior of a laser is equivalent to that of the well-known Lorenz equations ('chaos') [39]. Since then, numerous types of laser have been used to investigate nonlinear dynamics [40,41]. Chaotic behavior of semiconductor lasers subjected to optical injection has been analyzed in [42] and in [30] for the case of weak optical feedback.

In general, semiconductor lasers display richer chaotic dynamics in the case of PCF [29]. In fact, the output of these lasers is chaotic due to the optical feedback from a phase-conjugate mirror. The dynamics of the laser diode with external cavity can be generalized as a simple delay-differential model:

$$\frac{\mathrm{d}\boldsymbol{X}(t)}{\mathrm{d}t} = \mathbf{F}(\boldsymbol{X}(t), \boldsymbol{X}(t-\tau); \mu) \tag{24.4}$$

where $\boldsymbol{X}(t)$ is a vector variable, $\mathbf{F}$ is a nonlinear vector field, $\tau$ is the delay time introduced by the external feedback and $\mu$ is a system control parameter. In view of the dynamics of the delay-differential system, several phase transitions among steady, periodic, quasiperiodic and chaotic states have been appreciated.

For quantifying the laser dynamics, the use of bifurcation diagrams is well established [34]. It is a powerful tool, since the diagram shows at a glance for which strengths of feedback the laser operates stably, periodically or chaotically. For diode lasers with PCF, the bifurcation diagrams are obtained by generating a time series for each feedback level and noting the carrier number $N$ when the laser power crossed the solitary laser value $N_{\text{sol}}$ vs. $\kappa.\tau$, this last term meaning the feedback level. For the choice of the laser parameters given in Table 24.1 and $\tau = 200\,\text{ps}$, we vary $\kappa$ such that $\kappa.\tau \in [1, 5]$ (i.e. $\kappa \in [0.005, 0.025]$). The corresponding bifurcation diagram is displayed in Figure 24.2. The figure has been split into two parts: (a) shows the diagram for the whole interval $\kappa \in [0.005, 0.025]$, and (b) a zoom of the chaotic region $\kappa \in [0.0235, 0.025]$.

As the reader may appreciate from Figure 24.2(a), the bifurcation diagram for diode lasers with PCF is very complicated. Roughly speaking, a single crossing in the bifurcation diagram implies a periodic output, whereas multiple crossings indicate period doublings, quasiperiodicity or chaos. In Figure 24.2(a), the laser exhibits a steady state (corresponding to the blank regions without any dot) on an interval for $\kappa$ starting at the initial value $\kappa = 0.005$, which becomes unstable at a critical value of $\kappa$, and the laser becomes periodic. After a period-doubling region, quasiperiodicity (for example, for $\kappa = 0.0187$) and chaotic behaviors appear. However, chaos is interrupted by regions of nonchaotic output although, in general, this output does not become completely stable. Figure 24.2(b) shows a zoom of the chaotic region corresponding to $\kappa \in [0.0235, 0.025]$.

For the synchronization process to be possible we need a chaotic behavior for the laser. From Figure 24.2, we have considered a feedback coefficient $\kappa = 0.0238$, for which the laser behaves chaotically. To check this, in Figure 24.3 we have represented both the trajectory of the laser intensity (a) and its Fourier spectrum (b) for this parameter value.

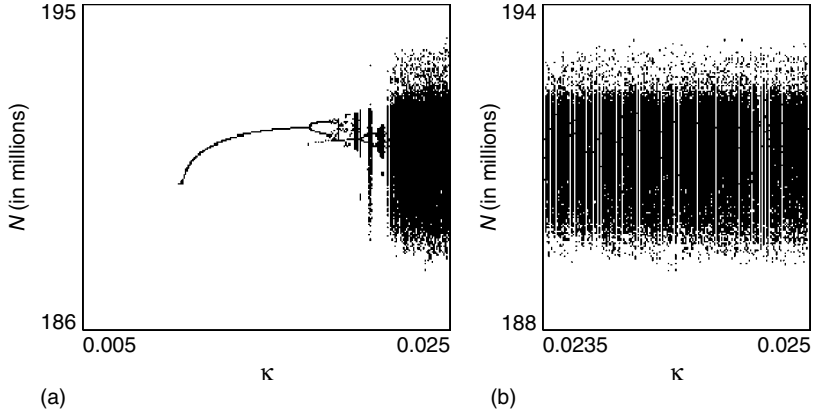



**Figure 24.2** Bifurcation diagram for a semiconductor laser subjected to phase-conjugate feedback. (a) Diagram for $\kappa \in [0.005, 0.025]$; (b) a zoom of the chaotic region $\kappa \in [0.0235, 0.025]$.

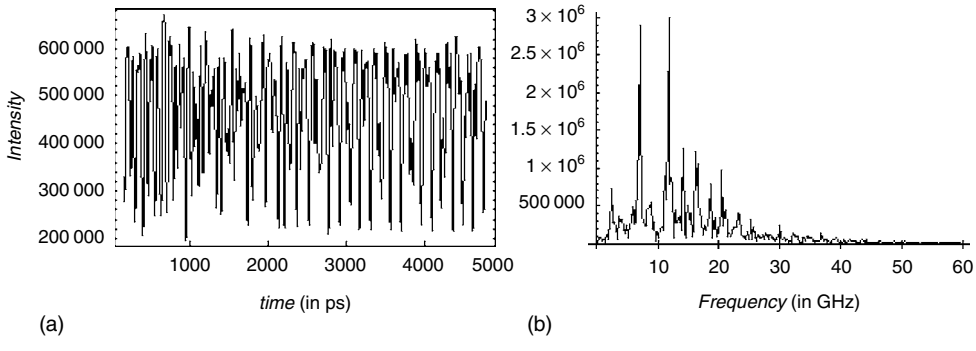



**Figure 24.3** Chaotic behavior of the diode laser with PCF for $\kappa = 0.0238$. (a) Intensity of the chaotic orbit during 5 ns; (b) Fourier spectrum.

## *2.4 Step 4: Synchronization of the Chaotic Transmitter and Receiver Systems*

The synchronization problem consists of making two or more chaotic systems oscillate in a synchronized way. Although at first sight this seems impossible, many recent results have shown that synchronization can be achieved under certain conditions. In 1990, Pecora and Carroll [18] showed that certain subsystems of chaotic systems can be synchronized by linking them with common signals. Since then, many other chaotic synchronization schemes have been reported in the literature. Among them, some generalizations of the previous Pecora–Carroll scheme [43], the use of control [44], convex combinations [45], system separation [46], etc. Applications of these schemes include discrete-time chaotic systems [47,48], hyperchaos [24], spatiotemporal chaos in coupled nonlinear oscillators [49], chaotic circuits [9,16,50], optical systems [8], lasers [10,15], model neurons [51], etc.

In this section we explore the synchronization of two semiconductor lasers subjected to PCF and, hence, described by Equations (24.1)–(24.3). Since the effect of the noise will be discussed later, its associated term will be neglected, so we rewrite Equations (24.1)–(24.3) as:

$$\frac{\mathrm{d}E_{\mathrm{t,r}}(t)}{\mathrm{d}t} = \frac{1}{2}(1-i\alpha)\left(G_{\mathrm{t,r}}(t) - \frac{1}{\tau_{\mathrm{t,r}}}\right)E_{\mathrm{t,r}} + \kappa E^{*}_{\mathrm{t,r}}(t-\tau) + K_{\mathrm{r}}E_{\mathrm{t}} \tag{24.5}$$

$$\frac{\mathrm{d}N_{\mathrm{t,r}}(t)}{\mathrm{d}t} = \frac{I}{q} - \frac{1}{\tau_n}N_{\mathrm{t,r}}(t) - G_{\mathrm{t,r}}(t)\left|E_{\mathrm{t,r}}(t)\right|^2 \tag{24.6}$$

where:

$$G_{\mathrm{t,r}}(t) = \frac{g(N_{\mathrm{t,r}}(t) - N_0)}{(1 + s\left|E_{\mathrm{t,r}}\right|^2)} \tag{24.7}$$

and the subscripts t and r are used to indicate respectively the transmitter and the receiver. The parameter values are those listed in Table 24.1 with $\kappa = 0.0238$, for which both lasers behave chaotically, as shown in Section 2.3. On examining these equations, the receiver is seen to be similar to the transmitter except for the fact that it incorporates a new term $K_{\mathrm{r}}E_{\mathrm{t}}$, meaning that we inject a small amount (given by the coupling parameter $K_{\mathrm{r}}$) of the transmitter signal $E_{\mathrm{t}}$ to the receiver. Thus, by varying this parameter $K_{\mathrm{r}}$, we modify the amount of the injected signal. Furthermore, it can be proven that, for certain choices of the laser parameter values, this synchronization is mathematically perfect. This property can only be obtained for semiconductor lasers with PCF and involves a new parameter $\Delta\omega_{\mathrm{t,r}}$ (which has been assumed to be zero in our simulations), describing the frequency mismatch between the solitary semiconductor laser and the laser used to pump the phase-conjugate mirror.

Figure 24.4 shows the synchronization for two different values of $K_{\mathrm{r}}$: 0.025 and 0.05. The figure represents the number of carriers of the transmitter laser $N_1$ (above) and the difference between the transmitter and the receiver $N_1$–$N_2$ (below). Of course, the difference $N_1$–$N_2$ should vanish for a perfect synchronization. This occurs for both values of $K_{\mathrm{r}}$, meaning that synchronization is robust to small perturbations of this parameter. From these figures, it becomes clear that this kind of laser allows a very accurate synchronization with a very brief transient, two remarkable features for communications.

The above scheme for a couple of systems can also be extended to a chain, or network, of chaotic systems. In this case, the response system (with respect to a given drive) acts as the drive of a second

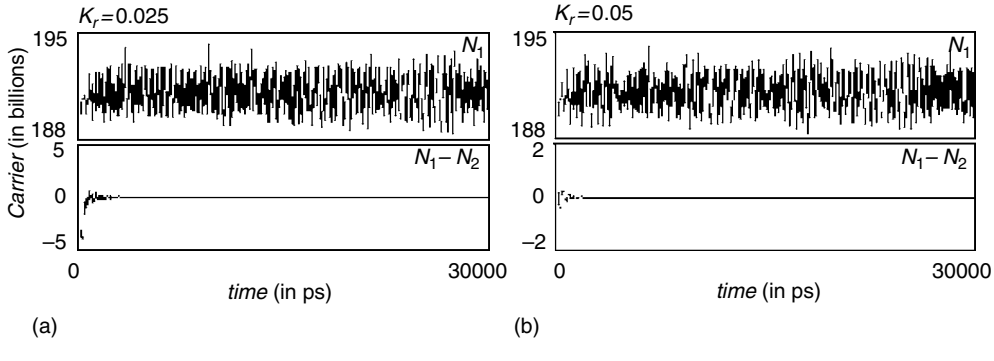



**Figure 24.4** Synchronization of chaotic lasers for different values of $K_{\mathrm{r}}$. (a) 0.025; (b) 0.05. Each figure represents the transmitter (above) and the difference between the transmitter and the receiver (below).

response system, and so on. Thus, a *cascade* of synchronized systems can be obtained. We remark that this set-up has very interesting applications in the field of secure communications.

## 3. Applications to Cryptographic Communications

Most of the interest in chaotic synchronization has been motivated by its potential application to cryptographic communication [10–12,14–16]. Here, the information message is 'coded' by means of a chaotic signal and then transmitted. The receiver contains a copy of the chaotic system that generates the cipher key and produces the decoding signal by synchronizing with it. To accomplish this task, a number of different methods have been described. Among them, perhaps the most interesting and easiest to develop are chaotic masking and chaotic switching. They are analyzed in the following sections.

### *3.1 Chaotic Masking*

In this scheme, depicted in Figure 24.5, the optical carrier modulated by a small signal (the message $E_M$) is added to the transmitter laser output $E_D$, i.e. $E_T = E_D + E_M$, and transmitted to a response system (the *receiver*), which is a replica of the laser that generates the chaotic signal (the *drive*). Decoding is performed by the chaotic receiver, providing that it synchronizes only with the chaotic component of the received signal, which is therefore available on the receiver laser output. A subtraction of the synchronized signal $E_R \approx E_D$ from the transmitted signal $E_T = E_D + E_M$ results in the recovery of the message.

We have applied this scheme to digital communications, where both the transmitter and the receiver are semiconductor lasers with PCF. Figure 24.6 summarizes the different steps in this process: the message $M$ consists of a digital signal generated by a table of bits 1 and $-1$, as shown in Figure 24.6(a). The message is encoded by a small variable attenuation of the chaotic output (for example, the electric field) of a laser subjected to PCF (transmitter) by:

$$|E'_t(t)| = |E_t(t)|\,(1 + \omega.M(t)) \tag{24.8}$$

where $\omega$ is a real number. Of course, the amplitude of the signal to be transmitted $M(t)$ must be small in comparison to the chaotic signal. Otherwise, information could be extracted directly from the waveform. The transmitted signal $||E'||$ (Figure 24.6(b)) is chaotic and virtually indistinguishable from that of the chaotic drive (for instance, they both have a similar power spectrum). Therefore, no information can be extracted from the encrypted signal. Then, it is coupled to another chaotic laser (receiver) that is similar to the transmitter. The receiver synchronizes with the chaotic carrier itself $|E_r(t)| \approx |E_t(t)|$. The message can be recovered from the transmitted signal as $D(t) = \sqrt{|E'_t(t)|^2/|E_r(t)|^2} - 1$ (Figure 24.6(c)) and

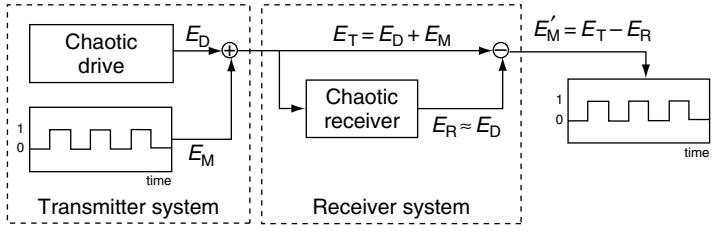



**Figure 24.5** Scheme of secure communication based on chaotic masking.

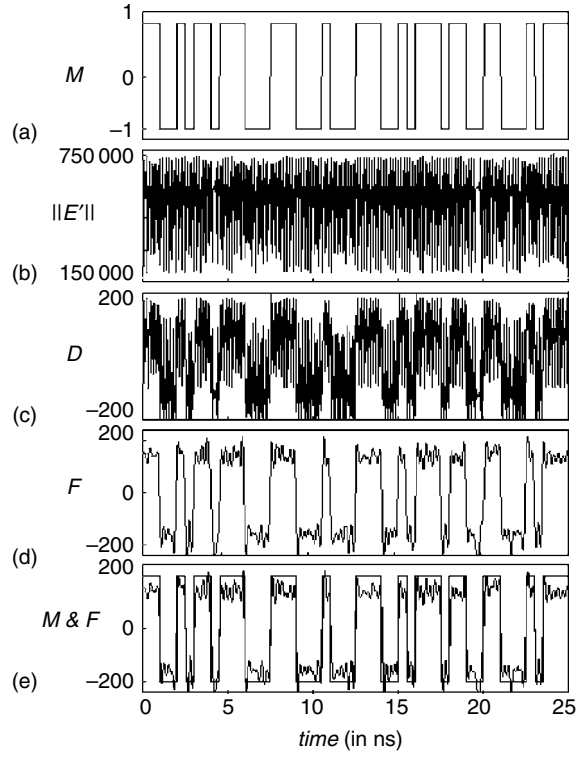



**Figure 24.6** Chaotic masking scheme. (a) A digital message; (b) the transmitted signal encoded; (c) the recovered signal; (d) the recovered signal filtered; (e) the original and filtered messages.

then filtered (Figure 24.6(d)). A simple visual comparison of the original and the filtered messages (Figure 24.6(e)) shows the excellent performance of the method.

Another interesting application of this methodology is the encoded transmission of digital images. As an example, let us consider the image given in Figure 24.7, which consists of a $100 \times 100$ matrix of gray-level intensities ranging from 0 to 255.

In this case, we can use the couple of drive-response drivers to encode and decode the image. In the chaotic masking method, the chaotic signal from the transmitter is used to encode the list of gray-level intensities after a transient of 20 iteration steps. Afterwards, the scheme displayed in Figure 24.5 is applied to recover the original image by using the synchronization process to clean the chaotic signal out. Figure 24.8 shows the transmitted and received images. A simple visual comparison of Figures 24.7 and 24.8(b) shows the excellent performance of the method. Note that no information about the appearance of the initial image can be obtained from the noisy transmitted signal, shown in Figure 24.8(a).

## *3.2 Chaotic Switching*

Chaotic masking is not totally efficient, because only a part of the transmitted power carries information, most being used to mask information. *Chaotic switching* (also known as *chaotic shift keying*) allows a higher level of efficiency and security [13].

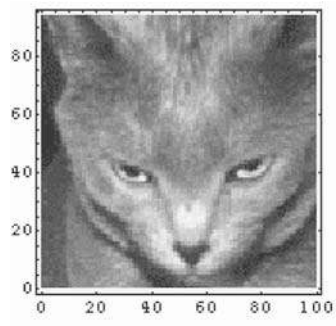



**Figure 24.7** Example of a digital image.

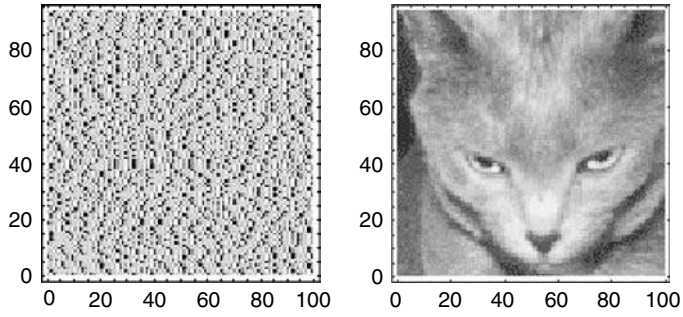



**Figure 24.8** (a) Transmitted and (b) recovered images.

In this case, the information is binary and switches the transmitted signal between two different attractors associated with different chaotic receivers. This scheme is summarized in Figure 24.9. The transmitter is switched between two different chaotic orbits by modulating the supply current of the laser by the bit stream to be transmitted; bits $M_1$ and $M_2$ correspond to different pump currents $P_1$ and $P_2$. Decoding at the receiver is performed by using two replicas of the chaotic drive, the first pumped at current $P_1$ and the second at $P_2$. The received signal is sent to both decoders; however, only the one having the pump current corresponding to the transmitted bit will synchronize. Signals $E_1$ and $E_2$ are then obtained as the difference between the output of each decoder and the chaotic received signal. Finally, the transmitted bits can be conveniently recovered, detecting which of $E_1$ or $E_2$ is zero. In Figure 24.10, the outputs $E_1$, $E_2$ of the decoders for bit $M_1$ and $M_2$ are compared to the transmitted signal (at the bottom). Signal $E_1/E_0$ falls to zero when system 1 synchronizes (bit $M_1$ transmitted), while it is chaotic when it does not (bit $M_2$ transmitted). Signal $E_2/E_0$ from the other decoder has a complementary trend.

It should be noticed that the selection of the laser parameters is critical, since it must guarantee both secure communication and efficient decoding. Too closely spaced orbits would require a very precise matching between transmitter and receiver; in addition, synchronization transients would be long. On the contrary, too distant orbits would allow an attacker to make a working copy of the receiver, even without an accurate knowledge of the parameters.

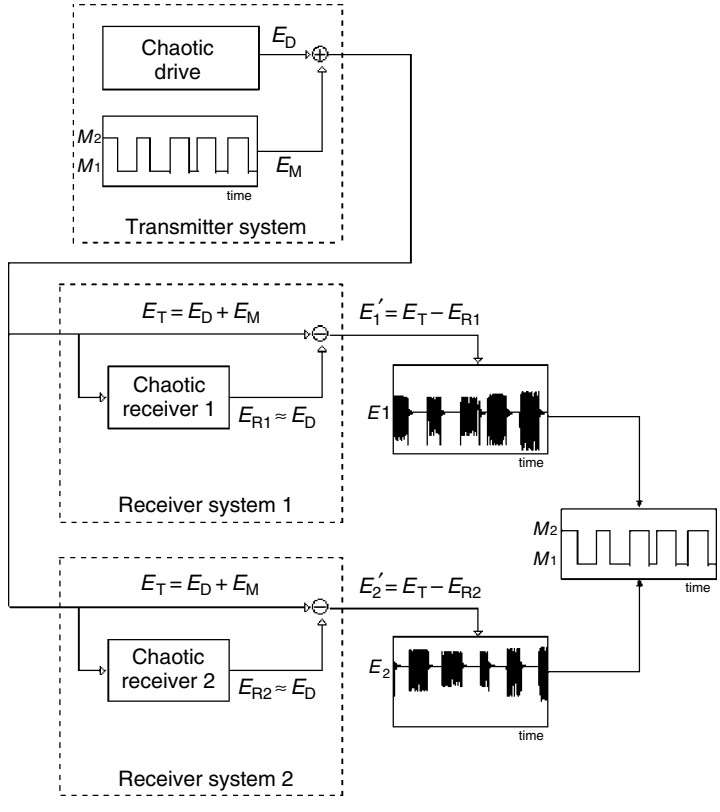



**Figure 24.9** Scheme of secure communication based on chaotic switching.

## 4. Conclusions

In this chapter, two chaotic schemes for cryptographic communications with semiconductor lasers have been described. Both systems consist of a network of chaotic semiconductor lasers subjected to phase-conjugate feedback which, under the appropriate conditions, can be synchronized. This ability of chaotic systems to synchronize can be successfully applied to secure communications. The first scheme, chaotic masking, is simpler to implement as it requires only two systems, while the second scheme, chaotic switching, is more efficient. The proposed methods improve substantially many other methods based on hardware for tasks such as access control, integrity, authentication and nonrepudiation. In particular, these systems are neither hard to make nor very expensive, they are very portable, they support much more information than any other communication system and finally, they are impossible to duplicate with current technology. Potential applications of the methods presented in this chapter include all communications systems based on semiconductor lasers, ranging from e-safety on network systems such as the Internet and the Web, to security in distributed environments. The proposed methods also offer a reasonable alternative for authentication to the methods based on biometrics.

We remark that both systems are completely digital, thus allowing perfect matching (at the coding level) of the receiver to the transmitter to be obtained. In addition, chaotic masking and switching have been shown to be more secure than masking with noise, since in the latter case, data bits can be extracted by using a correlator. This is not possible for chaotic lasers, since the correlation time of the

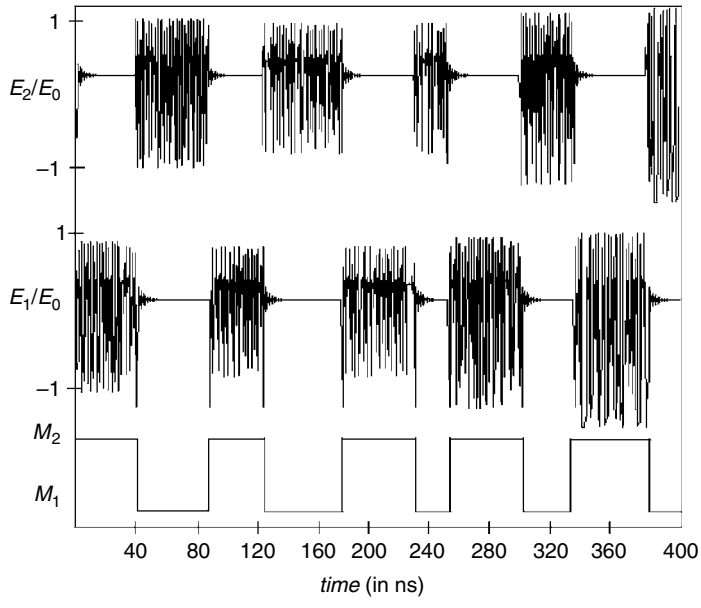



**Figure 24.10** Output signals $E_1/E_0$ and $E_2/E_0$ showing demodulation of bits $M_1$ and $M_2$.

bits and of the chaos are almost identical. Furthermore, because the systems do not include electrical feedback (they are fully optical), these schemes are suitable for high-speed optical transmission.

Finally, it should be remarked that fiber dispersion and nonlinearities can deteriorate the transmitted signal and make decoding impossible [26]. Although not included here because of space limitation, we have checked that, under the appropriate circumstances (such as a fiber length not longer than 50 km and a loss coefficient $\nu = 0.2\,\mathrm{dB/km}$, among others), this does not occur here. However, further research is required to analyze the response of the systems under less favorable conditions. Our current work on this issue is still in progress and the obtained results will be reported elsewhere.

## References

[1] Agrawal, G. P. and Dutta, N. K. *Long Wavelength Semiconductor Lasers*, second, edition, Van Nostrand Reinhold, New York, 1993.

[2] Tistarelli, M., Bigun, J. *et al.* "Biometric Authentication," *Proceedings of the International European Conference on Computer Vision ECCV 2002*, Lecture Notes in Computer Science 2359, Springer Verlag, Berlin Heidelberg, 2002.

[3] Davies, D. and Price W. *Security for Computer Networks*, John Wiley & Sons, Inc., New York, 1989.

[4] Seberry, J. and Pieprzyk, J. *Cryptography: An Introduction to Computer Security*, Prentice Hall, Englewoods Cliffs, N.J., 1989.

[5] Menezes, A. J. and van Oorschot, P. C. *Handbook of Applied Cryptography*, CRC Press, Boca Raton, Florida, 1996.

[6] Rhee, M. Y. *Cryptography and Secure Data Communications*, McGraw-Hill, Boston, 1994.

[7] Schneier, B. *Applied Cryptography*, second edition, John Wiley & Sons, Inc., New York, 1996.

[8] Celka, P. "Chaotic synchronization and modulation of nonlinear time-delayed feedback optical systems," *IEEE Transactions on Circuits and Systems*, **42**, pp. 455–463, 1995.

[9] Chua, L. O., Kocarev, L. *et al.* "Experimental chaos synchronization in Chua's circuit," *International Journal of Bifurcation and Chaos*, **2**, pp. 705–708, 1992.

[10] Colet, P. and Roy, R. "Digital communications with synchronized chaotic lasers," *Optics Letters*, **19**, pp. 1056–2058, 1994.
[11] Cuomo, K. M. and Oppenheim, A. V. "Circuit implementation of synchronized chaos with applications to communications," Physical Review Letters, **71**, pp. 65–68, 1993.
[12] Cuomo, K. M., Oppenheim, A. V. *et al*. "Synchronization of Lorenz-based chaotic circuits with applications to communications," *IEEE Transactions on Circuits and Systems*, **40**, pp. 626–633, 1993.
[13] Dedieu, H., Kennedy, M. P. *et al*. "Chaos shift keying: modulation and demodulation of chaotic carrier using self-synchronizing Chua's circuits," *IEEE Transactions on Circuits and Systems*, **40**, pp. 634–642, 1993.
[14] Kocarev, L., Halle, K. S. *et al*. "Experimental demonstration of secure communications via chaotic synchronization," *International Journal of Bifurcation and Chaos*, **2**, pp. 709–713, 1992.
[15] Mirasso, C. R. and Colet, P. "Synchronization of chaotic semiconductor lasers: application to encoded communications," *IEEE Photonics Technology Letters*, **8**(2), pp. 299–301, 1996.
[16] Murali, K. and Lakshmanan, M. "Transmission of signals by synchronization in a chaotic Van der Pol–Duffing oscillator," *Physical Review E*, **48**, pp. 1624–1626, 1993.
[17] Parlitz, U. and Chua, L. O. "Transmission of digital signals by chaotic synchronization," *International Journal of Bifurcation and Chaos*, **2**, pp. 973–977, 1992.
[18] Pecora, L. M. and Carroll T. L. "Synchronization on chaotic systems," *Physical Review Letters*, **64**, pp. 821–824, 1990.
[19] Pecora, L. M. and Carroll, T. L. "Driving systems with chaotic signals," *Physical Review A*, **44**, pp. 2374–2383, 1991.
[20] Perez, G. and Cerdeira, H. A. "Extracting messages masked by chaos," *Physical Review Letters*, **74**, pp. 1970–1973, 1995.
[21] Yang, T. "Recovery of digital signals from chaotic switching," *International Journal of Circuit Theory and Applications*, **23**, pp. 611–615, 1995.
[22] Yang, T., Yang, L. B. *et al.* "Application of neural networks to unmasking chaotic secure communication," *Physica D*, **124**, pp. 248–257, 1998.
[23] Zhou, C. S. and Chen, T. L. "Extracting information masked by chaos and contamined with noise: some considerations on the security of communication approaches using chaos," *Physics Letters A*, **234**, pp. 429–435, 1997.
[24] Peng, J. H., Ding, E. J. *et al.* "Synchronizing hyperchaos with a scalar transmitted signal," *Physical Review Letters*, **76**, pp. 904–907, 1996.
[25] Zhou, C. S. and Lai, C. H. "Extracting messages masked by chaotic signals of time-delay systems," *Physical Review E*, **60**, pp. 320–323, 1999.
[26] Agrawal, G.P. *Nonlinear Fiber Optics*, Academic Press, San Diego, 1989.
[27] Breck, C. *Understanding Laser Technology*, PennWell Publishing, Tulsa, Oklahoma, 1985.
[28] van Tarwijk, G. H. M. and Lenstra, D. "Semiconductor lasers with optical injection and feedback," *Quantum Semiclassical Optics*, **7**, pp. 87–143, 1995.
[29] Gray, G. R., Huang, D. *et al*. "Chaotic dynamics of semiconductor lasers with phase-conjugate feedback," *Physical Review A*, **49**, pp. 2096–2105, 1994.
[30] Homar, M., San Miguel, M. *et al*. "Semiconductor lasers with weak optical feedback: spectral properties and frequency-dependent losses," *Optics Letters*, **18**, pp. 1329–1331, 1993.
[31] Lenstra, D., Verbeek, B. H. *et al.* "Coherence collapse in single-mode semiconductor lasers due to optical feedback," *IEEE Journal of Quantum Electronics*, **21**, pp. 674–679, 1985.
[32] Lenstra, D. "Theory of a diode laser with phase-conjugate feedback," *Optics Letters*, **17**, pp. 1590–1592, 1992.
[33] Lenstra, D., van Tartwijk, G. H. M. *et al.* "Multi-wave mixing dynamics in a diode laser," in *Chaos in Optics*, SPIE 2039, pp. 11–22, 1993.
[34] Mork, J., Tromborg, B. *et al.* "Chaos in semiconductor lasers with optical feedback: theory and experiment," *IEEE Journal of Quantum Electronics*, **28**, pp. 93–108, 1992.
[35] Liu, Y. and Kikuchi, N. "Controlling dynamical behavior of a semiconductor laser with external optical feedback," *Physical Review E*, **51**, pp. 2697–2700, 1995.
[36] Ryan, A. T., Agrawal, G. P. *et al.* "Optical-feedback-induced chaos and its control in multimode semiconductor lasers," *IEEE Journal of Quantum Electronics*, **30**, pp. 668–679, 1994.
[37] Lang, R. and Kobayashi, K. "External optical feedback effects on semiconductor injection laser properties," *IEEE Journal of Quantum Electronics*, **16**, pp. 347–355, 1980.
[38] Haken, H. "Analogy between higher instabilities in fluids and lasers," *Physics Letters A*, **53**, pp. 77–78, 1975.
[39] Lorenz, E. N. "Deterministic nonperiodic flow," *Journal of the Atmospheric Sciences*, **20**, pp. 130–141, 1963.

[40] Weiss, C. O. and Brock, J. "Evidence for Lorenz-type chaos in a laser," *Physical Review Letters*, **57**, pp. 2804–2806, 1986.
[41] Weiss, C. O. and Vilaseca, R. *Dynamics of Lasers*, VCH, Weinheim, 1991.
[42] Annovazzi-Lodi, V., Donati, S. and Manna, M. "Chaos and locking in a semiconductor laser due to external injection," *IEEE Journal of Quantum Electronics*, **30**, pp. 1537–1541, 1994.
[43] Guemez, J. and Matias, M. A. "Modified method for synchronizing and cascading chaotic systems," *Physical Review E*, **52**, pp. 2145–2148, 1995.
[44] Kapitaniak, T. "Synchronization of chaos using continuous control," *Physical Review E*, **50**, pp. 1642–1644, 1994.
[45] Gutierrez, J. M. and Iglesias, A. "Synchronizing chaotic systems with positive conditional Lyapunov exponents by using convex combinations of the drive and response systems," *Physics Letters A*, **239**(3), pp. 174–180, 1998.
[46] Lu, H. and He, Z. "Synchronization of chaotic systems by system separation approach," *Physics Letters A*, **219**, pp. 271–276, 1996.
[47] Angeli, A., Genesio, R. and Tesi, A. "Dead-beat synchronization in discrete-time systems," *IEEE Transactions on Circuits and System*, **42**, pp. 54–56, 1995.
[48] de S. Vieira, M., Lichtenberg, A. J. *et al.* "Synchronization of regular and chaotic systems," *Physical Review A*, **46**, pp. 7359–7362, 1992.
[49] Kocarev, L. and Parlitz, U. "Synchronizing spatiotemporal chaos in coupled nonlinear oscillators," *Physical Review Letters*, **77**(11), pp. 2206–2209, 1996.
[50] Mozdy, E., Newell, T. C. *et al.* "Synchronization and control in a unidirectionally coupled array of chaotic diode resonants," *Physical Review E*, **51**, pp. 5371–5376, 1995.
[51] Guemez, J. and Matias, M. A. "Synchronous oscillatory activity in assemblies of chaotic model neurons," *Physica D*, **96**, pp. 334–343, 1996.

# Index