Chapter 7: The motors of the robot

- Learn about different types of motors
- Learn to control different kinds of motors using open-loop and closedloop control
- Learn to use motors in robot building

7.1 Introduction

In this chapter we will discuss three types of motors, namely direct current (DC) motors, servomotors and stepping motors.

- DC motors: A DC motor with a gearbox can offer a large torque that is suitable for the motor drive system for carrying the robot around. It is analogous to the engine of a land vehicle. We will discuss the open-loop and closed-loop methods for controlling the speed of a DC motor.
- Servo motors: On the other hand, servo motors are positional motors that the rotational angle of it shaft can be controlled precisely to a particular angular position but limited to a certain range within 360 degrees, therefore it is suitable for carrying sensors (e.g. ultra-sonic radar) for pointing to different directions. It is analogous to the human neck supporting our head that it can move to a particular angle within a certain range with precision. In fact servomotors with high torque are not limited for carrying sensors, it can also be the power drive system to carry the robot around in the case of a legged robot as shown below. We will talk about legged robot later in this chapter.



Figure 7. 1: 6-leg crawling robots, from <u>http://www.ai.mit.edu/projects/hannibal/hannibal.html</u> & <u>http://www.lynxmotion.com</u>)

• Stepping motors: A stepping motor can rotate to a fixed angular position according to the input pulses, so it is very suitable for digital controlled machines. We will talk about methods for controlling such motors and also discuss the advantages and disadvantages of using these motors.

7.1 DC motors

The DC motor we are using is small but can run at high speed (~ 2000 rpm). It operates on a 3~5Volt source and drains about 300~500m current when in motion. With the help of a gearbox (ratio 58:1), the torque can be increased but the speed is reduced. The main reason for choosing such motors is that they are low cost and powerful enough for our robot, and moreover it is widely available in most model shops.

As said before the current supply for a DC is about 300~500mA (1A during start-up) which is quite large for a digital system, say an I/O pin at the 8255 (typically 2.5 mA), therefore the following issues are needed to be solved.



Figure 7. 2 A DC motor (FA130); The motors, gearbox(58:1) and wheel of our robot

- A current driver circuit is required to increase the current from an I/O of a digital IO pin (~2.5mA) to 500mA required by a motor.
- Develop ways to control effectively the rotational direction and velocity of the motors.
- To have such a large current system in the system it is bound to create noise when the current is switched on and off at the power source. It may affect other digital circuits that drain current from the same power supply. Therefore a separate power system is designed for the DC motors to avoid noise interference.

We will deal with the problems above, and the structure of the following sections are summarized below:

- Overview of the robotic system
- H-bridge circuit for producing large current for motors.
- How to use pulse-width modulation for controlling the speed of the motors.
- How to use open-loop and closed-loop methods for speed control
- How to use servo motors and stepping motors



7.2 Overview of the mobile robot system

Figure 7.3 The robot with motors, and motor drive block diagram for the left motor

Similarly there is a right motor drive system for the motor on the right hand side, but they can share the same power stabilizer to save cost. For the left motor there are two control bits: LDIR, LEN for controlling the rotational direction and enable motor on/off, respectively. For the motor on the right the corresponding bits are RDIR and REN.

Exercise 7. 1Why does the robot use front wheel drive? Exercise 7. 2 Estimate how many power stabilizer systems are needed for the whole system. Draw a block diagram showing the power stabilizer circuits and the sub-modules of the robot systems including DC motors, Servomotors, 8031SBC, and an ultra-sonic radar system.

7.3 Power electronic system for the DC motors:

- Mechanical and solid state relay
- Power transistor
- H-Bridge circuit

7.3.1 Relays – mechanical and solid state types

We can use a simple mechanical replay to turn on or off a DC motor. A mechanical relay consists of a two-way switch that can be turned on or off by a solenoid as shown below. When the input is low, the spring will pull the switch upwards and A-B will be connected. However, when the input is high, the current passing through the solenoid will attract the magnet downwards hence the point A-B will be disconnected and B-C will be joined together.



Figure 7. 4: two views of a mechanical replay



Figure 7. 5 A mechanical replay setup

The major advantage of the mechanical replay is its simplicity and low cost. Because a small solenoid is powerful enough to energize a large switch, hence a small current input can be used to turn on a high power device. It is widely used in electrical appliances, such as televisions, monitors, or even washing machines where very high current switching is employed. Many commercial available relays have rating printed on the device; they can range from a few Amperes to hundreds of Amperes.

However, such devices have also their shortcomings, the mechanical contact point of the switch ionizes the carbon in air each time it switches on, so after some time the carbon deposited may be accumulated to an amount to block current flow.

Also the mechanical switching time is too slow for fast switching applications; hence it is not suitable for generating pulse width modulation signals.

There are also solid-state relays that work on the solid state MOS switching principle, but they are more expensive. Students may find details of these products in the web.

7.3.2 Power transistors

In order to implement pulse-width-nodulation, power transistor for DC motor switching is a good choice. The main idea is to amplify the current from the input and passes the large current output to energize the motor. We used some design found in the reference book *C* and 8051 building efficient applications by Schultz, Prentice Hall [2]; some of the circuit diagrams shown below are also taken from that reference.

• One direction current drive: If we only need the motor to turn to one direction only we have four different designs here.





• Two directional drive circuit: If we need the motor to turn to both rotational directions we need the following diagram. It is also called the H-bridge circuit



Figure 7.7: H-bridge Bi-directional drive

• Analog speed control: Apart from the pulse-width-modulation method discussed in the last chapter, we can also control the speed of a motor by the analog method,



Figure 7. 8: Analog variable –speed drive

7.3.3 The L293D H-bridge circuit package

The H-bridge current driver device L293 is ideal to be used in our small robots. It is small size, low cost and very efficient. It can drive up to 2A of current, which is quite enough for our application.



Figure 7. 9 The use of a H-bridge circuit L293D for dual motor control

The signal names and the H bridge circuit

- L293D: H-bridge circuit, one for two motors, up 2A
- LDIR: left motor direction; RDIR: right motor direction
- LEN : left motor enable; REN : right motor enable

The idea of the H-bridge circuit is very simple: Using the left motor for the example, when LEN is 0, all drivers for the left motor are disabled so that the motor is not moving. When LEN is 1, then LDIR determines the polarity of the voltage applied to the motor, so it governs the rotational direction of the motor. In our control program we only control 2 bits for the left motor: one bit for the direction (LDIR); another bit is the pulse width

modulated signal to the left motor to control its speed. The same applies to the motor on the right hand side.

Here, the drivers in L293D are special power electronic circuits that raise the current 100 times to increase output power.

Exercise 7. 3: Give the states of LEN, LDIR, REN and RDIR for turning the robot clockwise. Exercise 7. 4: What alternatives other than PWM are available for controlling the speed of the DC motors?

7.4 Open-loop motor control and its problem

- Change motor supply power change speed
 - Problem: How much power is right?
 - Problem: How to make the robot walk straight?
- How to control power (Ton) by ISR & 8253?
- Solution: Use feedback control to read actual wheel,
 - Slower, increase power (+ Ton)
 - Faster, reduce power (- Ton)

Using the pulse width modulation program we developed earlier in the last chapter we can now make the motor drive to rotate at a required speed.



Figure 7. 10 Figure: An open loop motor drive

Using the following ISR (isr_left_pwm())to drive the motor, we can control roughly the speed of the left motor but would not be too accurately.

int T_on; isr_left_pwm2() //interrupted at 64Hz
{
 set 8253 with Ton = T_on;
}

This is called "open loop motor control". That means we control the supply of the energy to the motor hoping that it will rotation at a speed we expected. This method has a problem that we don't know how much energy we should give to the motor to make it rotates at a certain speed. And even more serious is that, a motor is not always at a fixed efficiency level, thus the speed may change even you give it the same energy at different times. It becomes a very serious problem especially when the differential-wheeled robot (a robot uses left-right wheels powered separately) we designed needs to move in a straight line, because the two motors are not turning at the same speed. In other words, we don't know how much we should power the left and the right motors to make them behave the same.

To overcome this problem, a technique called closed-loop feedback control is needed which requires the system to read back the rotational speed of the motors and determines how much energy (PWM pulse width Ton) the controllers should deliver to them. To achieve this goal we do need to know the rotational speed of the wheel, which is the theme of the section on closed-loop control.

Exercise 7. 5: When using the open-loop control method with a constant PWM signal for both wheels, explain why the robot would slow down when climbing up hill.

7.4.1 Experiment for open loop motor control hardware counter method Based on the ISR (interrupted at 64Hz) shown above, write a program to control the

Based on the ISR (interrupted at 64Hz) shown above, write a program to control the motor. The PWM signal control levels are 64 steps, and use the following program structure and variable names. (No need to drive motor; observe the waveform by a Oscilloscope at 8253 out0)

int T_on; //no need to drive motor; observe the waveform by a Oscilloscope at 8253 out0
isr_left_pwm2() //interrupted at 64Hz
{

set 8253 with $Ton = T_on$;

Procedure:

- 1. Design an interrupt based, hardware counter method (PWM type2 using 8253) for generating a PWM signal for the left motor: Interrupt rate 64Hz, control steps is 64.
- **2.** Feed the PWM signal to the H-bridge (L293) for driving the left DC motor.

7.5 Infrared Red (IR) wheel speed encoder

- Used to read actual wheel speed
- Use photo interrupter
- Use reflective disk to safe space
 - Based on 8255, 8031SBC interrupts

In order to have precise control of the motor, we need to know the actual velocity of the wheel. This can be achieved by using an infrared photo-interrupter as the speed encoder.



Figure 7. 11: The picture of the speed encoder of our robot



Figure 7. 12:Read back rotating speed of a wheel by IR interrupter

As we can see that when the motor turns, the disk also rotates and chops the light on-andoff periodically, by measuring the pulses at the output of the IR detector we can deduce the speed of the motor. However if you are developing a small robot, sometimes it is not easy to fit two IR photo-interrupters in the chassis hence you can adopt an alternative method -- the reflective disk method, which would save some space.



Figure 7. 13: The rotating wheel speed encoder of our robot

The reflective disk method is done by placing the IR transmitter and receiver pointing to the same direction towards a disk. The disk is painted with a pattern shown in the diagram. When the disk rotates, the I.R. receiver will receive on-off pulses corresponding to the angular position of the disk. The other hardware needed is the 8255 for receiving the IR receiver signal, and the 8031 is interrupted regularly and uses the ISR for measuring the wheel speed.

7.5.1 IR encoder circuit and program

Now we will discuss how the chopping of the light can result into a value representing the speed of the motor. As you can see from the above diagram that the IR receiver is connected to the 8255 of our system, it is a good point to begin our discussion because the system can read in the chopping light information. Now we need a program to turn this information into a variable inside our program. Because precise timing is required, it is no wonder an interrupt service routine is definitely required. In the following interrupt service routine with interrupt rate of 1024Hz, the unsigned char "wsl_speed_measured" will represent the actual speed of the wheel on the left-hand side.

In the following ISR program for the left wheel, we need to do the following.

- Wsl_counter is a counter to keep track of how many changes of pattern of the wheel disk has been detected since wsl_count is last reset.
- At the beginning of every 1/4 seconds (after 256 interrupts) wsl_count will be stored in wsl_speed_measured and wsl_count is reset to 0.

```
Unsigned char wsl speed measured;
// wsl_speed_measured=a global variable (pattern changes per 1/4 seconds) of left wheel
//timer counter : an unsigned char
main
 {
     set isr_read_actual_lspeed() interrupt rate at 1024 Hz
     do_something;
}
isr_read_actual_lspeed() // ISR example for the left wheel; interrupt at 1024 Hz
 {
     unsigned char wsl_count; //keep track of the change of pattern since wsl_count reset
     timer_counter++; // it is a global 8-bit unsigned char
     if ( the IR receiver reports pattern change) //from black to white or vice versa
     {
        wsl count++;
       if (timer_counter = = 0)
       //meaning if( ISR executed 256 times after last reset occurs); happen once in 0.25s
        {
              wsl_speed_measured = wsl_count; //integrated result
              wsl count = 0;
 }
Exercise 7. 6: What are the disadvantages of using the reflected IR disc method for building an
```

Exercise 7. 7: The wheels disk has 16 white and 16 black strips, what is the relation between wsl_speed_measured and rotation per second of the wheel?

7.5.2 Experiment 7 1, Program to read the speed of the wheels

motor speed encoder.

- 1. Design the IR wheel speed encoder circuit.
- 2. Write an program to read the speed of the left wheels

isr_read_actual_lspeed(), results stored at actual_lspeed

7.6 Closed-loop feed back control system

7.6.1 A simple closed-loop control system

As we have discussed before, in order to achieve precise speed control, a closed loop feedback system is necessary. A simple method is show below. For example, we have a certain required speed "wsl_speed_input", say 10 units for the robot, if the "wsl_speed_measured" is lower than 10, it increases power by adding 1 unit to "wsl_t_on", otherwise it decreases power by subtracting 1 unit from "wsl_t_on". It is effective and simple, but stability is a may be a problem if we look at the following example.

If the initialize value of "wsl_speed_measured" is low, say 5 and we want "wsl_speed_input" to be 10. Power will be increased and it adds 1 to wsl_t_ton and the wheel will turn faster. Then we will have the following two scenarios happening alternatively:

- Because of certain mechanical delay the speed of the wheel will not change to 10 instantly as required. But the control system sees that delta_speed is still negative, hence it will increase power again. With a few computational cycle, "wsl_t_ton" will be increased many times and reaches to the top level. So the system will exert full power to increase the speed from 5 to 10.
- Later, when "delta_speed" finally becomes negative (wsl_speed_measured > wsl_speed_input), say 11, the system will decrease "wsl_t_on" by 1, but the wheel is not moving slower because of inertia, hence after a few computational cycle "wsl_st_on" will decrease to 0. So the system will exert no power (wsl_t_ton=0) to make the wheel to return from 11 to 10.

The above two scenarios will happen alternatively. The frequency of the occurrence will increase as "wsl_speed_measured" and "wsl_speed_input" are close, hence the wheels will jerk. If we look at the cause of the problem, we found that the system may be too sensitive, it is responding too quickly. Hence we conclude that if a system is too sensitive, it is unstable. Do you find the same thing happen to humans as well?



Figure 7. 14: A simple closed loop feedback control of the left motor speed



Figure 7. 15: If the feedback loop is too fast, the system is unstable. It is now turned into a bang-bang (a bad name for unstable systems) control system.

7.6.2 Increase stability by using delay

There are many methods to make a system more stable. For example we can add delay to the feedback path, such as to add a rule saying that if "wsl_speed_measured" is not changed, we keep the current power "wsl_t_ton" level. It is effective because the system would not change "wsl_t_on" so rapidly. However we may have some other problems, such as, the speed may be locked into a local minimum; it will not change speed at a certain value. A case in point is that if "wsl_speed_measured" is initially at 0 speed, it may not change forever. Think about the reason! Students can experience this when they perform the experiments. Of course some methods for fixing this deficiency can be devised and it will be left for an exercise for readers.

//The simplified algorithm for feedback control of the left wheel of desired speed //wsl_speed_input: the desired moving speed //wsl_speed_measured: actual speed measured

//wsl_t_on: Ton time of the left wheel

```
//A simple control algorithm with stabilizer
Main() //only left motor control is shown, a simplified algorithm
{
    Initialize wsl_t_on ;
   Loop forever
      { change "wsl_speed_input" to the desired speed by user or program;
         Do_something();
       }
}
//combined both isr_left_pwm1 and isr_read_actual_lspeed()
Isr left feedback() //interrupted at 1024Hz
    generate PWM Ton according to "wsl_t_on", see chapter 5;
{
   read "wsl_speed_measured", see isr_read_actual_lspeed( ).
    If (wsl_speed_measured has changed since last interrupt) //stabilize the system
    {
        if (wsl_speed_measured < wsl_speed_input)
              increase Ton by "wsl_t_on= wsl_t_on+1";
        else
              decrease Ton by "wsl t on= wsl t on-1";
    }
}
```

7.6.3 Increase stability by using proportional closed-loop control



Exercise 7. 8 Closed feed back control of the left motor speed

//The proportional feedback control method //The actual code is not available yet, students should write their own, // //Wsl_speed_input: the desired moving speed //Wsl_speed_measured: actual speed measured //wsl_t_on: Ton time

K1, K2, delta_speed are floating point numbers

```
Main() //only left motor control is shown, a simplified algorithm
{
    Initialize wsl_t_on ;
    Loop forever
      { change "wsl_speed_input" to the desired speed by user or program;
         Do_something();
       }
}
______
// K1,K2 are floating point numbers, by careful selecting K1,K2 we can have
    a stable and responsive control system.
//
//
//combined both isr_left_pwm1 and isr_read_actual_lspeed()
Isr_left_feedback() //interrupted at 1024Hz
    generate PWM Ton according to "wsl_t_on", see chapter 5;
{
    read "wsl_speed_measured", see isr_read_actual_lspeed( ).
    If (wsl_speed_measured changed above a limit since last interrupt)
    // the above may be used to stabilize the system
    {
         delta_speed = wsl_speed_input - (wsl_speed_measured) * K2
         ;
         wsl_t_on = wsl_t_on + delta_speed * K1;
    }
}
```



Figure 7. 16: Proportional feedback will enable the system to reach the desired speed in a more stable manner.

Proportional control is a formal method to make a system more stable as well responsive. In the system block diagram, the user can input the desired speed "wsl_speed_input" to the system, for example through the keyboard. If the motor is rotating slower then expected, then "delta_speed" should be positive, thus we proportionally increase the PWM pulse on-time Ton "wsl_t_on". The is the increase "wsl_t_on" proportional to K1* {wsl_speed_input - K2* (wsl_speed_measured) }.

We will do the opposite if "delta_speed" is negative. And this is the basic concept of a proportional closed-loop control system, which is used widely in many different fields of engineering, for example, space crafts, motor cars, rockets etc.

It is noted that in order to have accurate control, the variables used in this system should be in floating point format so that enough data accuracy can be utilized.

Exercise 7. 9: Write the algorithm for making the robot move straight in the forward direction at a constant speed. (Hint: (1) set two wheels at the same speed, or (2) compare speed of two wheels, comment on the differences.)

Exercise 7. 10: *Modify the algorithm in the above question to make the robot to follow a circular path of radius R1.*

7.7 Control theory

The above control system is a simple but effectively one, however, by selecting the forward and feedback gain (K1,K2 resp.) factors carefully, we can achieve an effective and stable control system. However, this is only a brief example, interested students should consult books on control theory for more in depth discussions of this issue.

7.8 Servo motor system

The servomotor motor is a positional motor system that carries a load accurately to a particular rational angle. We have discussed the pulse width modulated signal for a servomotor, please refer back to the chapter on PWM for details.

In our robot we will use it to carry an ultra-sonic radar system. Since the radar system can only detect obstacles directly ahead of its transmitters and emitter, to widen its viewing scope, we can put the radar system on top of a rotational table for pointing it to different directions.



Figure 7. 17 : Application of a servomotor in a ultra-sonic radar system

Another application of servomotors is use them for making the join of an insect like the legged robot as shown in figure 1. In such a robot, each leg consists of three servomotors enabling the leg to perform complex motion. By carefully setting the legs to move synchronously we can move the robot just like a crawling insect. However, it is a highly complex system because the control mechanism involves so many degrees of freedom. Hence it is still a research problem for many researches all over the world, interested students should find more information from the Internet.

7.9 Stepping motors and control

Stepping motors are good positional control motors used widely in digital controlled machineries such as digital arm-clocks, printers, disk drives etc.



Figure 7. 18 small stepping motors

The shaft of a stepping motor can rotate to precise angular positions according to the input control signals. For a small 4-phase stepping motor as shown above, there are 5 excitation input lines including in0, in1, in2, in3 and a common (com). The interface circuit is as follows.

Even for the small stepping motors shown above, the requirement of current for each excitation input is rather high (over 500mA), so a power electrical driver circuit is necessary.

In our experiment we used a L293 H-bridge circuit (in fact 74245 current driver is more cost effective) and four TIP3055 transistors to boost up the current and voltage from the TTL to the required level.



Figure 7. 19: Steeping motor interface circuit

The method of making the motor to rotate is to give a step pattern to the inputs of the motor at a regular time interval. The motor will rotate according to the step rate. Therefore the motor rotating speed can be controlled precisely. Also if the sequence is reversed the motor will rotate to the opposite direction. It is shown that there are two methods for excitation: (1) full step method; (2) half step method.

Туре	Step index	pb0	Pb1	Pb2	Pb3	Code
Full wave	1	0	0	1	1	0011=0x03
stepping	2	0	1	1	0	0110=0x06
	3	1	1	0	0	1100=0x0c
	4	1	0	0	1	1001=0x09
Half wave	1	0	0	0	1	0001=0x01
stepping	2	0	0	1	1	0011=0x03
	3	0	0	1	0	0010=0x02
	4	0	1	1	0	0110=0x06
	5	0	1	0	0	0100=0x04
	6	1	1	0	0	1100=0x0c
	7	1	0	0	0	1000=0x08
	8	1	0	0	1	1001=0x09

table 7 1 Stepping motor excitation sequence

Generating the excitation pulse sequence can be done by either software delay method or interrupt method. Of course the interrupt method should give better accuracy and controllability. Students should work out the algorithms themselves.

Excitation method	Advantages	Disadvantages
Full step	Shorter sequence	Oscillate, relatively unstable.
	Higher torque	Easy to slip steps when excitation time step is too short; slower rotation speed limit
Half step	Longer sequence Lower torque	Smooth rotation, relative stable Higher rotation speed limit

table 7 2 The relative advantages and disadvantages are shown below

A major problem for stepping motors is their stepping rate limit. There is a relatively lower rotating speed limit compared to that of the DC motors. Because when the stepping

rate is too fast, the stepping motor will slip steps and sometimes results in local oscillating at the same position.

Therefore the main advantage of stepping motors is their precision in speed and positional control because using a simple digital control sequence one can control the speed and rotation position precisely. While, for a DC motor, a more complex feedback control setup must be used for precise speed or positional control as discussed before.



Figure 7. 20 : (Left)Top,(Middle) side views of a micro-mouse using 2 stepping motors. (Right) The standard maze of a micro-mouse competition, the mouse is at the far right corner of the picture.(See also http://www.ai.com.sg/products.htm)

7.10 Method to solve the step-slipping problem

Because of the advantages of easy control, stepping motors are used widely for building micro-mouse robots for micro-mouse competitions. For example we can change the stepping rate of the input sequence to control the speed of the motor, however, if the change is too rapidly, some controlling steps may be missed (step-slipping). It is higher undesirable in micro-mouse control, because the mouse may lose track of where it is. To overcome this, rotation profile is used. The idea is illustrated in the following diagrams. It is shown that if we increase the rotating step too sudden the stepping motor may slip some steps or may not rotate at all. Using a profile method to increase or decrease the rotating speed gradually the problem can be solved. The precise profile parameter depends on the motor and its loading so it must be worked out by actual experimentation.



Figure 7. 21 : The method of using a speed profile to solve the step-slipping problem

7.11 Conclusion

In this chapter we studied the methods of controlling the speed of a DC motor. First an open-loop method is discussed then followed by a more precise closed-loop feedback method. Finally applications and use of servomotors and stepping motors for robot engineering are covered.

7.12 Reference

- 1. Data sheet of H-bridge driver circuit device by Texas instrument http://www.ti.com/sc/docs/products/analog/1293.html#Datasheets
- 2. Schultz, C and 8051 building efficient applications, Prentice Hall

з.