

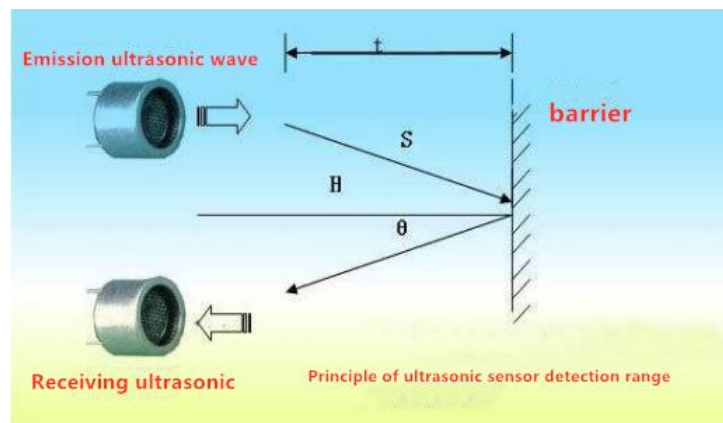
Lesson 5 Introduce Automatic Obstacle Avoidance

In this lesson, we will learn how to use PiCar-b to achieve automatic obstacle avoidance.

5.1 Introduction to Automatic Obstacle Avoidance

Since the camera used by our Raspberry Pi robot is a monocular camera and cannot collect depth information, many of our robot products use ultrasonic ranging modules to obtain depth information and detect whether there are obstacles in a certain direction to get the distance of the obstacle.

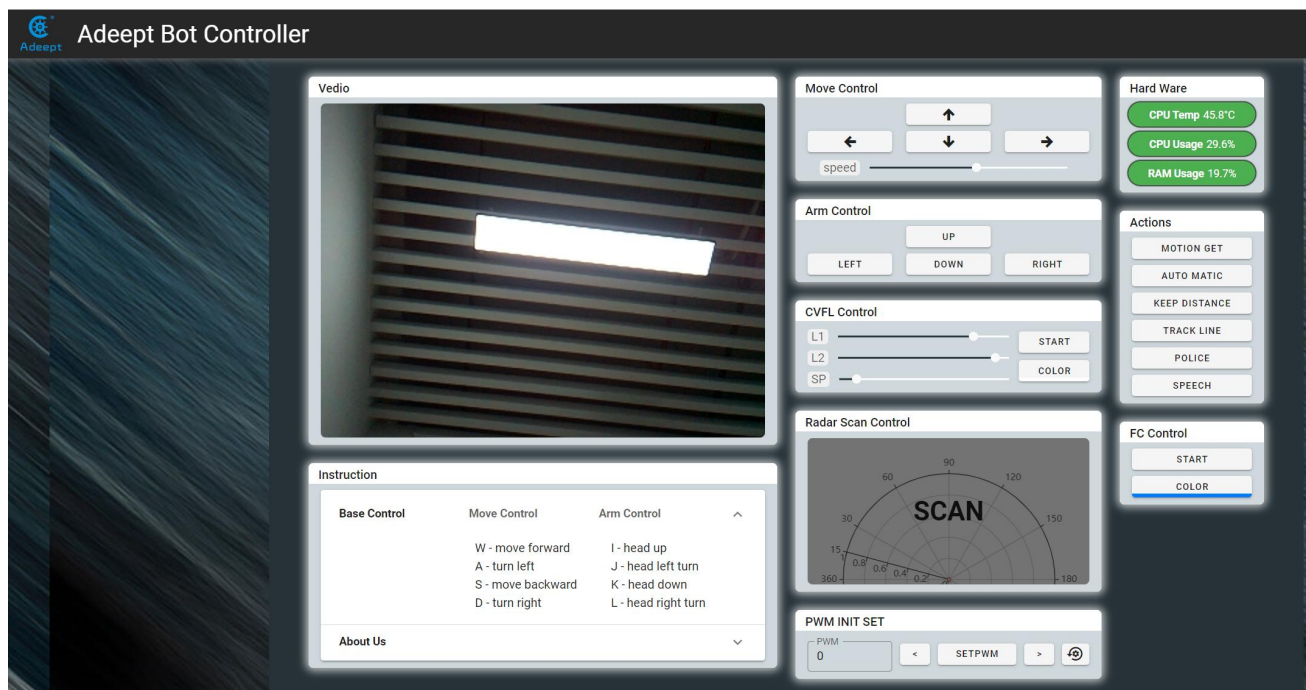
The principle of distance detection by ultrasonic ranging sensor: the method of detecting distance by ultrasonic is called echo detection method, that is, the ultrasonic transmitter emits ultrasonic waves in a certain direction, and the timer starts timing at the same time as the launch time. The ultrasonic waves propagate in the air and encounter obstacles on the way. When the object surface (object) is blocked, it will be reflected back immediately, and the ultrasonic receiver will immediately stop timing when the reflected ultrasonic wave is received. The propagation speed of ultrasonic waves in the air is 340m/s. According to the time t recorded by the timer, the distance s from the launch point to the obstacle surface can be calculated, namely: $s=340t/2$. Using this principle of ultrasound, the ultrasonic ranging module is widely used in practical applications, such as car reversing radar, unmanned aerial vehicle, and smart car.



5.2 Turning on Automatic Obstacle Avoidance

Running the Automatic Obstacle Avoidance program

1. Turn on PiCar-b, the boot time is about 30s-60s.
2. After PiCar-b is turned on, enter the IP address of your Raspberry Pi with the Google browser of your mobile phone or computer, and access port 5000, for example: 192.168.3.44:5000 . The web controller will then be displayed on the browser.



3. Place the car on the prepared patrol track.

4. After clicking "**AUTO MATIC**", the robot will automatically avoid obstacles when encountering obstacles.

5. When you want to terminate the Automatic Obstacle Avoidance function, you can click "**AUTO MATIC**" again.

5.3 Main Code

For the complete code, please refer to the file [functions.py](#).

```
01 def automaticProcessing(self):
02     print('automaticProcessing')
03     dist = self.distRedress()
04     print(dist, "cm")
05     time.sleep(0.2)
06     if dist >= 40:          # More than 40CM, go straight.
07         scGear.moveAngle(0, 0)
08         move.move(80, 1, "mid")
09         print("Forward")
10     # More than 20cm and less than 40cm, detect the distance between the left and right sides.
11     elif dist > 20 and dist < 40:
12         move.move(0, 1, "mid")
13         scGear.moveAngle(1, 30)
14         time.sleep(0.3)
15         distLeft = self.distRedress()
16         self.scanList[0] = distLeft
17
18     # Go in the direction where the detection distance is greater.
19     scGear.moveAngle(1, -30)
20     time.sleep(0.3)
21     distRight = self.distRedress()
22     self.scanList[1] = distRight
23     print(self.scanList)
24     scGear.moveAngle(1, 0)
25     if self.scanList[0] >= self.scanList[1]:
26         scGear.moveAngle(0, 30)
27         time.sleep(0.3)
28         move.move(80, 1, "left")
29         print("Left")
30         time.sleep(1)
31     else:
```

```
32         scGear.moveAngle(0, -30)
33         time.sleep(0.3)
34         move.move(80, 1, "right")
35         print("Right")
36         time.sleep(1)
37     else:         # The distance is less than 20cm, back.
38         scGear.moveAngle(0, 0)
39         move.move(80, -1, "mid")
40         print("Back")
41         time.sleep(1)
```