Lesson 15 Obstacle Avoidance

Table

1. Principle 1
2. Upload the code 4
3. Troubleshooting 5
4. Code 5
5. Any questions and suggestions are welcome 5

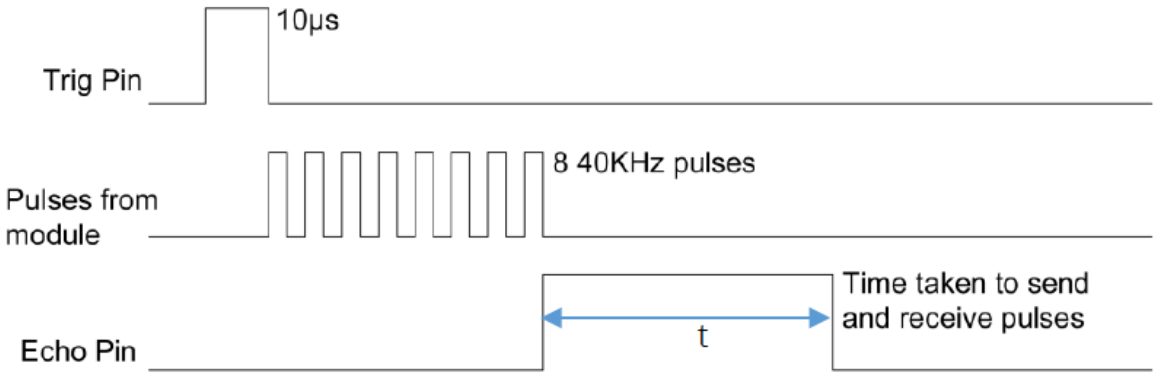
In Lesson 3, we have learned the basics of the ultrasonic module and its Arduino-based control logic.

Now let's write code to control the car to avoid obstacles automatically.

1. Principle

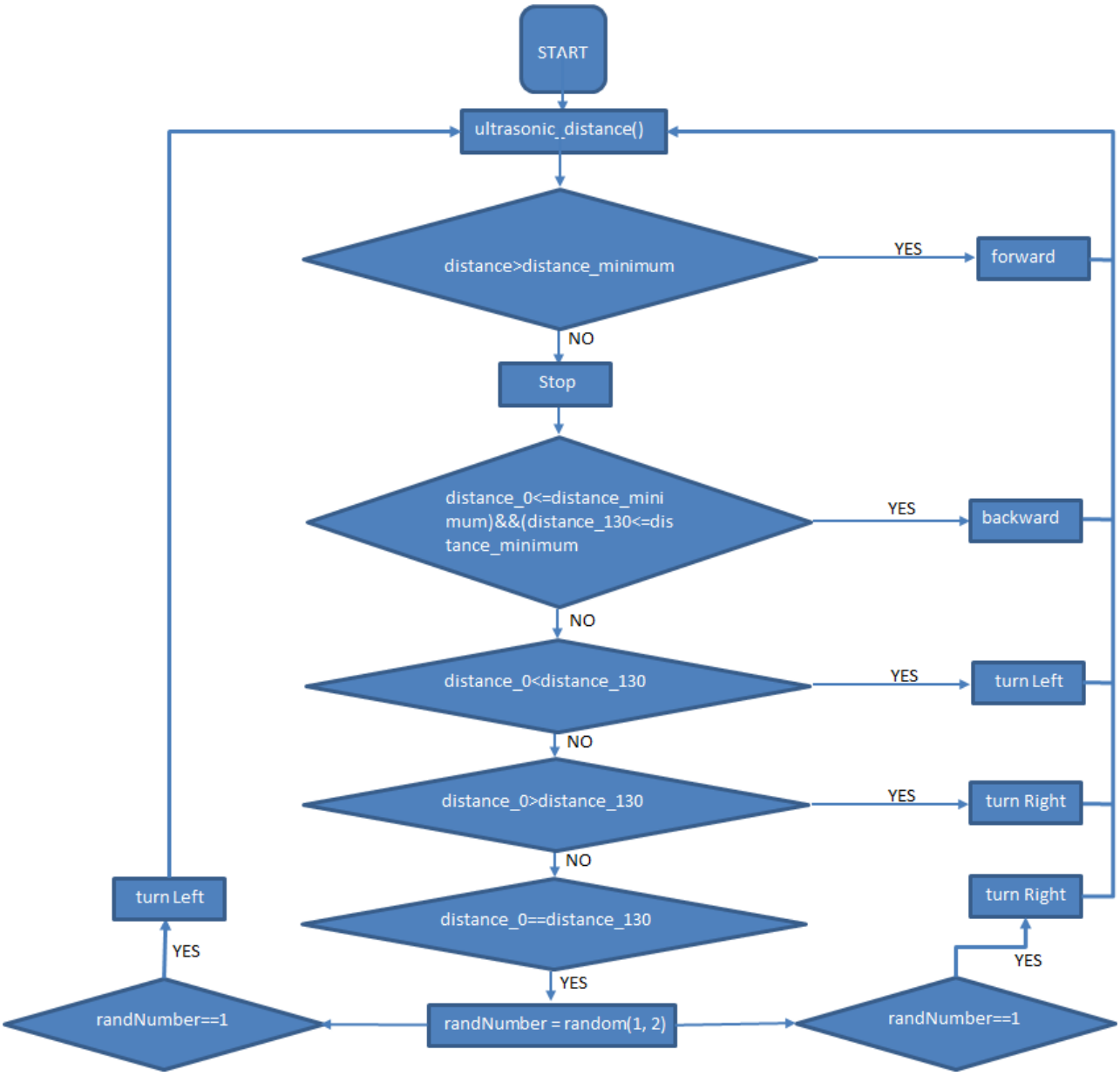
The HC-SR04 Ultrasonic Ranging Module integrates a both an ultrasonic transmitter and a receiver. The transmitter is used to convert electrical signals (electrical energy) into high frequency (beyond human hearing) sound waves (mechanical energy) and the function of the receiver is opposite of this.

Output a high-level pulse in Trig pin lasting for least 10uS, the module begins to transmit ultrasonic waves. At the same time, the Echo pin is pulled up. When the module receives the returned ultrasonic waves from encountering an obstacle, the Echo pin will be pulled down. The duration of high level in the Echo pin is the total time of the ultrasonic wave from transmitting to receiving, s=vt/2. This is done constantly.



Then it makes the servo of the robot rotate at different angles, the robot recognizes the distance of obstacles at each angle through the Ultrasonic Module, and then compares the each distance, and then judges whether to go straight, turn left, turn right or back.

The judgment flow chart is as follows:


START
ultrasonic_distance()
distance>distance_minimum
YES
forward
NO
Stop
distance_0<=distance_minimum)&&(distance_130<=distance_minimum
YES
backward
NO
distance_0<distance_130
YES
turn Left
NO
distance_0>distance_130
YES
turn Right
NO
distance_0==distance_130
YES
randNumber = random(1, 2)
randNumber==1
YES
turn Right
randNumber==1
YES
turn Left

2. Upload the code

2.1 The code used in this lesson is placed in this folder：

E:\CKK0002-master\Tutorial\sketches\13_1_Automatic_Obstacle_Avoidance

2.2 Before uploading the code, turn the ESP-01 switch on the control board to the side away from the "ESP-01" silk screen.

2.3 After uploading the code, unplug the USB cable, put the Smart Robot Car on the ground, turn on the power switch on the control board, the car will start to drive and avoid obstacles

2.4 Build a maze-shaped model with a cardboard box or other tools, place the Smart Robot Car at the entrance of the maze, and verify whether the car can automatically run out of the maze

3. Troubleshooting

3.1 Smart Robot Car cannot avoid obstacles

Check whether the ultrasonic module and the control board are correctly and effectively connected

The detection range of the ultrasonic module is limited, and it will not be able to detect obstacles higher or lower than the position of the ultrasonic module. The area of the obstacle needs to be large enough to return the ultrasonic signal.

3.2 The angle of rotation on both sides of the servo is different

Note that you need to set it to 65 degrees before installing the servo

The acrylic of the head is not installed correctly on the servo

4. Code

13_1_Automatic_Tracking_Line.ino：

```
#include <SCoop.h> //Import multithread library
#include <Wire.h>
#include <Servo.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define WS2812_PIN 6     //WS2812 PIN
#define WS2812_COUNT 12   // How many NeoPixels are attached to the Arduino?
#define BRIGHTNESS 10   // NeoPixel brightness, 0 (min) to 255 (max)
// Declare our NeoPixel strip object:
```

```
Adafruit_NeoPixel strip = Adafruit_NeoPixel(WS2812_COUNT, WS2812_PIN, NEO_GRB + NEO_KHZ800);

Servo carservo;
byte carservoOffset = 0;

LiquidCrystal_I2C lcd(0x27,16,2);


#define Buzz 11       //buzzer PIN
#define led_R 9
#define led_L 5

#define Trig_Pin 13 //define Trig pin
#define Echo_Pin 12   //define Echo pin
#define distance_minimum 30     //cm
#define MAX_distance 300     //cm
float distance,distance_0,distance_130;//Import the middle, right, and left distance variables
int randNumber=0;
int L_Distance=0;
int M_Distance=0;
int R_Distance=0;
float cm;

defineTask(TaskOne);      // Create subthread 1
defineTask(TaskTwo);      // Create subthread 2
//////////////////////////////////////////////////////////////////////////////////
void TaskOne::setup()     //Thread 1 setup
{
  Serial.begin(9600);
  pwm.begin();
  pwm.setPWMFreq(50);
  pwm.setPWM(2, 0, 0);
  pwm.setPWM(3, 0, 0);
  pwm.setPWM(4, 0, 0);
  pwm.setPWM(5, 0, 0);

  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(3,0);
  lcd.print("HELLO WORLD!");
  lcd.setCursor(2,1);
```

```
  lcd.print("HELLO COKOINO!");

  pinMode(Buzz, OUTPUT);
  pinMode(Trig_Pin, OUTPUT);
  pinMode(Echo_Pin, INPUT_PULLUP);
  carservo.attach(10);
  carservo.write(65);
}
void TaskTwo::setup(){   //Thread 2 setup
  pinMode(led_R, OUTPUT);
  pinMode(led_L, OUTPUT);
  strip.begin();
  strip.show();
  strip.setBrightness(BRIGHTNESS);
}

void setup(){
  mySCoop.start();//Start multithreading
}
////////////////////////////////////////////////////////////////////////////
void TaskOne::loop() //loop subthread 1
 {

   //LED();//light the green LED
   ultrasonic_distance();
  //Serial.println(distance);
  if(distance>distance_minimum)
    {
    forward();     //forward
    }
  if(distance<=distance_minimum)
    {
    Stopcar();       //stop
    buzz();          //the buzzer sounds
    servo_wheel(); //The servo rotates, and the ultrasonic module identifies the distance
                   //of the 65-degree obstacle in the left front of the car body
    if((distance_0<=distance_minimum)&&(distance_130<=distance_minimum))
      {backward();//backward
      delay(300); }
    if(distance_0<distance_130)//The right obstacle is less distant than the left
      {turnLeft();
      delay(80);
```

```
      }
    if(distance_0>distance_130)//The left obstacle is less distant than the right
      {turnRight();
      delay(80);
      }
    if(distance_0==distance_130)
      {
      randNumber = random(1, 2);   //randnumber
      if(randNumber==1)
        {turnLeft();delay(50);
       }
      if(randNumber==2)
        {turnRight();delay(50);
       }
      }
    delay(250);
    Stopcar();       //stop
    }
}



void forward()
{
 pwm.setPWM(2, 0, -800);
 pwm.setPWM(3, 0, 800);
 pwm.setPWM(4, 0, 800);
 pwm.setPWM(5, 0, -800);
}

void backward()
{
 pwm.setPWM(2, 0, 800);
 pwm.setPWM(3, 0, -800);
 pwm.setPWM(4, 0, -800);
 pwm.setPWM(5, 0, 800);
}

void turnLeft()
{
 pwm.setPWM(2, 0, 1200);
 pwm.setPWM(3, 0, -1200);
```

```
 pwm.setPWM(4, 0, 1200);
 pwm.setPWM(5, 0, -1200);
}
void turnRight()
{
 pwm.setPWM(2, 0, -1200);
 pwm.setPWM(3, 0, 1200);
 pwm.setPWM(4, 0, -1200);
 pwm.setPWM(5, 0, 1200);
}
void Stopcar()
{
 pwm.setPWM(2, 0, 0);
 pwm.setPWM(3, 0, 0);
 pwm.setPWM(4, 0, 0);
 pwm.setPWM(5, 0, 0);
}
//////////////////////////////////////////////////////////ultrasonic ranging
void ultrasonic_distance()
{
delay(100);
digitalWrite(Trig_Pin, HIGH);// make trigPin output high level lasting for 10μs to triger HC_SR04
delayMicroseconds(10);
digitalWrite(Trig_Pin, LOW);
distance = pulseIn(Echo_Pin, HIGH) * 340 /2/ 10000.0;// calculate the distance according to the time
if(distance==0)
   distance=300;
delay(100);
}
//////////////////////////////////////////////////////////servo rotation
void servo_wheel()
{
carservo.write(0);
delay(250);
ultrasonic_distance();
distance_0=distance;
delay(250);

carservo.write(130);
delay(250);
ultrasonic_distance();
distance_130=distance;
```

```
delay(250);

carservo.write(65);
delay(300);
}

void buzz()
{
  for(int i = 0;i < 100; i++)
  {
    digitalWrite(Buzz, HIGH);
    delay(1);
    digitalWrite(Buzz, LOW);
    delay(1);
  }
  for(int j = 0;j< 180; j++)
  {
    digitalWrite(Buzz, HIGH);
    delay(2);
    digitalWrite(Buzz, LOW);
    delay(2);
  }
}

void TaskTwo::loop() //// loop subthread 2
{
  digitalWrite(led_R,HIGH);
  digitalWrite(led_L,HIGH);
  colorWipe(strip.Color(255, 0, 0), 10); // Red
  delay(800);
  colorWipe(strip.Color(255, 150, 0), 10); // yellow
  delay(800);
  colorWipe(strip.Color(0, 255, 0), 10); // Green
  delay(800);
  colorWipe(strip.Color(0, 255, 255), 10); // CYAN
  delay(800);
  colorWipe(strip.Color(0, 0, 255), 10); // Blue
  delay(800);
  colorWipe(strip.Color(180, 0, 255), 10); // purple
  delay(800);
  colorWipe(strip.Color(127, 127, 127), 10); // White
  delay(800);
```

```
  colorWipe(strip.Color(0, 0, 0), 30); // Clear
  Serial.println("OK");
}

void colorWipe(uint32_t c, uint8_t wait)
{
  for(uint16_t i=0; i<strip.numPixels(); i++) {// For each pixel in strip...
    strip.setPixelColor(i, c);                //  Set pixel's color (in RAM)
    strip.show();                             //  Update strip to match
    delay(wait);
  }
}

//Theatre-style crawling lights.
void theaterChase(uint32_t c, uint8_t wait) {
  for (int j=0; j<10; j++) {  //do 10 cycles of chasing
    for (int q=0; q < 3; q++) {
      for (int i=0; i < strip.numPixels(); i=i+3) {
        strip.setPixelColor(i+q, c);    //turn every third pixel on
      }
      strip.show();

      delay(wait);

      for (int i=0; i < strip.numPixels(); i=i+3) {
        strip.setPixelColor(i+q, 0);        //turn every third pixel off
      }
    }
  }
}

void loop(){
  yield();//loop Multithreaded task
}
```

5. Any questions and suggestions are welcome

Thank you for reading this document!

If you find any errors and omissions in the tutorial, or if you have any suggestions and questions, please feel free to contact us:
cokoino@outlook.com

We will do our best to make changes and publish revisions as soon as possible.

If you want to learn more about smart cars, robots, learning kits and other technology products from us, please bookmark and pay attention to our website:

http://cokoino.com/

We will continue to launch interesting, cost-effective, innovative, user-friendly products.

LK COKOINO