

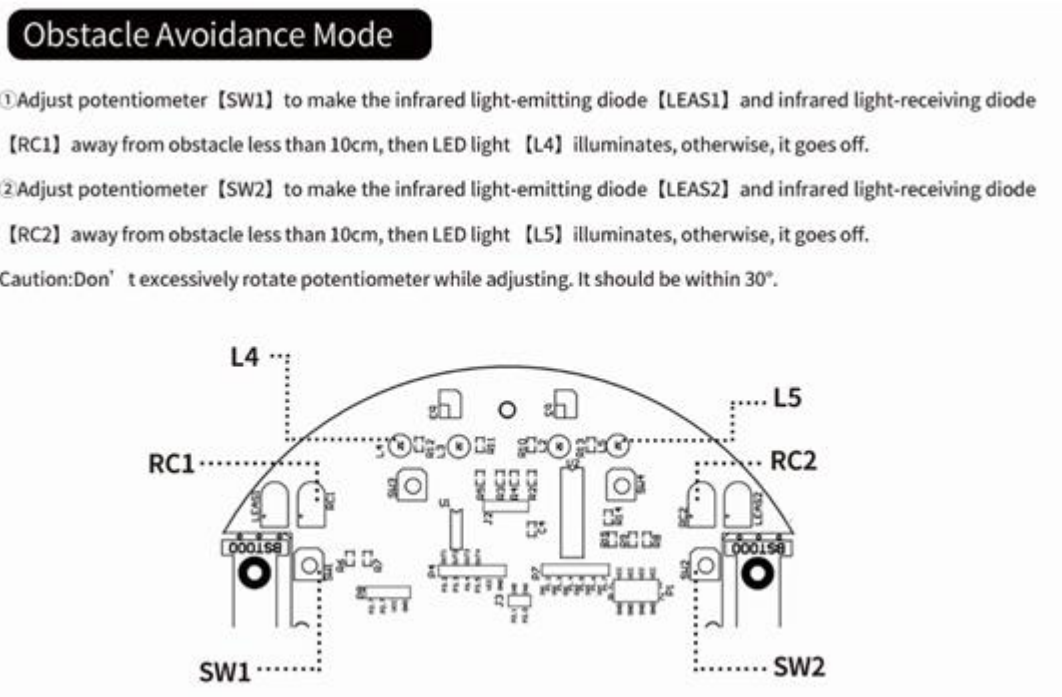
6 Infrared obstacle avoidance obstacle - basic

The purpose of the experiment:

The carton simulates obstacles on the spacious floor, and the smart car that has uploaded the program **avoid.ino** is placed on the ground. After pressing the start button of the car, the car starts to avoid the obstacles in front.

Precautions:

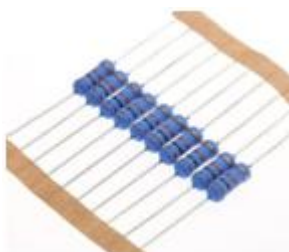
1. Before conducting the experiment, you need to debug the adjustable resistors SW1 and SW2 in front of the smart car, as shown below.



2. This experiment must be carried out in an environment without outdoor light, and it is also necessary to pull curtains to block outdoor light indoors.

List of components required for the experiment:

- Arduino Smart Car* 1
- USB cable* 1
- Active buzzer* 1
- DuPont Line* 13
- Breadboard* 1
- Button * 1
- 10K resistor * 1



Experimental code analysis:

```
//=====yahboom=====
=====
```

```
// Intelligent car infrared obstacle avoidance 1(basic obstacle avoidance)
```

```

//=====================================================
=====
int Left_motor_back=9;    //(IN1)
int Left_motor_go=5;      //(IN2)
int Right_motor_go=6;     //(IN3)
int Right_motor_back=10;  //(IN4)
int key=A0;//Define the key A0 interface
int beep=A1;//Define the buzzer A1 interface
const int SensorRight = A2;    //Right tracking infrared sensor(P3.2 OUT1)
const int SensorLeft = A3;     //Left tracking infrared sensor(P3.3 OUT2)
const int SensorLeft_2 = A4;   //Left infrared sensor(P3.4 OUT3)
const int SensorRight_2 = A5;  //Right infrared sensor(P3.5 OUT4)
int SL;    //Left tracking infrared sensor state
int SR;    //Right tracking infrared sensor state
int SL_2;  //Left infrared sensor state
int SR_2;  //Right infrared sensor state
void setup()
{
    //Initialize the motor drive IO for output mode
    pinMode(Left_motor_go,OUTPUT); // PIN 5 (PWM)
    pinMode(Left_motor_back,OUTPUT); // PIN 9 (PWM)
    pinMode(Right_motor_go,OUTPUT); // PIN 6 (PWM)
    pinMode(Right_motor_back,OUTPUT); // PIN 10 (PWM)
    pinMode(key,INPUT); //Define the key interface for the input interface
    pinMode(beep,OUTPUT);
    pinMode(SensorRight, INPUT); //Define Right tracking infrared sensor for
the input interface
    pinMode(SensorLeft, INPUT); //Define left tracking infrared sensor for the
input interface
    pinMode(SensorRight_2, INPUT); //Define right infrared sensor for the input
interface
    pinMode(SensorLeft_2, INPUT); //Define left infrared sensor for the input
interface
}
//=====================================================The basic action of
car=====
//void run(int time)

```

```

void run()
{
    digitalWrite(Right_motor_go,HIGH); // right motor go
    digitalWrite(Right_motor_back,LOW);
    analogWrite(Right_motor_go,200);//PWM ratio 0~255 speed control,
                                //the difference of left and right wheel slightly increase or
decrease
    analogWrite(Right_motor_back,0);
    digitalWrite(Left_motor_go,HIGH); // left motor go
    digitalWrite(Left_motor_back,LOW);
    analogWrite(Left_motor_go,200);//PWM ratio 0~255 speed control,
                                //the difference of left and right wheel slightly increase or
decrease
    analogWrite(Left_motor_back,0);
    //delay(time * 100); //execution time, can be adjusted
}
void brake(int time)
{
    digitalWrite(Right_motor_go,LOW);
    digitalWrite(Right_motor_back,LOW);
    digitalWrite(Left_motor_go,LOW);
    digitalWrite(Left_motor_back,LOW);
    delay(time * 100); //execution time, can be adjusted
}
//void left(int time)
void left() //turn left(left wheel stop,right wheel go)
{
    digitalWrite(Right_motor_go,HIGH); // right motor go
    digitalWrite(Right_motor_back,LOW);
    analogWrite(Right_motor_go,175);
    analogWrite(Right_motor_back,0); //PWM ratio 0~255 speed control
    digitalWrite(Left_motor_go,LOW);
    digitalWrite(Left_motor_back,LOW);
    analogWrite(Left_motor_go,0);
    analogWrite(Left_motor_back,0); //PWM ratio 0~255 speed control
    //delay(time * 100); //execution time, can be adjusted
}

```

```

}
void spin_left(int time)      //left rotation(left wheel back, right wheel go)
{
    digitalWrite(Right_motor_go,HIGH); //right motor go
    digitalWrite(Right_motor_back,LOW);
    analogWrite(Right_motor_go,250);
    analogWrite(Right_motor_back,0);    //PWM ratio 0~255 speed control
    digitalWrite(Left_motor_go,LOW);    //left motor back
    digitalWrite(Left_motor_back,HIGH);
    analogWrite(Left_motor_go,0);
    analogWrite(Left_motor_back,100);    //PWM ratio 0~255 speed control
    delay(time * 100); //execution time, can be adjusted
}
//void right(int time)
void right()      //turn right (right wheel stop,left wheel go)
{
    digitalWrite(Right_motor_go,LOW);
    digitalWrite(Right_motor_back,LOW);
    analogWrite(Right_motor_go,0);
    analogWrite(Right_motor_back,0);    //PWM ratio 0~255 speed control
    digitalWrite(Left_motor_go,HIGH);    //left motor go
    digitalWrite(Left_motor_back,LOW);
    analogWrite(Left_motor_go,175);
    analogWrite(Left_motor_back,0);    //PWM ratio 0~255 speed control
    //delay(time * 100);    //execution time, can be adjusted
}
void spin_right(int time)      //right rotation(right wheel back,left wheel go)
{
    digitalWrite(Right_motor_go,LOW);
    digitalWrite(Right_motor_back,HIGH); //right motor back
    analogWrite(Right_motor_go,0);
    analogWrite(Right_motor_back,100);    //PWM ratio 0~255 speed control
    digitalWrite(Left_motor_go,HIGH);    //left motor go
    digitalWrite(Left_motor_back,LOW);
    analogWrite(Left_motor_go,100);
    analogWrite(Left_motor_back,0);    //PWM ratio 0~255 speed control
}

```

```

    delay(time * 100); //execution time, can be adjusted
}
void back(int time)
{
    digitalWrite(Right_motor_go,LOW); //right motor back
    digitalWrite(Right_motor_back,HIGH);
    analogWrite(Right_motor_go,0);
    analogWrite(Right_motor_back,100);//PWM ratio 0~255 speed control
    digitalWrite(Left_motor_go,LOW);
    digitalWrite(Left_motor_back,HIGH); //left motor back
    analogWrite(Left_motor_go,0);
    analogWrite(Left_motor_back,150);//PWM ratio 0~255 speed control
    delay(time * 100); //execution time, can be adjusted
}
//=====================================================
void keysachn()
{
    int val;
    val=digitalRead(key);//Read the value of the port 7 level to the val
    while(!digitalRead(key))//When the key is not pressed, circulate all the time
    {
        val=digitalRead(key);//This sentence can be omitted and the circulate can
run away.
    }
    while(digitalRead(key))//When the key is pressed
    {
        delay(10);
        val=digitalRead(key);//Read the value of the port 7 level to the val
        if(val==HIGH) //Judge whether the key is pressed again
        {
            digitalWrite(beep,HIGH); //buzzer sound
            while(!digitalRead(key)) //Judge whether the key isreleased
                digitalWrite(beep,LOW); //buzzer no sound
        }
    }
    else
        digitalWrite(beep,LOW); // buzzer no sound
}

```

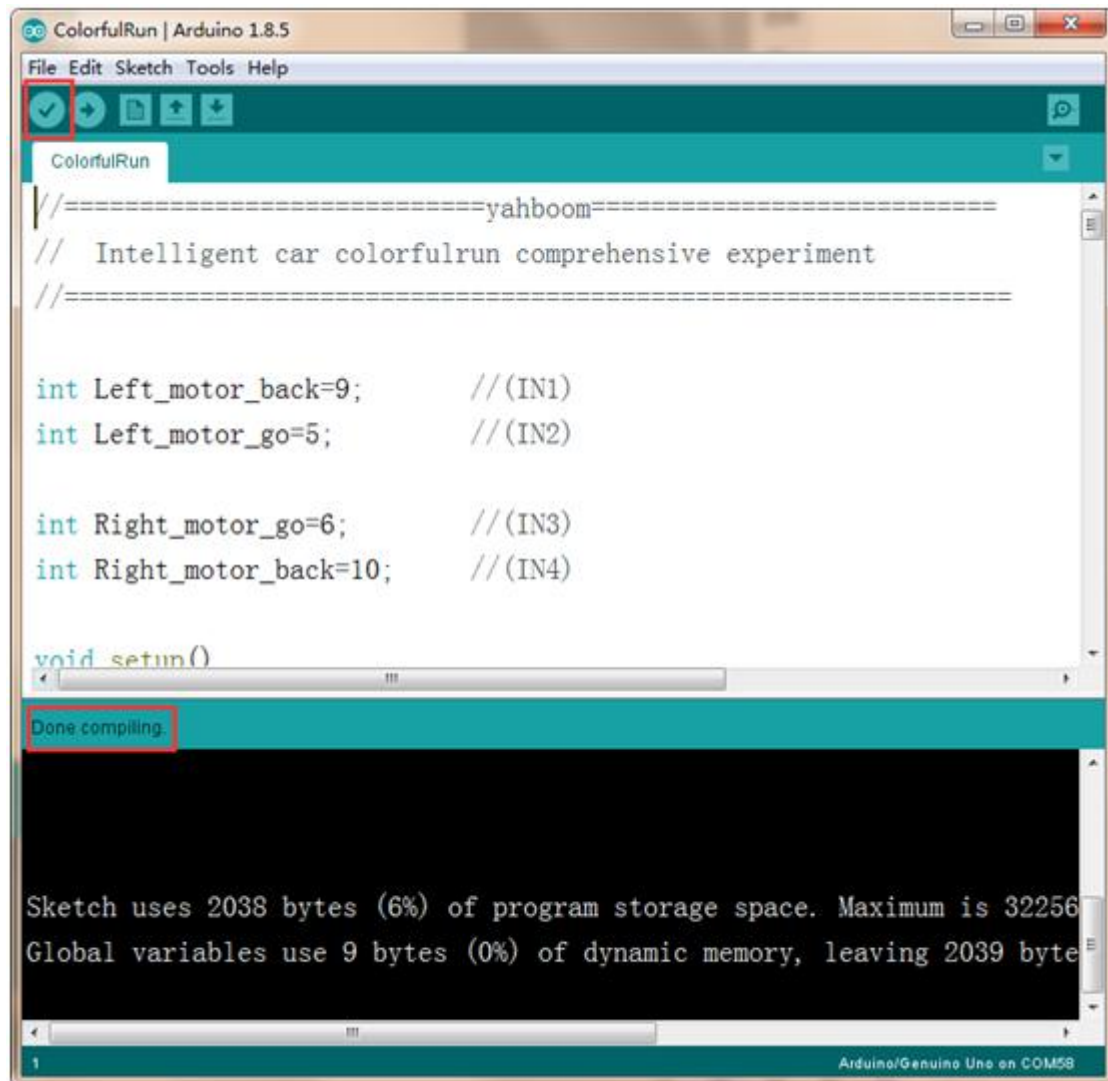
```

    }
}
void loop()
{
    keysacn();    //Call key scan function
    while(1)
    {
        //There is a signal is LOW , no signal is HIGH
        SR_2 = digitalRead(SensorRight_2);
        SL_2 = digitalRead(SensorLeft_2);
        if (SL_2 == HIGH&&SR_2==HIGH)
            run();    //Call run function
        else if (SL_2 == HIGH & SR_2 == LOW)
            //There is an obstacle on the right,return signal,turn left.
            left();
        else if (SR_2 == HIGH & SL_2 == LOW)
            //There is an obstacle on the left,return signal,turn right
            right();
        else //There are obstacles on both sides , back
            back(3);
    }
}

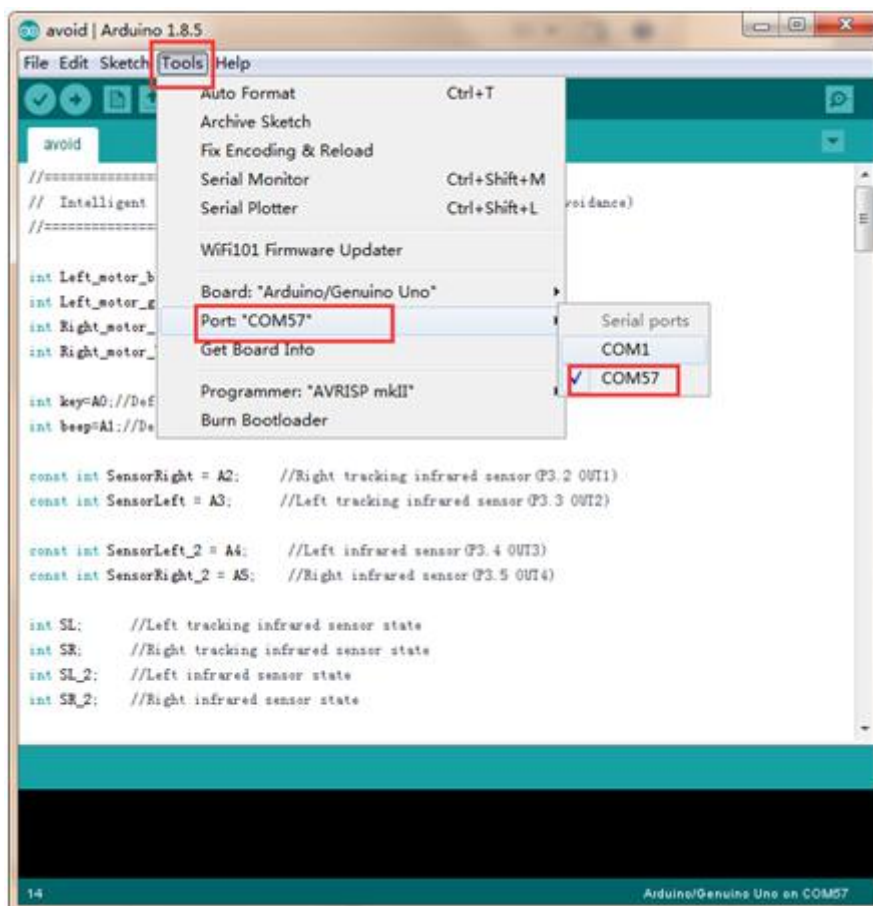
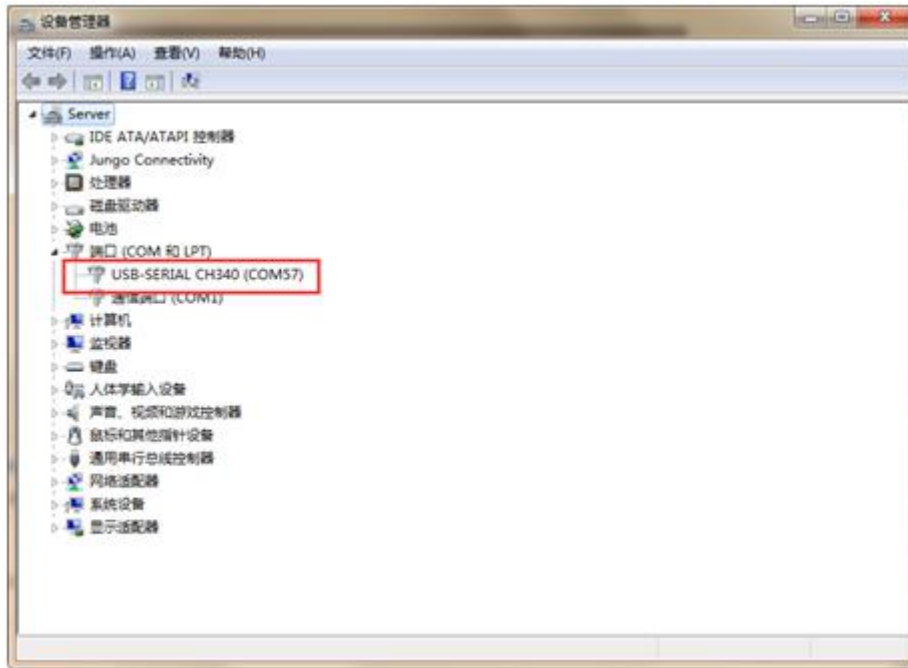
```

Experimental steps:

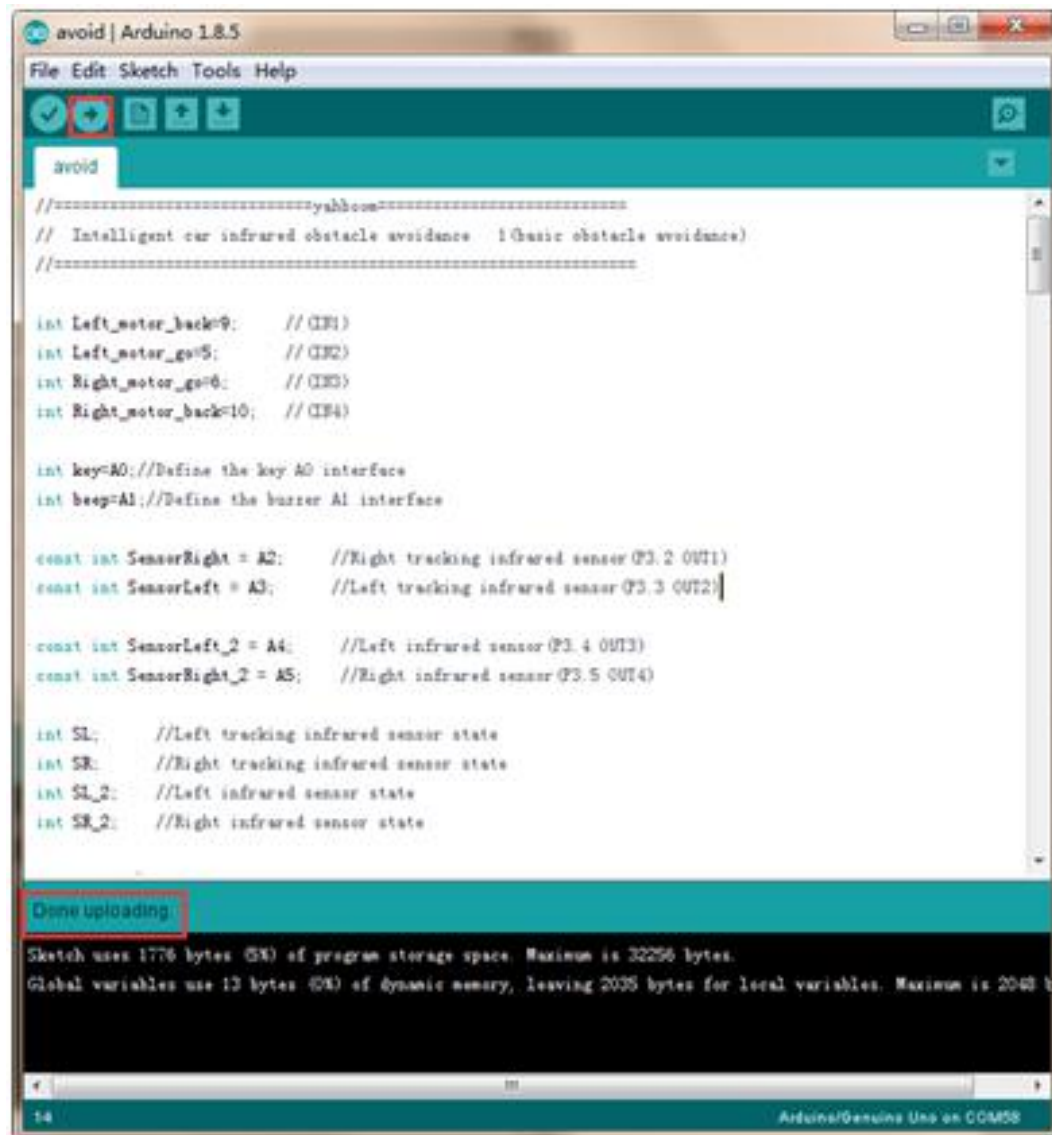
1. We need to open the code of this experiment: **avoid.ino**, click“√” under the menu bar to compile the code, and wait for the word "**Done compiling** " in the lower right corner,as shown in the figure below.



2. In the menu bar of Arduino IDE, we need to select **【Tools】** --- **【Port】** --- selecting the port that the serial number displayed by the device manager just now, as shown in the figure below.

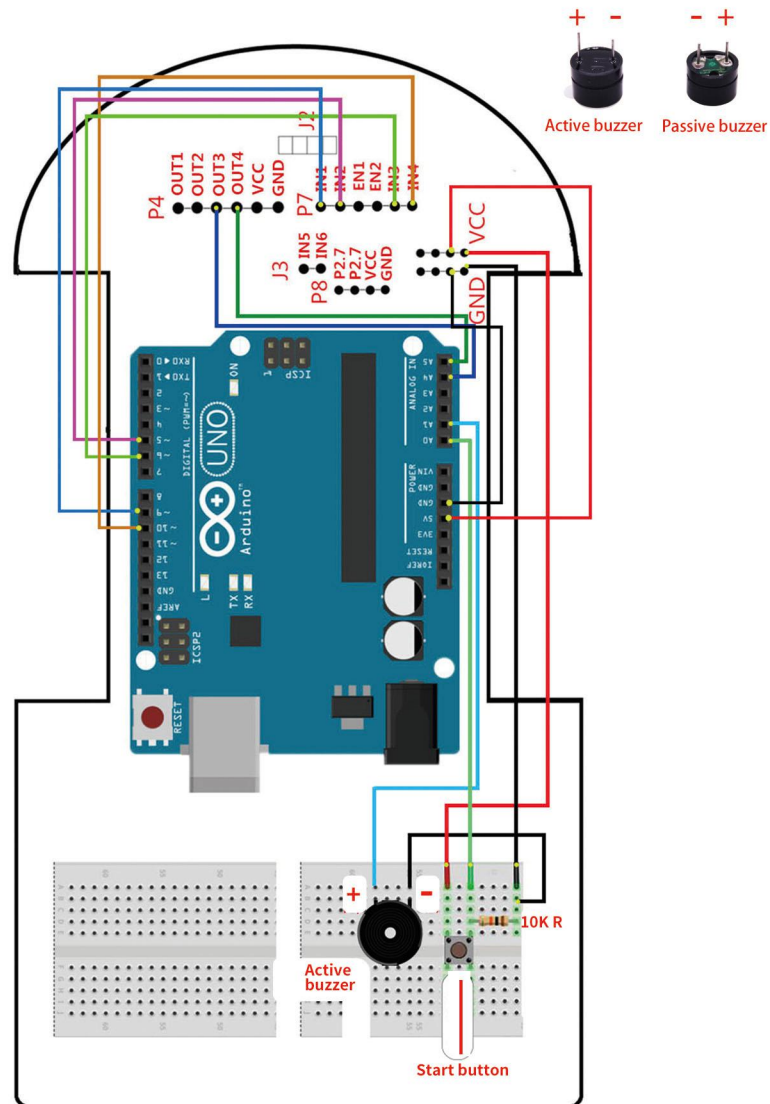


3. After the selection is completed, you need to click “→” under the menu bar to upload the code to the Arduino UNO board. When the word **“Done uploading”** appears in the lower left corner, the code has been successfully uploaded to the Arduino UNO board, as shown in the figure below.



4. Please wire the Smart Car as shown below.

4.3 Infrared obstacle avoidance wiring diagram



According to the wiring diagram, the smart car can realize infrared obstacle avoidance and infrared follow-up functions after uploading the corresponding program. Before the experiment, please refer to (III. Functions of Usage Instructions).

//////////////////////////////////// 10 //////////////////////////////////////

5. Place the debugged smart car on the ground with obstacles and press the start button. The car will avoid obstacles.

