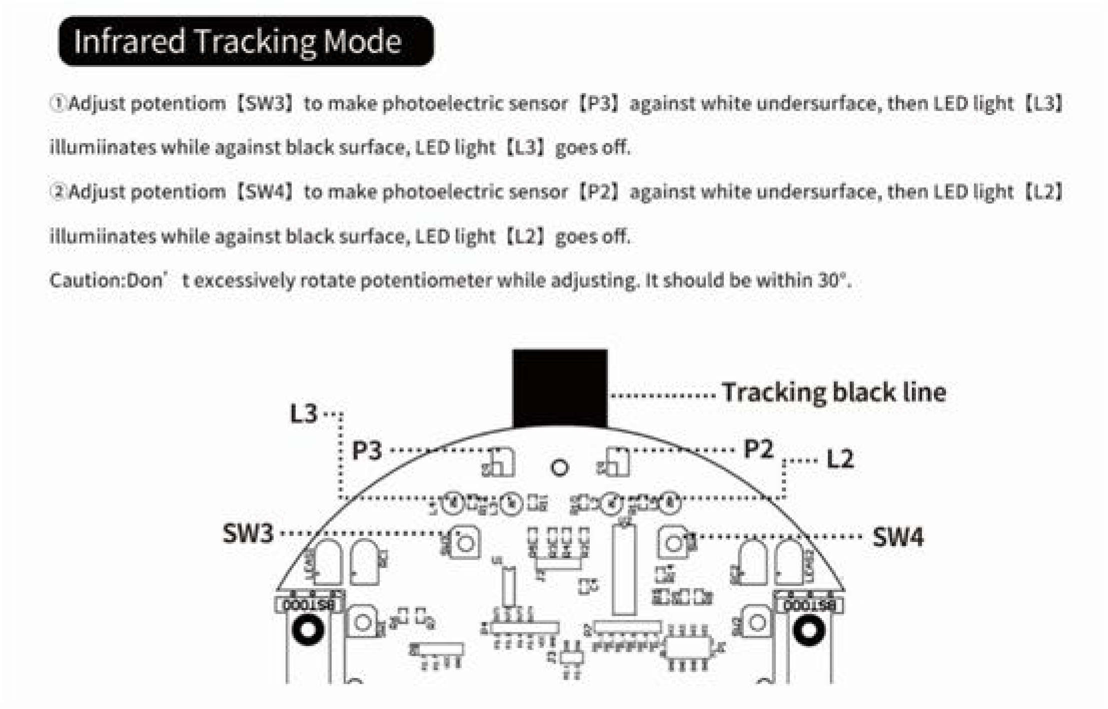
5 Tracking

The purpose of the experiment:

Place the smart car on the patrol track and press the start button. With a short whistle, the car begins to walk along the patrol track.

Precautions:

1. Before conducting the experiment, you need to debug the adjustable resistors sw3 and sw4 in front of the smart car, as shown below.



Infrared Tracking Mode

①Adjust potentiom 【SW3】 to make photoelectric sensor 【P3】 against white undersurface, then LED light 【L3】 illumiinates while against black surface, LED light 【L3】 goes off.

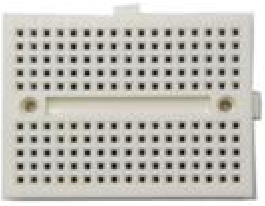
②Adjust potentiom 【SW4】 to make photoelectric sensor 【P2】 against white undersurface, then LED light 【L2】 illumiinates while against black surface, LED light 【L2】 goes off.

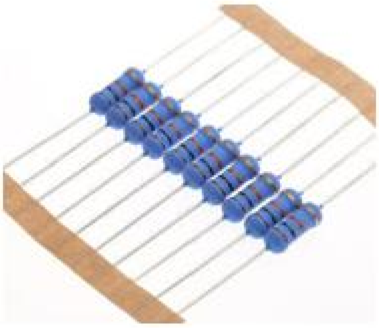
Caution:Don' t excessively rotate potentiometer while adjusting. It should be within 30°.

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 data cable* 1
Active buzzer* 1
DuPont Line* 13
Breadboard* 1
Button * 1
Electrical tape with a width of about 1.6cm* 1
10K resistor * 1





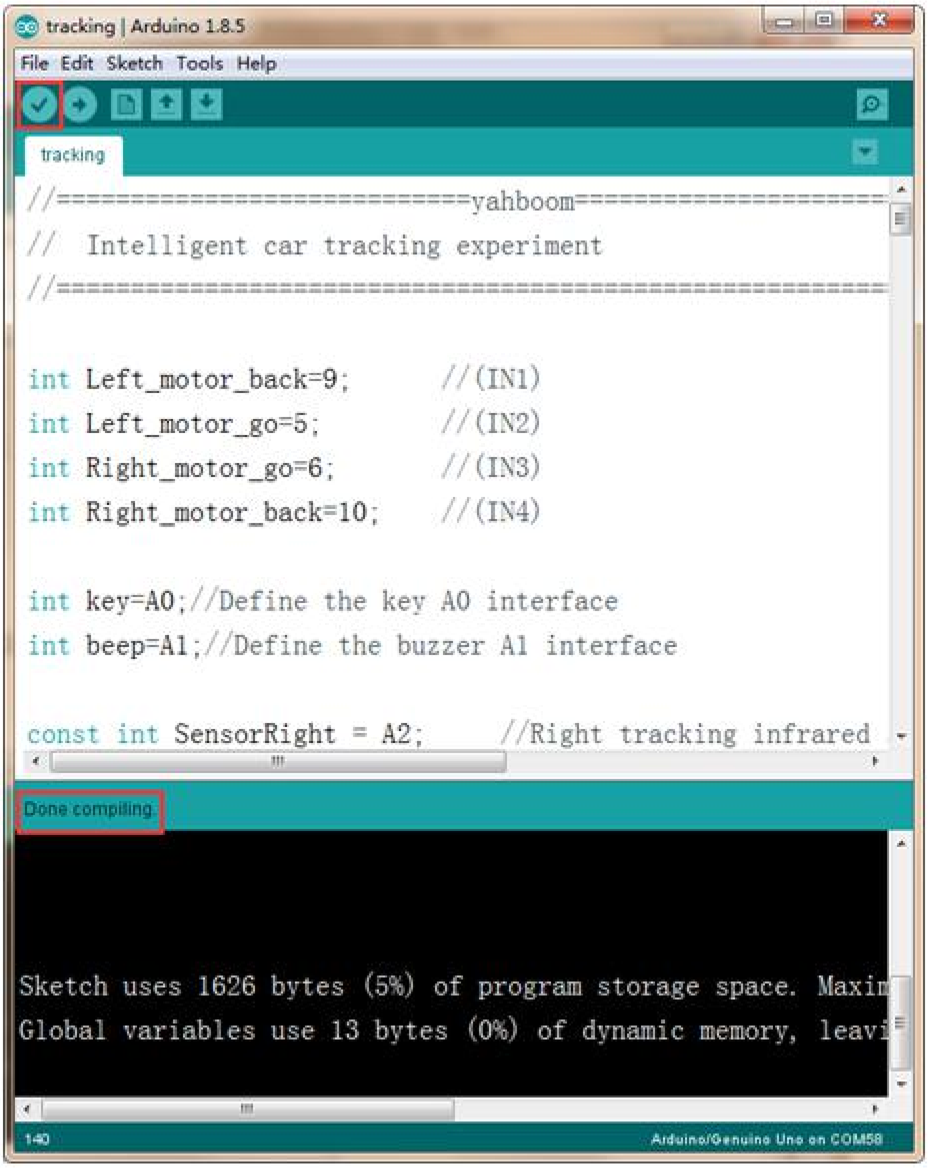
Experimental code analysis:

```
//============================yahboom=======================
====
// Intelligent car tracking experiment
```

```
//============================================================
=====
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)
int SL;     //Left tracking infrared sensor state
int SR;     //Right tracking 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
}
//======================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,150);//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,150);//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)
void brake()
{
  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,150);
  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,200);
  analogWrite(Right_motor_back,0);       //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,200);//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,150);
  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,200);     //PWM ratio 0~255 speed control
  digitalWrite(Left_motor_go,HIGH);      //left motor go
  digitalWrite(Left_motor_back,LOW);
  analogWrite(Left_motor_go,200);
  analogWrite(Left_motor_back,0);       //PWM ratio 0~255 speed control
  delay(time * 100);    //execution time, can be adjusted
}
//void back(int time)
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,150);//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 keysacn()
{
  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 = digitalRead(SensorRight);
  //There is a signal that in the white area the L3 is bright on the car floor;
  // no signal indicates that on the black line and the L3 is extinguishing on the
car floor.

  SL = digitalRead(SensorLeft);
  //There is a signal that in the white area the L2 is bright on the car floor;
  // no signal indicates that on the black line and the L2 is extinguishing on the
car floor.

  if (SL == LOW&&SR==LOW)
    run();    //Call run function
  else if (SL == HIGH & SR == LOW)
  //Left tracking infrared sensor signal is detected,the car deviates from   track,
turn left
    left();
  else if (SR == HIGH & SL == LOW)
  //Right tracking infrared sensor signal is detected,the car deviates
from   track, turn right
    right();
  else //all white, stop
  brake();
  }
}
```

Experimental steps:

1. We need to open the code of this experiment: **tracking.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.

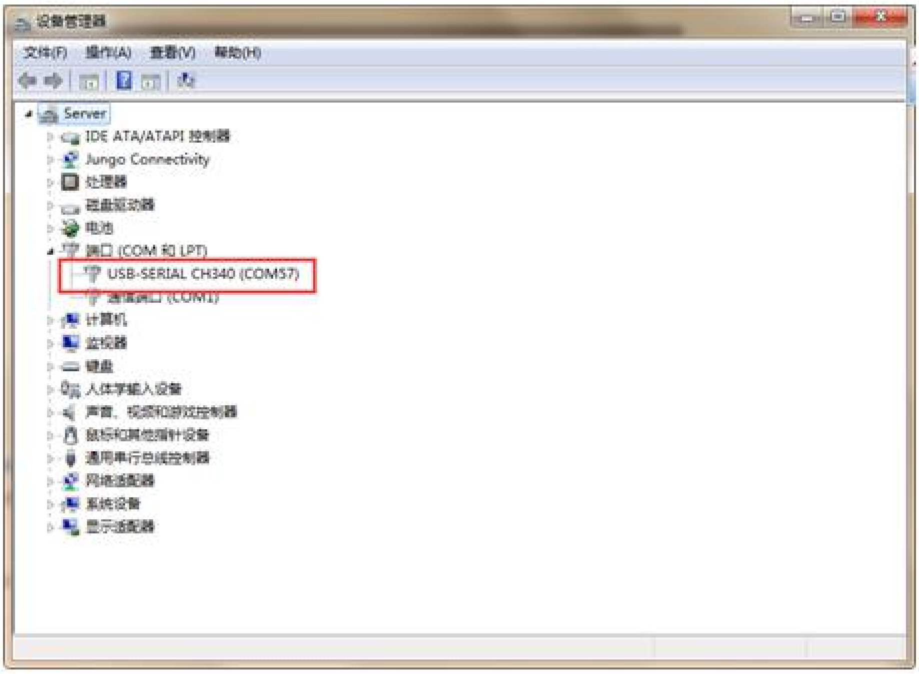


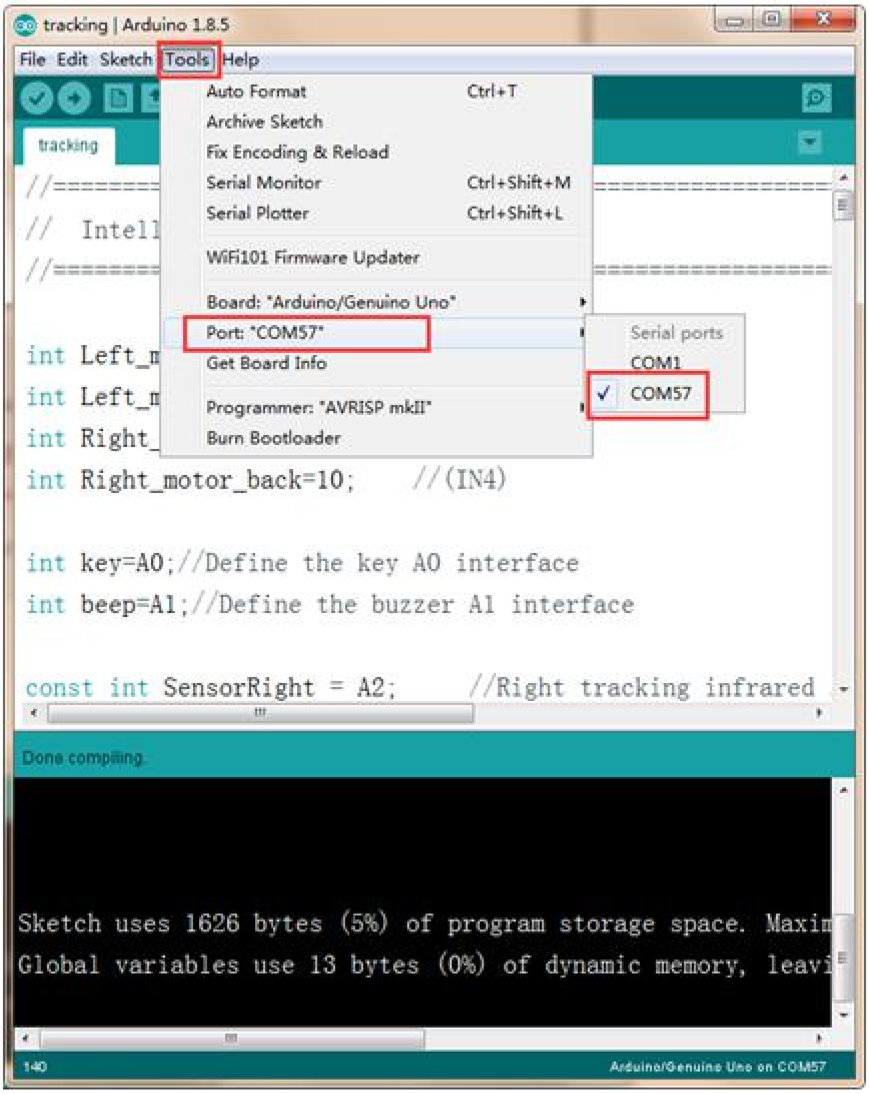
```
//===========================yahboom====================
//  Intelligent car tracking experiment
//=======================================================

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
```

Done compiling.

```
Sketch uses 1626 bytes (5%) of program storage space. Maxim
Global variables use 13 bytes (0%) of dynamic memory, leavi
```

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.



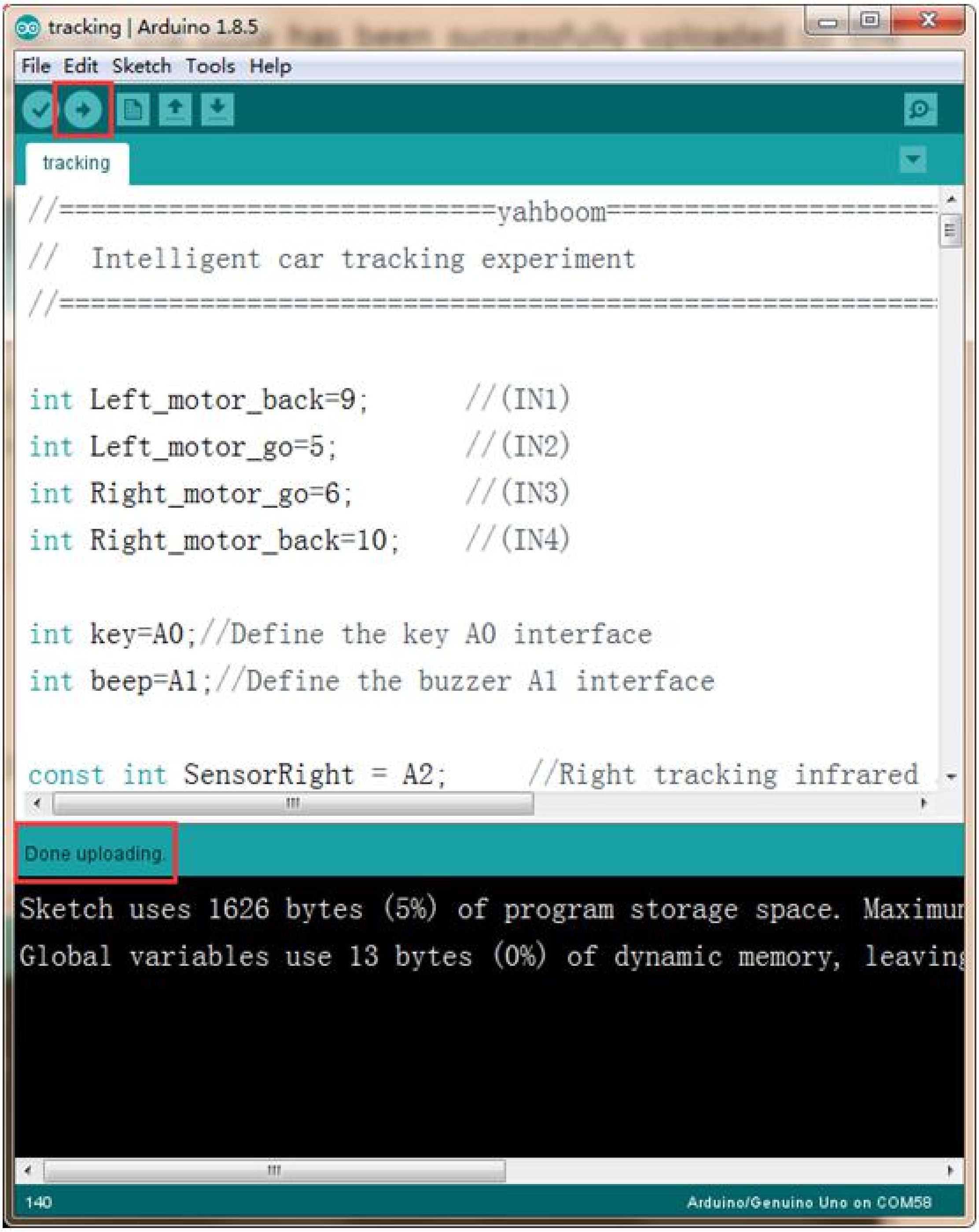


```
//========
//  Intell
//========

int Left_m
int Left_m
int Right_
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
```

Done compiling.

```
Sketch uses 1626 bytes (5%) of program storage space. Maxim
Global variables use 13 bytes (0%) of dynamic memory, leavi
```

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.

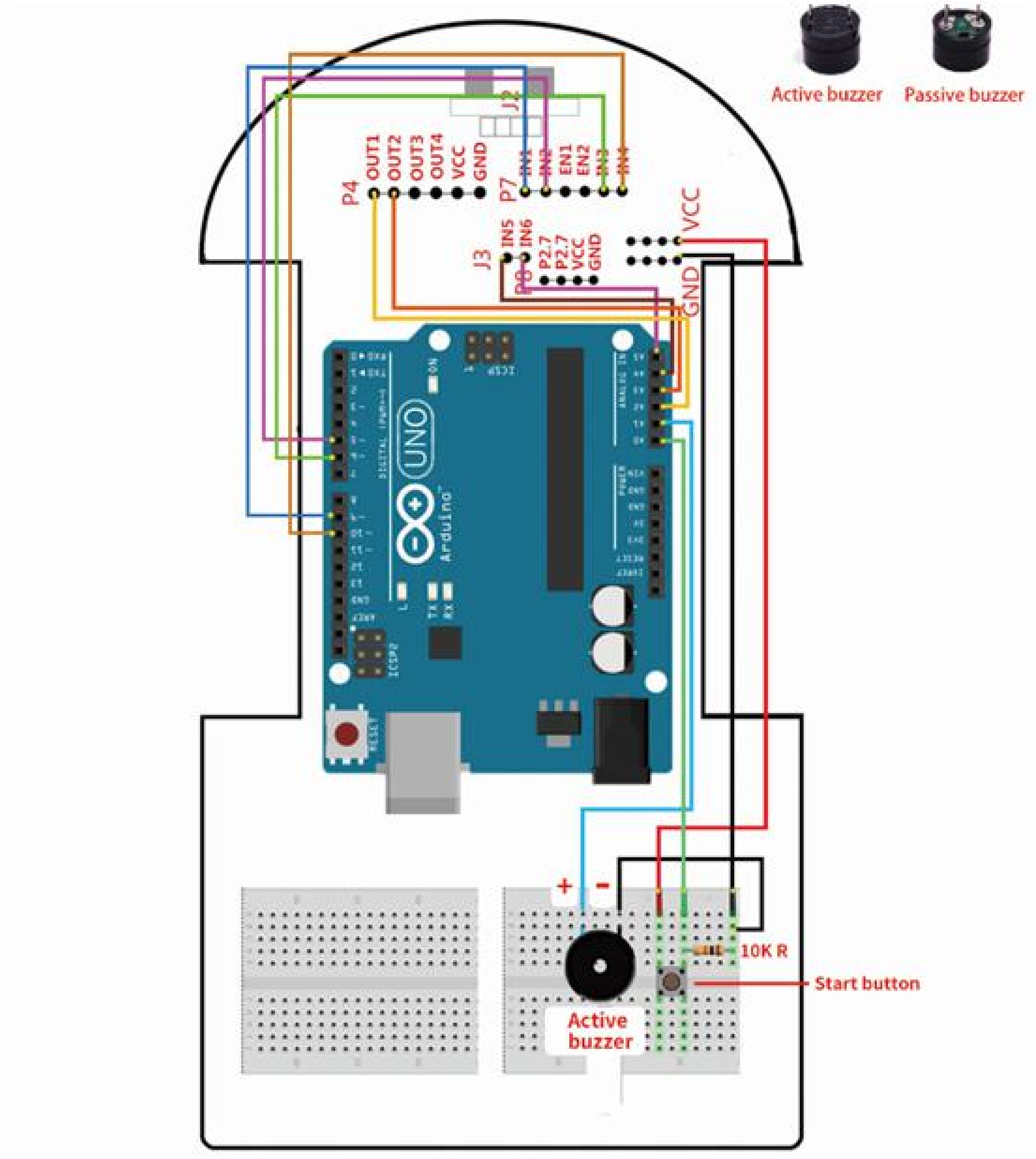


```
//=========================yahboom====================
//  Intelligent car tracking experiment
//=====================================================


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
```

Done uploading.

```
Sketch uses 1626 bytes (5%) of program storage space. Maximur
Global variables use 13 bytes (0%) of dynamic memory, leavin
```

4.Please wire the Smart Car as shown below.



Note: At the J2 slot, insert the ultrasonic sensor as picture.

This experiment is a 2in1 comprehensive experiment. The car can detect the obstacles while tracking. When encountering an obstacle, the car stops waiting in place. After clearing obstacle, the car continues to track.

5.Use a 1.6cm wide black electrical tape to attach the curved track as shown in the figure below on the light ground or tabletop. Place the debugged smart car on the track and press the start button. The car starts to walk along the black line.