12 Infrared remote control experiment

The purpose of the experiment:

After uploading the Remote_contorl.ino program, place the car indoors and pull the curtains to block the outdoor lights. Align the infrared emitter of the infrared remote control with the infrared receiver at the rear of the Smart Car, then press the numeric keypad of the infrared remote control to control the Smart Car to complete the corresponding action.

Precautions:

1. Incorrect connection of the infrared receiving head will cause the receiving head to burn out, so this test must be strictly wired according to the wiring diagram.

2.This experiment requires the use of an infrared remote control. Remove the insulating plastic sheet from the bottom of the remote control before use.

List of components required for the experiment:

Arduino Smart Car* 1
USB cable* 1
DuPont Line* 9
Infrared receiver * 1
Infrared remote control* 1

Experimental code analysis:

```
//============================yahboom=========================
================
//   Intelligent car IRemote contorl
//In the experiment, the received infrared signal is used as the signal for the distribution remote control,
//and the signal value can be printed out with other infrared signals control.
//The speed of the motor can not be adjusted in this experiment.
//The adjustment of the PWM value will affect the signal reception of the infrared
//=============================================================
==================
#include <IRremote.h>
int RECV_PIN = A4;//declare port
IRrecv irrecv(RECV_PIN);
decode_results results;//declare structure
int on = 0;//Marker bit
unsigned long last = millis();
long run_car = 0x00FF18E7;    //key 2
long back_car = 0x00FF4AB5;   //key 8
long left_car = 0x00FF10EF;   //key 4
```

```
long right_car = 0x00FF5AA5; //key 6
long stop_car = 0x00FF38C7;   //key 5
long left_turn = 0x00ff30CF; //key 1
long right_turn = 0x00FF7A85;//key 3
//==============================
int Left_motor_back=9;      //(IN1)
int Left_motor_go=5;        //(IN2)
int Right_motor_go=6;       //(IN3)
int Right_motor_back=10;    //(IN4)
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(13, OUTPUT);   //Define the key interface for the input interface
  Serial.begin(9600); //baud rate 9600
  irrecv.enableIRIn();   // Start the receiver
}
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()
{
  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()         //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,200);
  //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()         //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);    //left motor back
  digitalWrite(Left_motor_back,HIGH);
  //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()        //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,200);
  //analogWrite(Left_motor_back,0);//PWM ratio 0~255 speed control
  //delay(time * 100);       //execution time, can be adjusted
}
void spin_right()        //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()
{
  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);     //left motor back
  digitalWrite(Left_motor_back,HIGH);
  //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 dump(decode_results *results)
{
  int count = results->rawlen;
  if (results->decode_type == UNKNOWN)
```

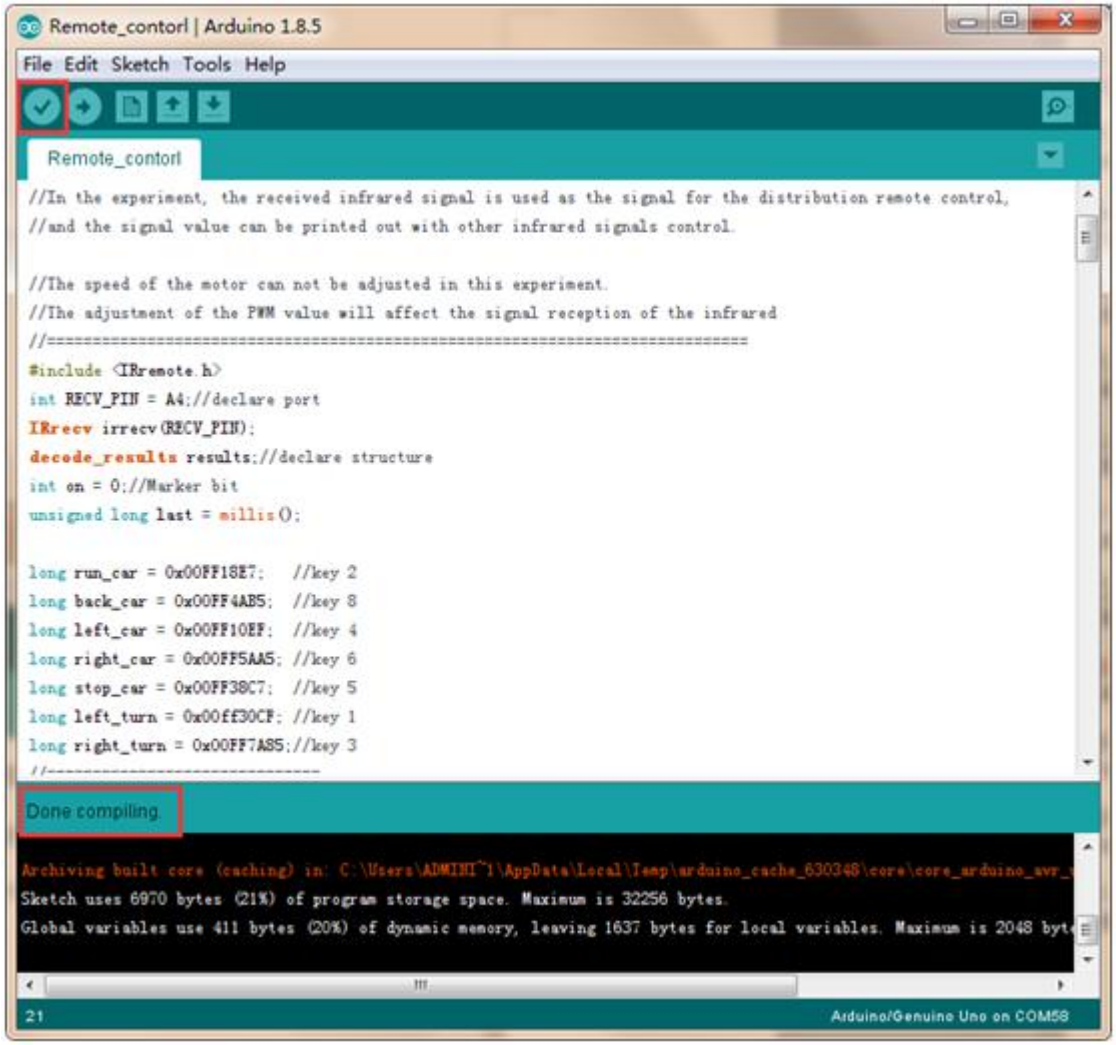
```
  {
    //Serial.println("Could not decode message");
    brake();
  }
//Serial port printing, debugging can be opened,
//the actual operation will affect the speed of reaction, it is recommended to shield
/*
  else
  {

    if (results->decode_type == NEC)
    {
      Serial.print("Decoded NEC: ");
    }
    else if (results->decode_type == SONY)
    {
      Serial.print("Decoded SONY: ");
    }
    else if (results->decode_type == RC5)
    {
      Serial.print("Decoded RC5: ");
    }
    else if (results->decode_type == RC6)
    {
      Serial.print("Decoded RC6: ");
    }
    Serial.print(results->value, HEX);
    Serial.print(" (");
    Serial.print(results->bits, DEC);
    Serial.println(" bits)");
  }
  Serial.print("Raw (");
  Serial.print(count, DEC);
  Serial.print("): ");
  for (int i = 0; i < count; i++)
```

```
  {
    if ((i % 2) == 1)
    {
      Serial.print(results->rawbuf[i]*USECPERTICK, DEC);
    }
    else
    {
      Serial.print(-(int)results->rawbuf[i]*USECPERTICK, DEC);
    }
    Serial.print(" ");
  }
  Serial.println("");
*/
}
void loop()
{
  if (irrecv.decode(&results)) //Call library function: decode
  {
    // If it's been at least 1/4 second since the last
    // IR received, toggle the relay
    if (millis() - last > 250) //Determine the received signal
    {
      on = !on;
      digitalWrite(13, on ? HIGH : LOW);
      //The signal is received on the board, led_twinkle
      dump(&results);//Decoded infrared signal
    }
    if (results.value == run_car )//key 2
      run();
    if (results.value == back_car )//key 8
      back();
    if (results.value == left_car )//key 4
      left();//turn left
    if (results.value == right_car )//key 6
      right();//turn right
    if (results.value == stop_car )//key 5
```

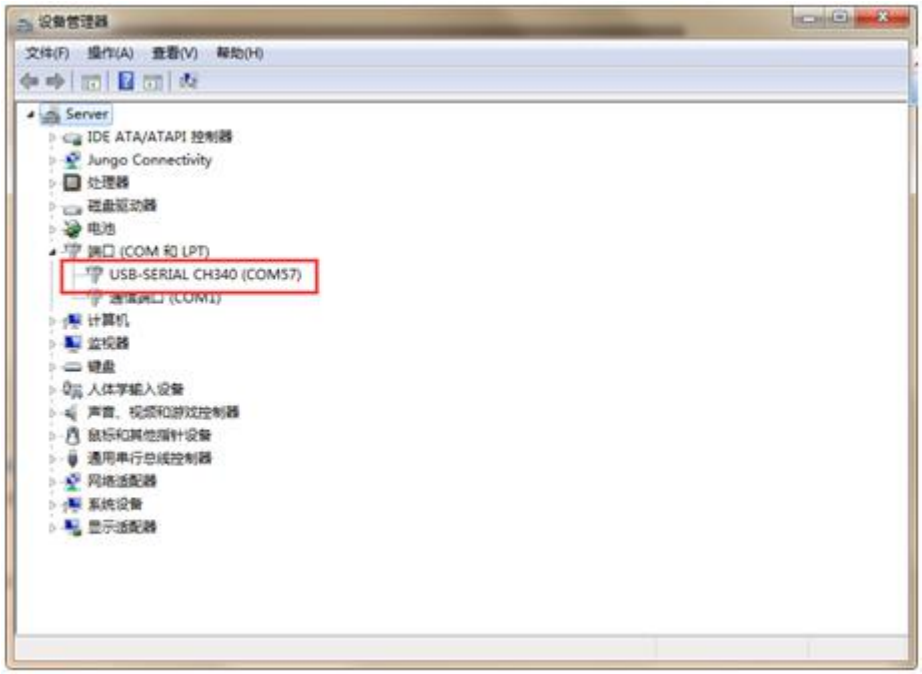
```
      brake();
    if (results.value == left_turn )//key 1
      spin_left();//left retation
    if (results.value == right_turn )//key 3
      spin_right();//right retation
    last = millis();
    irrecv.resume(); // Receive the next value
  }
}
```

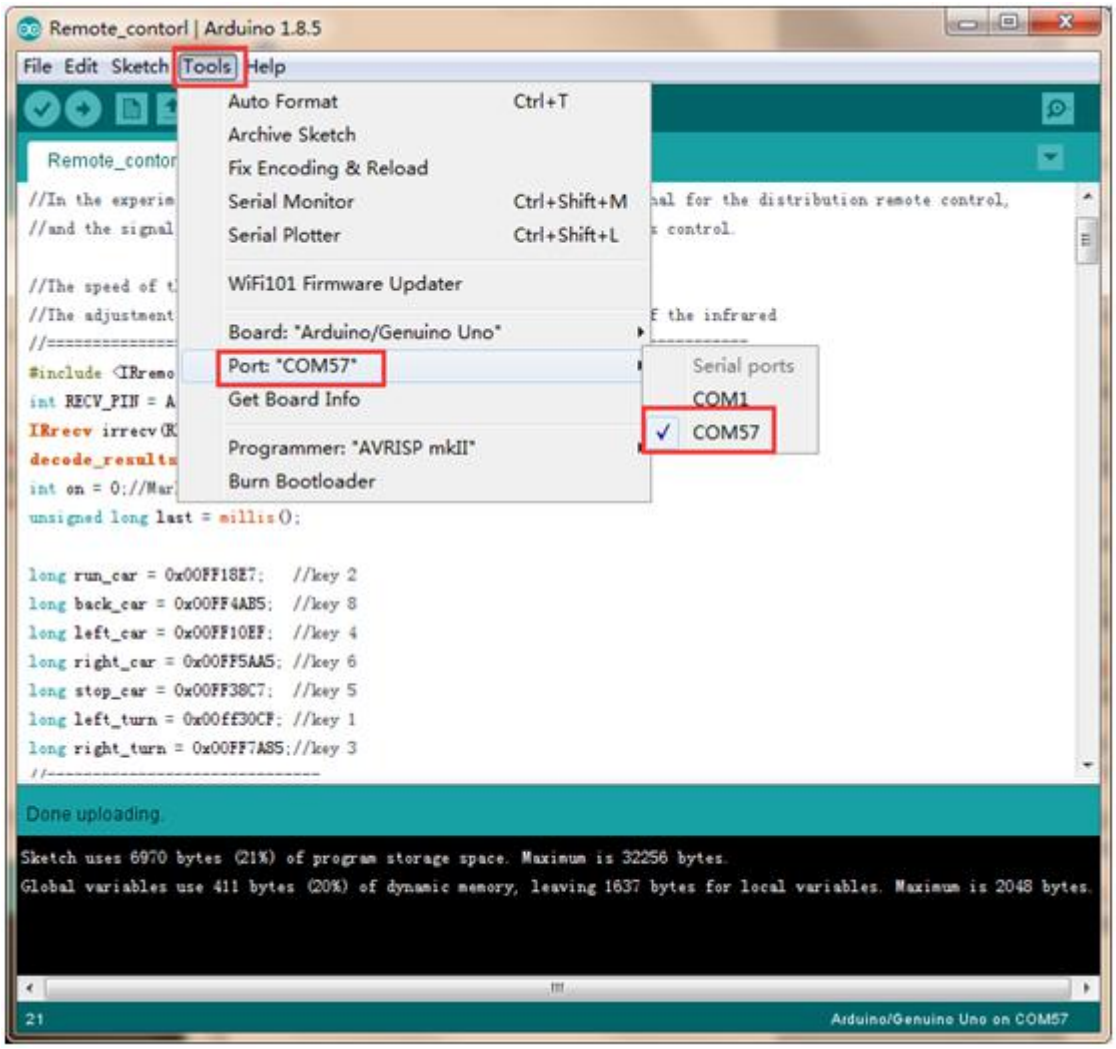
Experimental steps:

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

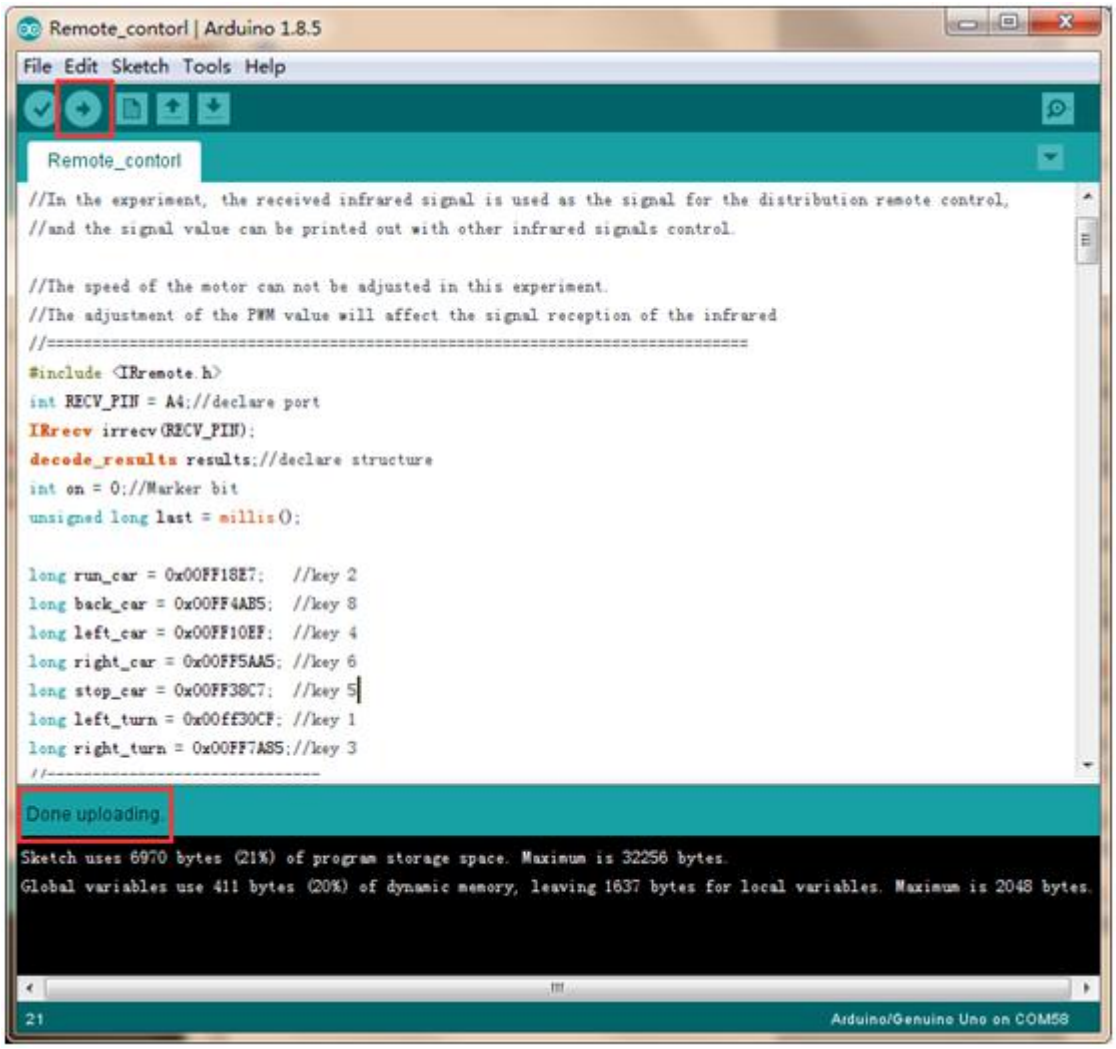


2. In the menu bar of Arduino IDE, we need to select 【Tools】---【Port】---selecting the port that the serial number displayed by the device manager just now, as shown in the figure below.



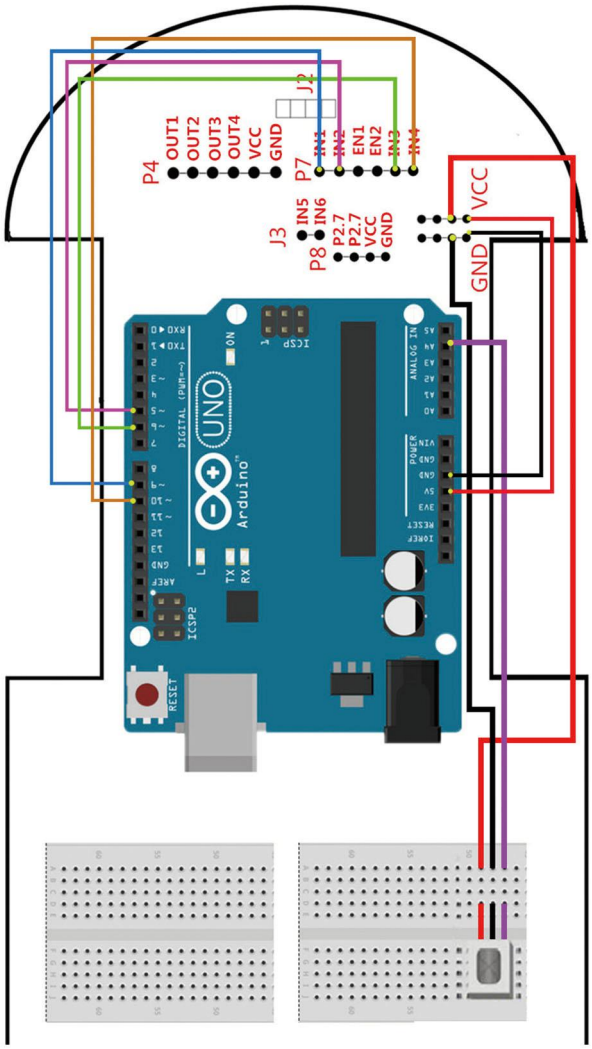


3. After the selection is completed, you need to click "→"under the menu bar to upload the code to the Arduino UNO board. When the word "**Done uploading**" appears in the lower left corner, the code has been successfully uploaded to the Arduino UNO board, as shown in the figure below.



4.Please wire the Smart Car as shown below.

4.6 Remote control wiring diagram



This experiment requires the use of an infrared remote control. Before use, please remove the insulated plastic sheet at the bottom of the remote control. The numbers 2,8,4,6 on the remote control correspond to advance, return back, turn left and ture right; 1,3 corresponds to left and right rotation; 5 is the stop button.

5.This experiment requires the use of an infrared remote control. Before use, please remove the insulated plastic sheet at thebottom of the remote control. The numbers 2,8,4,6 on the remote control correspond to advance, return back, turn left andture right; 1,3 corresponds to left and right rotation; 5 is the stop button.

6.The following is the user code value corresponding to the infrared remote control.

Corresponding user code value	The program controls the action of the BatCar	Remote control button
0x00FF9867	No control action	—

0x00FFB04F	No control action	C
0x00ff30CF	Left rotation	1
0x00FF18E7	Forward	2
0x00FF7A85	Right rotation	3
0x00FF10EF	Turn left	4
0x00FF38C7	Brake	5
0x00FF5AA5	Turn right	6
0x00FF42BD	No control action	7
0x00FF4AB5	Backward	8
0x00FF52AD	No control action	9