

Lesson 15 How to use the IR module to Control the Car





In the last lesson we know how to use IR module to pass data with Adeept Robot Control Board. In this course we learn how to use infrared remote control to control the car.

15.1 Components used in this course

Please assemble UnoCar-B according to the assembly tutorial and connect the circuit correctly.

15.2 Introduction of IR Control function

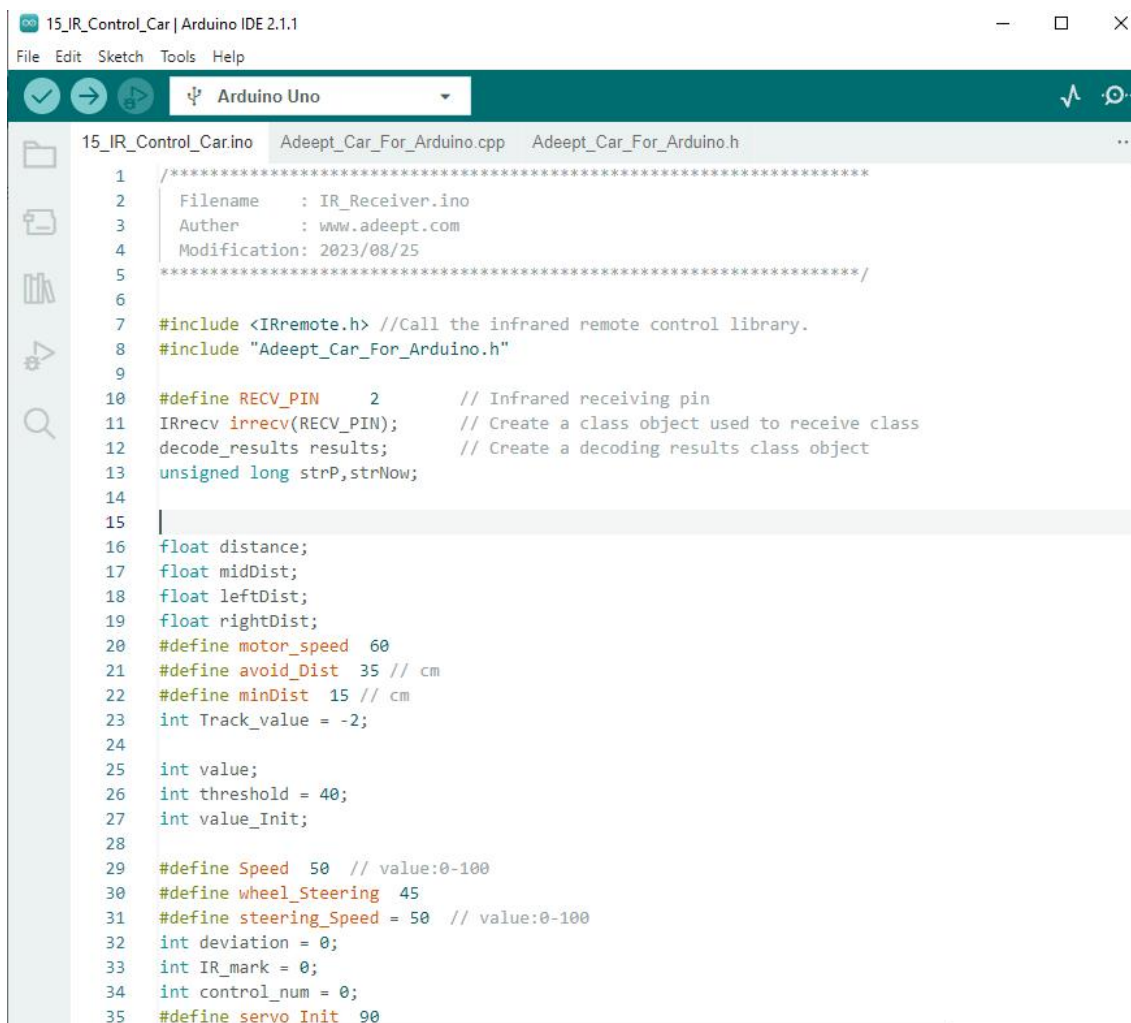
The corresponding relationship between the buttons of the infrared remote control and the functions of the car is as follows:

Button	Function	Button	Function
	Forward	5	Buzzer beep
	Backward	6	Turn off LED
	Left	7	Left Backward
	Right	8	Matrix screen lights up
OK	Stop Function	9	Right Backward
1	Servo 2 turn left	*	Avoid Obstacles Function
2	Servo 2 turn to 90	0	Matrix screen turns off
3	Servo 2 turn right	#	Light Tracking Function
4	Change the color of LED		



15.3 How to use the infrared control function

1. Connect your computer and Adeept Robot Control Board with a USB cable.
2. Open “15_IR_Control_car” folder in “[Adeept_UnoCar-B/Code](#)”, double-click “15_IR_Control_car.ino”.



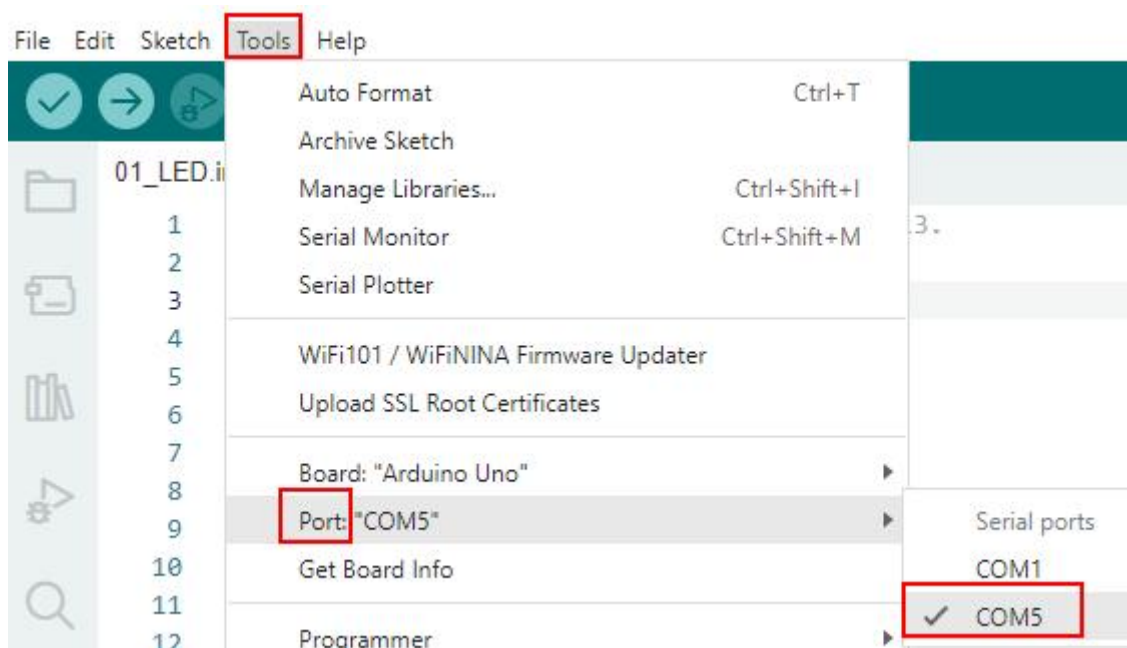
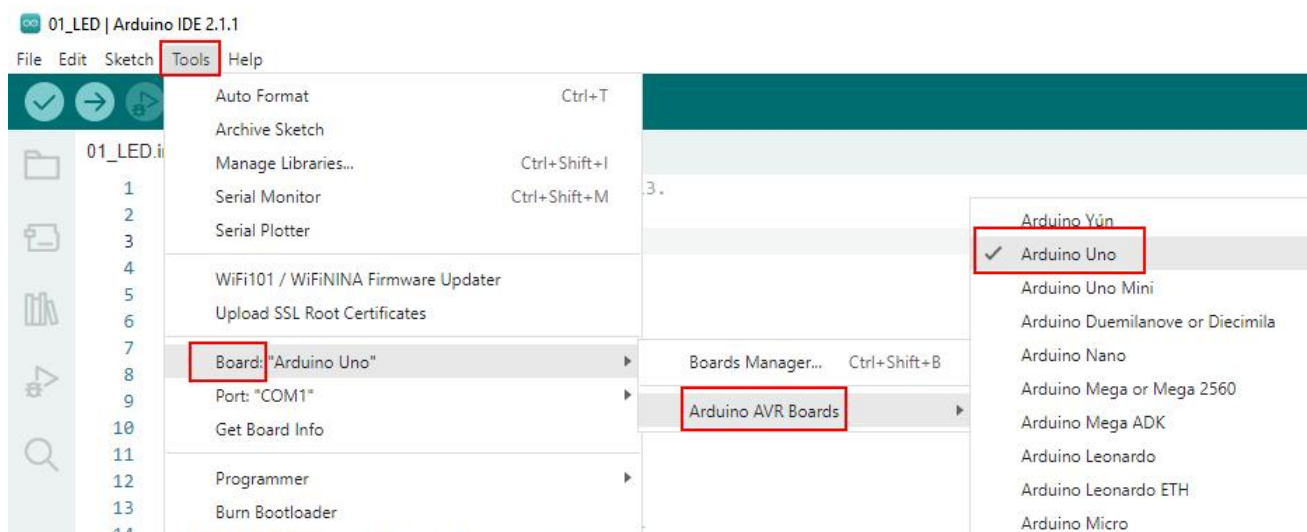
```
1  /*****
2  Filename   : IR_Receiver.ino
3  Author    : www.adeept.com
4  Modification: 2023/08/25
5  *****/
6
7  #include <IRremote.h> //Call the infrared remote control library.
8  #include "Adeept_Car_For_Arduino.h"
9
10 #define RECV_PIN 2 // Infrared receiving pin
11 IRrecv irrecv(RECV_PIN); // Create a class object used to receive class
12 decode_results results; // Create a decoding results class object
13 unsigned long strP, strNow;
14
15
16 float distance;
17 float midDist;
18 float leftDist;
19 float rightDist;
20 #define motor_speed 60
21 #define avoid_Dist 35 // cm
22 #define minDist 15 // cm
23 int Track_value = -2;
24
25 int value;
26 int threshold = 40;
27 int value_Init;
28
29 #define Speed 50 // value:0-100
30 #define wheel_Steering 45
31 #define steering_Speed = 50 // value:0-100
32 int deviation = 0;
33 int IR_mark = 0;
34 int control_num = 0;
35 #define servo_Init 90
```


3. Select development board and serial port.

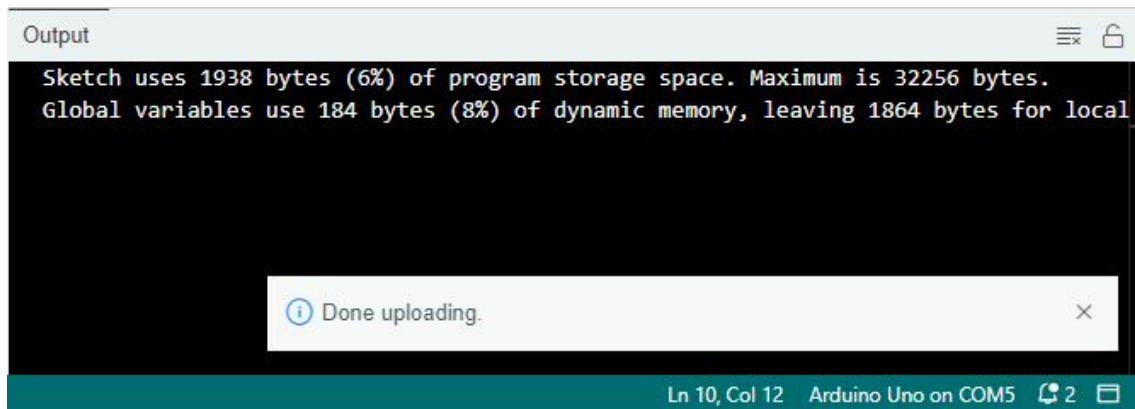
Board: **Tools**---->**Board**---->**Arduino AVR Boards**---->**Arduino Uno**

Port: **Tools** ---->**Port**---->**COMx**

Note: The port number will be different in different computers.

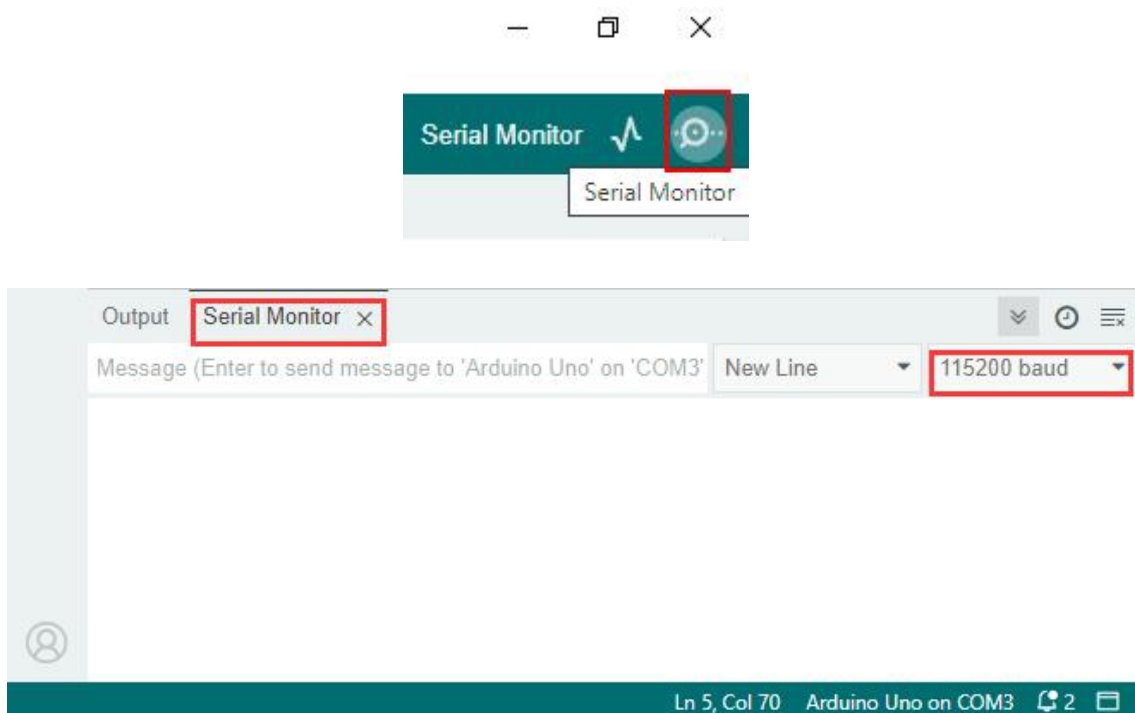


4. After opening, click  to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.







After assembling the car, please use the 18650 battery to provide power when uploading the program, otherwise the program may not be uploaded successfully due to excessive load.

5. click Serial Monitor, Set the baud rate as 115200.



6. Use the IR remote to aim at the IR receiver on the expansion board. Press different buttons on the infrared remote control, you can see that the car has different behaviors.

Button	Function	Button	Function
	Forward	4	Change the color of LED
	Backward	5	Buzzer beep
	Left	6	Turn off LED
	Right	7	Left Backward
OK	Stop Function (click multiple times)	8	Matrix screen lights up
		9	Right Backward
1	Servo 2 turn left	*	Avoid Obstacles Function
2	Servo 2 turn to 90	0	Matrix screen turns off
3	Servo 2 turn right	#	Light Tracking Function
Some Functions may require multiple clicks to stop.			

Please familiarize yourself with the button functions of the infrared remote control first. Please see the list above for the button functions.

It is recommended to raise the vehicle body during testing so that the wheels are suspended in the air.

15.4 Code

[Adeept_Car_For_Arduino.cpp](#) and [Adeept_Car_For_Arduino.h](#) are the library files of the car. These two files have modularized the code in the previous course, so that the program can call the code of each module.

The [15_IR_Control_Car.ino](#) program is the main program, which realizes the functions required by the car.

Note: Since the Adeept Robot Control Board uses the same chip as the Arduino Uno, the [program storage space](#) of the chip is 32kb. When there are too many modified program codes, it may fail to upload to the Arduino board. Please adjust the code content appropriately.

The following is a display of the [15_IR_Control_car.ino](#) program content, which may be subject to change. Please refer to the actual code provided.

```
1. #include <IRremote.h> //Call the infrared remote control library.
2. #include "Adeept_Car_For_Arduino.h"
3.
4. #define RECV_PIN    2          // Infrared receiving pin
5. IRrecv irrecv(RECV_PIN);      // Create a class object used to receive class
6. decode_results results;       // Create a decoding results class object
7. unsigned long strP,strNow;
8.
9. float distance;
10. float midDist;
11. float leftDist;
12. float rightDist;
13. #define motor_speed  60
14. #define avoid_Dist  35 // cm
15. #define minDist  15 // cm
16. int Track_value = -2;
17.
18. int value;
19. int threshold = 40;
20. int value_Init;
21.
22. #define Speed  50 // value:0-100
23. #define wheel_Steering  45
24. #define steering_Speed = 50 // value:0-100
25. int deviation = 0;
26. int IR_mark = 0;
27. int control_num = 0;
28. #define servo_Init  90
29. int servo_Angle2 = servo_Init;
30. int ws2812_flag= 0;
```

```
31. int Function_flag = 0;
32.
33. int time1;
34. int time2;
35.
36. byte color_value[][3] = {{250,0,0},{250,160,0},{250,250,0},{0,250,0},{0,250,250},{0,0,250},
    {130,0,250}};
37. void setup()
38. {
39.     Serial.begin(115200);        // Initialize the serial port and set the baud rate to 115200
40.
41.     RGB_Setup();                //RGB LED initialization
42.     RGB_brightness(2);          // value 0-10
43.     All_RGB(250,0,0); // Set RGB LED color value.
44.     Servo_Setup();              //Servo initialization
45.     PCA9685_Servo_Setup();      //PCA9685 Servo initialization
46.     Motor_Setup();              //Motor initialization
47.     AllMotorStop();
48.     Buzzer_Setup();             //Buzzer initialization
49.     WS2812_Setup();             //WS2812 LED initialization
50.     WS2812_Brightness(5);       // value 0-10
51.     Ultrasonic_Setup();         //Ultrasonic initialization
52.     Photosensitive_Setup();      //Light line initialization
53.     Tracking_Setup();           //Tracking Line initialization
54.     OLED_Setup();              //OLED initialization
55.     Matrix_Setup();
56.
57.     irrecv.enableIRIn();        // Start the receiver
58.     Serial.println("UnoCar-B IR Control Start!");
59.     // Serial.println(RECV_PIN); //print the infrared receiving pin
60.
61.     // Buzzer_Alert(1,1);
62.     WS2812ColorAll(255, 255,0); // Green
63.     Servo_Angle(1, 90);
64.     Servo_Angle(2, 90);
65.     PCA9685_Servo_Angle(6, 0, 90);
66.     PCA9685_Servo_Angle(7, 0, 90);
67.     Buzzer_Silence();
68.     OLED_clear();
69.     delay(1000);
70.     All_RGB(0,0,0); // Set RGB LED color value.
```

```
71. WS2812ColorAll(0,0,0);
72. }
73.
74. void loop()
75. {
76.     // if (irrecv.decode(&results)) {           // Waiting for decoding
77.     if (irrecv.decode(&results)){
78.         unsigned long value = results.value;
79.
80.         value = switch_irr(value);
81.
82.         Serial.println(value);
83.         // Serial.println(value, HEX);           // Print out the decoded results
84.
85.         irrecv.resume();                         // Release the IRremote. Receive the next value
86.         time1 = millis();
87.         control(value);
88.         time2 = millis() - time1;
89.         Serial.print("time:");
90.         Serial.println(time2);
91.         // irrecv.resume();                       // Release the IRremote. Receive the next value
92.     }
93.     // delay(100);
94. }
95.
96. else{
97.     delay(60); //delayed judgment.
98.     if (irrecv.decode(&results)==false){
99.         Servo_Angle(1, servo_Init + deviation);
100.         Motor(1, 1, 0);
101.         Motor(2, 1, 0);
102.     }
103. }
104. delay(100);
105. }
106.
107.
108.
109. void control(unsigned long value){
110.     switch (value) {
111.         // Move Control
```



```
112.     case 12: // Signal when UP is pressed, action: forward
113.         Servo_Angle(1, servo_Init + deviation);
114.         Motor(1, 1, motor_speed); //Motor1 forward
115.         Motor(2, 1, motor_speed); //Motor2 forward
116.         control_num = 12;
117.         break;
118.
119.     case 13: // Down,
120.         // Servo_1_Angle(servo_Init);
121.         Servo_Angle(1, servo_Init + deviation);
122.         Motor(1, -1, motor_speed); //Motor1 backward
123.         Motor(2, -1, motor_speed); //Motor2 backward
124.         control_num = 13;
125.         break;
126.
127.     case 14: // left,
128.         // Servo_1_Angle(servo_Init + wheel_Steering + deviation);
129.         Servo_Angle(1, servo_Init + deviation + wheel_Steering); // left
130.         Motor(1, 1, motor_speed);
131.         Motor(2, 1, motor_speed);
132.         control_num = 14;
133.         break;
134.
135.     case 15: // right
136.         // Servo_1_Angle(servo_Init - wheel_Steering + deviation);
137.         Servo_Angle(1, servo_Init + deviation - wheel_Steering); // right
138.         Motor(1, 1, motor_speed);
139.         Motor(2, 1, motor_speed);
140.         control_num = 15;
141.         break;
142.
143.     case 7:
144.         // Servo_1_Angle(servo_Init + wheel_Steering + deviation);
145.         Servo_Angle(1, servo_Init + deviation + wheel_Steering); // left
146.         Motor(1, -1, motor_speed);
147.         Motor(2, -1, motor_speed);
148.         control_num = 7;
149.         break;
150.
151.     case 9:
152.         Servo_Angle(1, servo_Init + deviation - wheel_Steering); // right
153.         Motor(1, -1, motor_speed);
```

```
154.     Motor(2, -1, motor_speed);
155.     control_num = 8;
156.     break;
157.
158.     // servo 2 control.
159.     case 1:
160.         servo_Angle2 = servo_Angle2 + 5;
161.         if (servo_Angle2 > 180){
162.             servo_Angle2 = 180;
163.         }
164.         Servo_Angle(2, servo_Angle2);
165.         // Serial.println(servo_Angle2);
166.         break;
167.
168.     case 2:
169.         Servo_Angle(2, servo_Init);
170.         servo_Angle2 = servo_Init;
171.         break;
172.
173.     case 3:
174.         servo_Angle2 = servo_Angle2 - 5;
175.         if (servo_Angle2 < 0){
176.             servo_Angle2 = 0;
177.         }
178.         Servo_Angle(2, servo_Angle2);
179.         // Serial.println(servo_Angle);
180.         break;
181.
182.     // LED
183.     case 4:
184.         // for (int i = 0; i < LEDS_COUNT; i++){
185.             WS2812ColorAll(color_value[ws2812_flag][0], color_value[ws2812_flag][1], color_value[ws2812_flag][2]);
186.             All_RGB(color_value[ws2812_flag][0], color_value[ws2812_flag][1], color_value[ws2812_flag][2]);
187.
188.             ws2812_flag++;
189.             // LED_status = 1;
190.             if(ws2812_flag > 6){
191.                 ws2812_flag = 0;
192.             }
193.             break;
```

```
194.
195.     case 6:
196.         WS2812ColorAll(0,0,0);
197.         All_RGB(0,0,0);
198.         break;
199.
200.     case 5:
201.         Buzzer_Alert(3, 1);
202.         break;
203.
204.     case 8:
205.         Matrix_Smile();
206.         break;
207.
208.     case 0:
209.         Matrix_Clear();
210.         break;
211.
212.     // Function.
213.     case 10:
214.         Avoid_Obstacles(); // Avoid Obstacles function
215.         break;
216.
217.     case 11:
218.         Light_Tracking(); // Light Tracking function
219.         break;
220.
221.     case 16: // OK. Stop function.
222.         Servo_Angle(1, servo_Init + deviation);
223.         Motor(1, 1, 0);
224.         Motor(2, 1, 0);
225.         control_num = -1;
226.         break;
227.
228.     default:
229.         break;
230. }
231. }
232.
233. int StopFunction(){
234.     if (irrecv.decode(&results)) { // Waiting for decoding
235.         unsigned long value = results.value;
```

```
236.   value = switch_irr(value);
237.   // Serial.println(value);
238.   if (value == 16){ // OK, Stop function.
239.       Function_flag = 1;}
240.   else{
241.       Function_flag = 0;
242.   }
243.   irrecv.resume();           // Release the IRremote. Receive the next value

244. }
245. // delay(100);
246. }
247.
248. void Avoid_Obstacles(){
249.   while (1){
250.       StopFunction();
251.       if (Function_flag == 1){ // Press OK, stop function.
252.           break;
253.       }
254.
255.       Servo_Angle(2, servo_Init + deviation);
256.       delay(80);
257.       int a = GetDistance();
258.       int b = GetDistance();
259.       int c = GetDistance();
260.       midDist = (a+b+c)/3;
261.       Serial.print("Mid:");
262.       Serial.println(midDist);
263.       // Servo_1_Angle(servo_Init); // front wheel
264.       Motor(1,1,0); //Stop the car
265.       Motor(2,1,0);
266.
267.       if (midDist > avoid_Dist){
268.           // Servo_1_Angle(servo_Init+ deviation); // front wheel
269.           Servo_Angle(1, servo_Init + deviation); // front wheel
270.           Motor(1,1,Speed); //forward
271.           Motor(2,1,Speed);
272.       }
273.       else if (midDist <= avoid_Dist){
274.
275.           // Serial.println("<");
276.           // Servo_1_Angle(servo_Init + deviation); // front wheel
```

```
277. Servo_Angle(1, servo_Init + deviation); // front wheel
278. Motor(1,1,0); //Stop the car
279. Motor(2,1,0);
280. // Servo_2_Angle(servo_Init - 60); // left distance.
281. Servo_Angle(2, servo_Init + deviation - 60); // left distance.
282. delay(400);
283. int a = GetDistance();
284. int b = GetDistance();
285. int c = GetDistance();
286. leftDist = (a+b+c)/3;
287. Serial.print("Left:");
288. Serial.println(leftDist);
289. // Servo_2_Angle(servo_Init + 60); // right distance.
290. Servo_Angle(2, servo_Init + deviation + 60); // right distance.
291. delay(400);
292. a = GetDistance();
293. b = GetDistance();
294. c = GetDistance();
295. rightDist = (a+b+c)/3;
296. Serial.print("Right:");
297. Serial.println(rightDist);
298. // Servo_2_Angle(servo_Init); // back to mid.
299. Servo_Angle(2, servo_Init + deviation); // back to mid.
300.
301. if ((leftDist < avoid_Dist)&&(rightDist < avoid_Dist)){ // Judgment left and right.
302.     if (leftDist >= rightDist){
303.         // There are obstacles on the right backward to the left.
304.         // Servo_1_Angle(servo_Init + wheel_Steering + deviation); //turn left backwa
            rd
305.         Servo_Angle(1, servo_Init + deviation + wheel_Steering); // turn left back
            ward
306.         Motor(1,-1,Speed); //backward
307.         Motor(2,-1,Speed);
308.         delay(500);
309.     }
310.     else{ //There are obstacles on the left.
311.
312.         // Serial.println("turn right backward");
313.         // Servo_1_Angle(servo_Init - wheel_Steering + deviation); //turn right backw
            ard
314.         Servo_Angle(1, servo_Init + deviation - wheel_Steering); // turn right bac
```

```
kward
315.      Motor(1,-1,Speed); //backward
316.      Motor(2,-1,Speed);
317.      delay(500);
318.  }
319.  }
320.  else if ((leftDist > avoid_Dist)&&(rightDist <= avoid_Dist)){
321.      if (midDist < minDist){ // Obstacle ahead
322.          // Serial.println("backward");
323.          // Servo_1_Angle(servo_Init+ deviation); // backward
324.          Servo_Angle(1, servo_Init + deviation); // backward
325.          Motor(1,-1,Speed);
326.          Motor(2,-1,Speed);
327.          delay(400);
328.      }
329.      // Serial.println("turn left backward");
330.      // Servo_1_Angle(servo_Init + wheel_Steering + deviation); // turn left backwar
d
331.      Servo_Angle(1, servo_Init + deviation + wheel_Steering); // turn left backwa
rd
332.      Motor(1,-1,Speed);
333.      Motor(2,-1,Speed);
334.      delay(500);
335.  }
336.  else if ((leftDist <= avoid_Dist) &&(rightDist > avoid_Dist)){ // There are obstacl
es on the left.
337.      if (midDist < minDist){ // Obstacle ahead
338.          // Serial.println(" backward");
339.          // Servo_1_Angle(servo_Init + deviation); // backward
340.          Servo_Angle(1, servo_Init + deviation); // backward
341.          Motor(1,-1,Speed);
342.          Motor(2,-1,Speed);
343.          delay(500);
344.      }
345.      // Serial.println(" turn right backward");
346.      // Servo_1_Angle(servo_Init - wheel_Steering + deviation); //turn right backwar
d
347.      Servo_Angle(1, servo_Init + deviation - wheel_Steering); // turn right backw
ard
348.      Motor(1,-1,Speed); //backward
349.      Motor(2,-1,Speed);
350.      delay(400);
```

```
351.     }
352.     else if ((leftDist >= avoid_Dist) &&( rightDist >= avoid_Dist)){
353.         if (leftDist > rightDist){ // The distance to the right is greater than the left
354.             if (midDist < minDist){
355.                 // Servo_1_Angle(servo_Init+ deviation); // backward
356.                 Servo_Angle(1, servo_Init + deviation); // backward
357.                 Motor(1,-1,Speed);
358.                 Motor(2,-1,Speed);
359.                 delay(500);
360.             }
361.             // Servo_1_Angle(servo_Init + wheel_Steering + deviation); // turn left backward
362.             Servo_Angle(1, servo_Init + deviation + wheel_Steering); // turn left backward
363.             Motor(1,-1,Speed);
364.             Motor(2,-1,Speed);
365.             delay(400);
366.         }
367.         else{
368.             if (midDist < minDist){
369.                 // Servo_1_Angle(servo_Init+ deviation); // backward
370.                 Servo_Angle(1, servo_Init + deviation); // backward
371.                 Motor(1,-1,Speed);
372.                 Motor(2,-1,Speed);
373.                 delay(500);
374.             }
375.             // Servo_1_Angle(servo_Init + wheel_Steering + deviation); // turn left backward
376.             Servo_Angle(1, servo_Init + deviation + wheel_Steering); // turn left backward
377.             Motor(1,-1,Speed);
378.             Motor(2,-1,Speed);
379.             delay(400);
380.         }
381.     }
382. }
383. // delay(100);
384. }
385. }
386.
387. void Light_Tracking(){
```

```
388. value_Init = GetPhotosensitive();
389. while (1){
390.     StopFunction();
391.     if (Function_flag == 1){ // Press OK, stop function.
392.         break;
393.     }
394.     value = GetPhotosensitive();
395.     if (value < (value_Init - threshold)){
396.         // Servo_1_Angle(servo_Init + wheel_Steering + deviation);
397.         Servo_Angle(1, servo_Init + deviation + wheel_Steering);
398.         Motor(1, 1, motor_speed);
399.         Motor(2, 1, motor_speed);
400.         Serial.print(value_Init);
401.         Serial.print(":");
402.         Serial.println(value);
403.
404.     }
405.     else if (value > (value_Init + threshold)){
406.         // Servo_1_Angle(servo_Init - wheel_Steering + deviation);
407.         Servo_Angle(1, servo_Init + deviation - wheel_Steering);
408.         Motor(1, 1, motor_speed);
409.         Motor(2, 1, motor_speed);
410.         Serial.print(value_Init);
411.         Serial.print(":");
412.         Serial.println(value);
413.     }
414.     else{
415.         // Servo_1_Angle(servo_Init);
416.         Servo_Angle(1, servo_Init + deviation);
417.         Motor(1, 1, 0);
418.         Motor(2, 1, 0);
419.         Serial.print(value_Init);
420.         Serial.print(":");
421.         Serial.println(value);
422.     }
423. }
424.
425. }
426.
427. int switch_irr(int irr_data)
428. {
429.     switch(irr_data)
```



```
430.  {
431.    case 16750695: return 0;
432.    case 16753245: return 1;
433.    case 16736925: return 2;
434.    case 16769565: return 3;
435.    case 16720605: return 4;
436.    case 16712445: return 5;
437.    case 16761405: return 6;
438.    case 16769055: return 7;
439.    case 16754775: return 8;
440.    case 16748655: return 9;
441.    case 16738455: return 10; // *
442.    case 16756815: return 11; // #
443.    case 16718055: return 12; // up
444.    case 16730805: return 13; // down
445.    case 16716015: return 14; // left
446.    case 16734885: return 15; // right
447.    case 16726215: return 16; // ok
448.  }
449. }
```