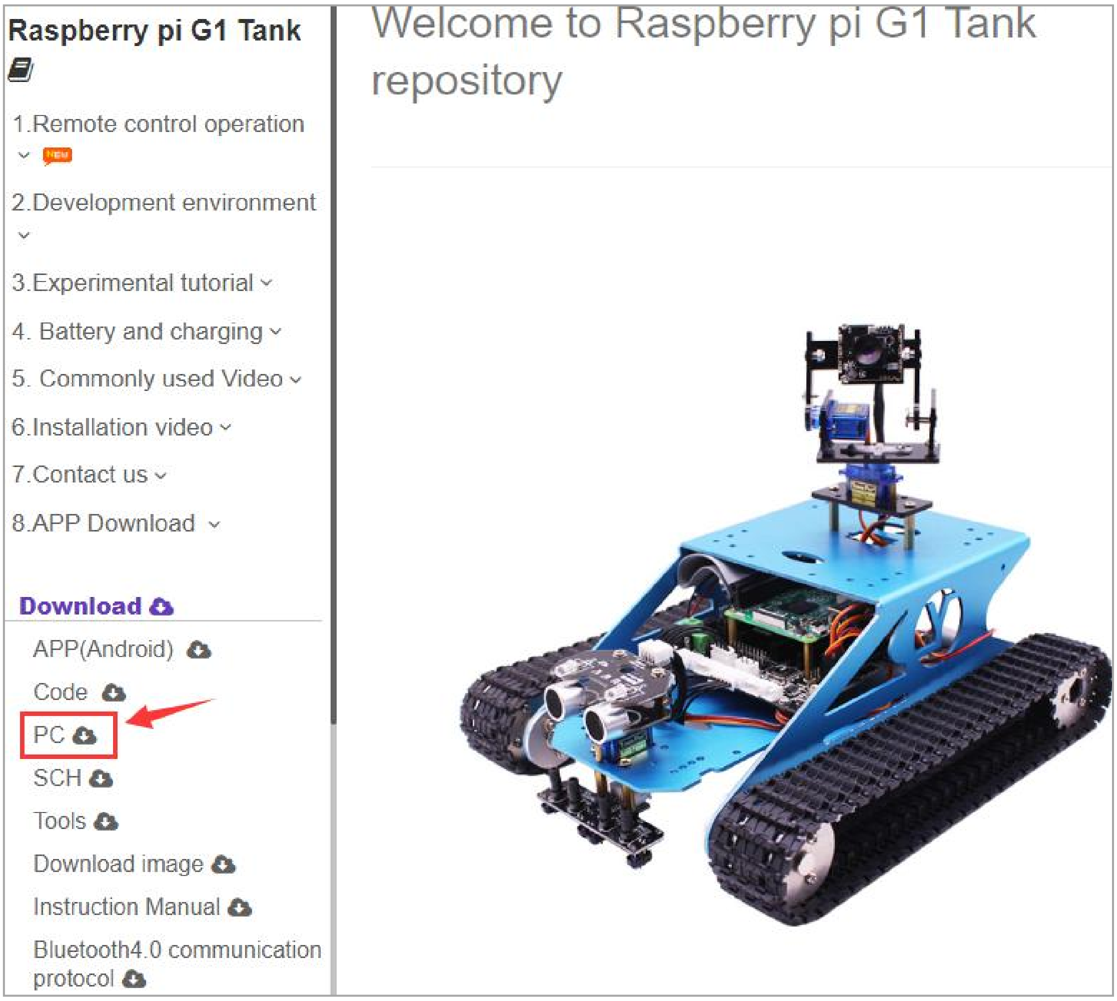
10.Raspberry Pi platform ------- TCP_control

Download Yahboom_PC software.



About the code

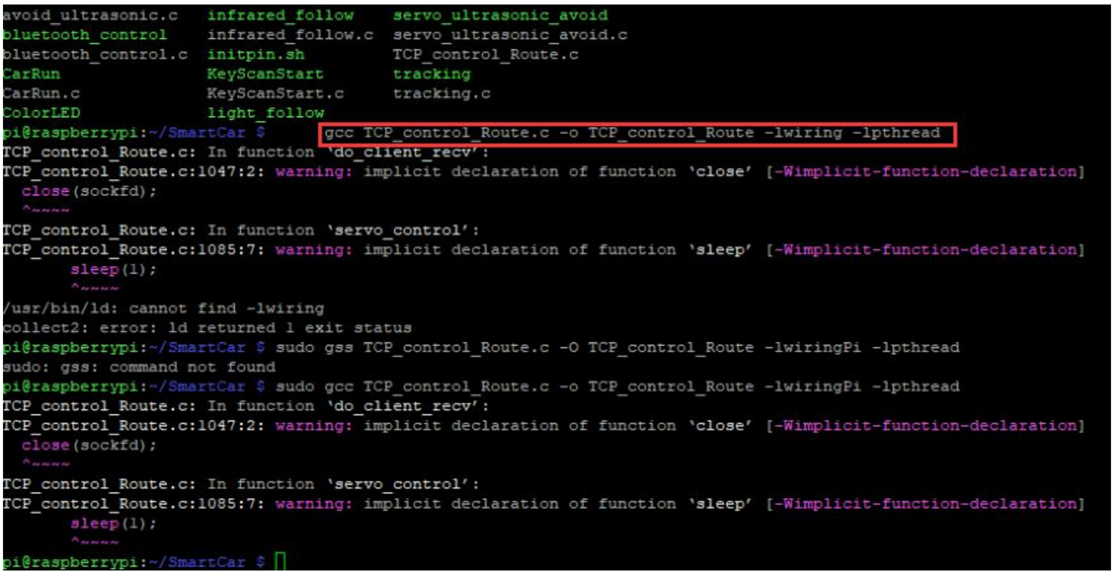
A. For .c code

(1) We need to compile this file in the Raspberry Pi system. (Note: we need to add -lwiringPi to the library file.)

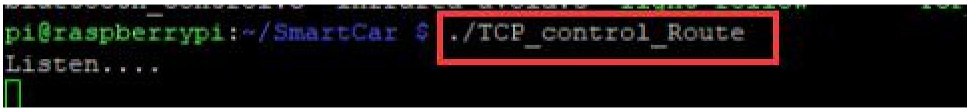
We need to input: gcc TCP_Control.c -o TCP_Control -lwiringPi -lpthread

(2)We need to run the compiled executable file in the Raspberry Pi system.We need to input: ./TCP_Control

As shown in the figure below.



```
avoid_ultrasonic.c  infrared_follow    servo_ultrasonic_avoid
bluetooth_control   infrared_follow.c  servo_ultrasonic_avoid.c
bluetooth_control.c initpin.sh         TCP_control_Route.c
CarRun              KeyScanStart       tracking
CarRun.c            KeyScanStart.c     tracking.c
ColorLED            light_follow
pi@raspberrypi:~/SmartCar $ gcc TCP_control_Route.c -o TCP_control_Route -lwiring -lpthread
TCP_control_Route.c: In function 'do_client_recv':
TCP_control_Route.c:1047:2: warning: implicit declaration of function 'close' [-Wimplicit-function-declaration]
  close(sockfd);
  ^~~~~
TCP_control_Route.c: In function 'servo_control':
TCP_control_Route.c:1085:7: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
       sleep(1);
       ^~~~~
/usr/bin/ld: cannot find -lwiring
collect2: error: ld returned 1 exit status
pi@raspberrypi:~/SmartCar $ sudo gss TCP_control_Route.c -O TCP_control_Route -lwiringPi -lpthread
sudo: gss: command not found
pi@raspberrypi:~/SmartCar $ sudo gcc TCP_control_Route.c -o TCP_control_Route -lwiringPi -lpthread
TCP_control_Route.c: In function 'do_client_recv':
TCP_control_Route.c:1047:2: warning: implicit declaration of function 'close' [-Wimplicit-function-declaration]
  close(sockfd);
  ^~~~~
TCP_control_Route.c: In function 'servo_control':
TCP_control_Route.c:1085:7: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
       sleep(1);
       ^~~~~
pi@raspberrypi:~/SmartCar $ 
```



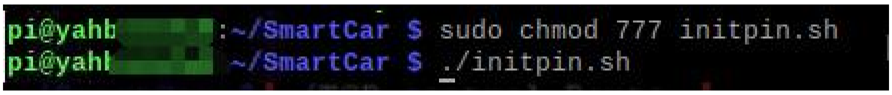
```
pi@raspberrypi:~/SmartCar $ ./TCP_control_Route
Listen....
```

(3)We can input: ctrl+c to stop this process, which mean is send a signal to the linux kernel to terminate the current process, but the state of the relevant pin is uncertain at this time, we also need to run a script to initialize all pins.

(Note:The initpin.sh script file is included in the SmartCar/python directory.)

You need to input: sudo chmod 777 initpin.sh

./initpin.sh

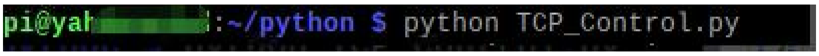


```
pi@yahb[redacted]:~/SmartCar $ sudo chmod 777 initpin.sh
pi@yahb[redacted] ~/SmartCar $ ./initpin.sh
```

B. For python code

(1) We need to input following command to run python code.

python TCP_Control.py

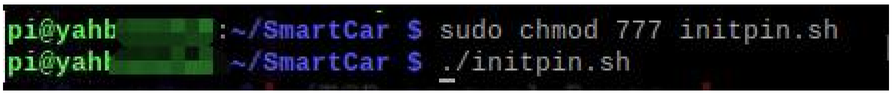


```
pi@yah[redacted]:~/python $ python TCP_Control.py
```

(2) We can input: ctrl+c to stop this process, which mean is send a signal to the linux kernel to terminate the current process, but the state of the relevant pin is uncertain at this time, we also need to run a script to initialize all pins.

(3) You need to input: sudo chmod 777 initpin.sh

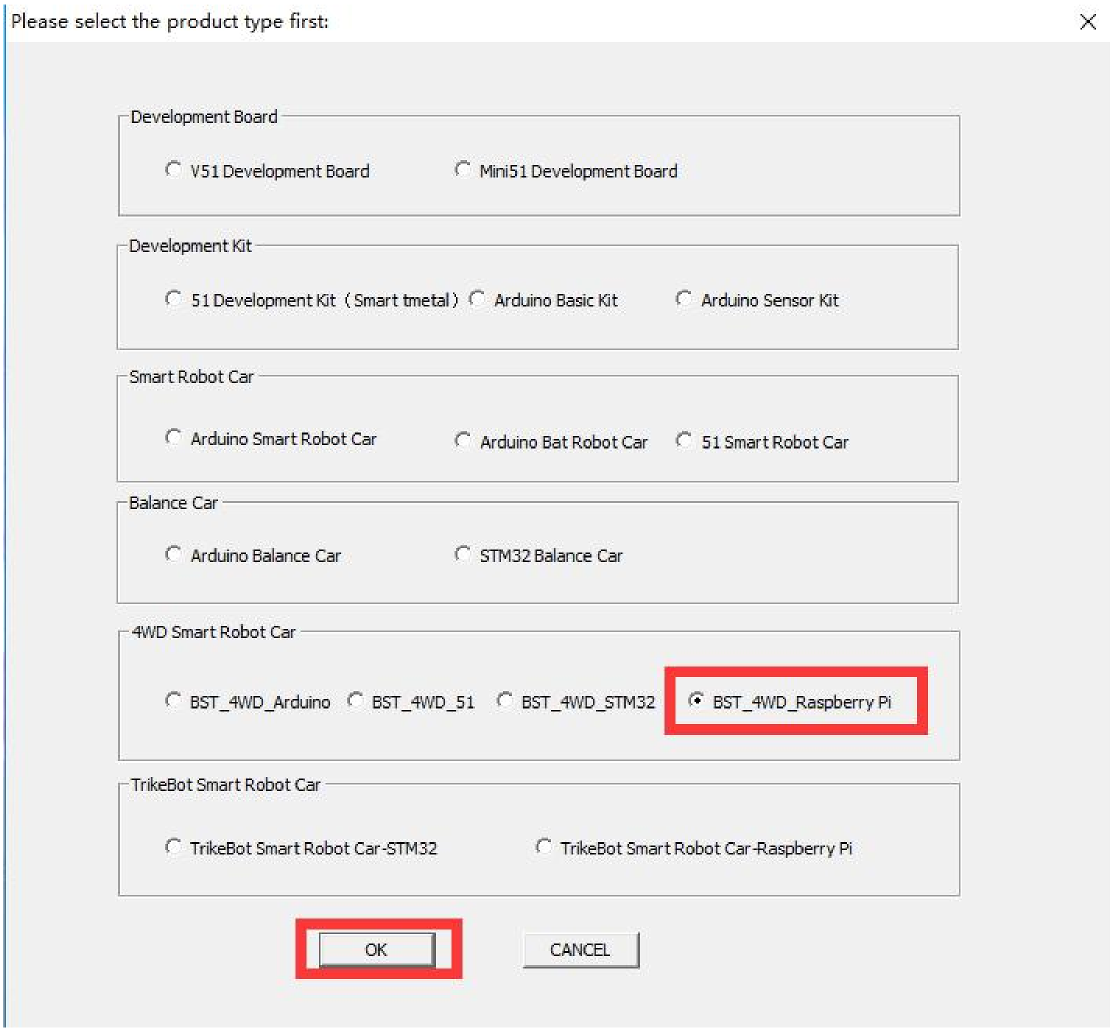
./initpin.sh



```
pi@yahb[redacted]:~/SmartCar $ sudo chmod 777 initpin.sh
pi@yahb[redacted] ~/SmartCar $ ./initpin.sh
```

About using Yahboom_PC software

1)We need to open the host computer software to choose BST_4WD_Raspberry Pi.



Please select the product type first:

Development Board

○ V51 Development Board ○ Mini51 Development Board

Development Kit

○ 51 Development Kit（Smart tmetal）○ Arduino Basic Kit ○ Arduino Sensor Kit

Smart Robot Car

○ Arduino Smart Robot Car ○ Arduino Bat Robot Car ○ 51 Smart Robot Car

Balance Car

○ Arduino Balance Car ○ STM32 Balance Car

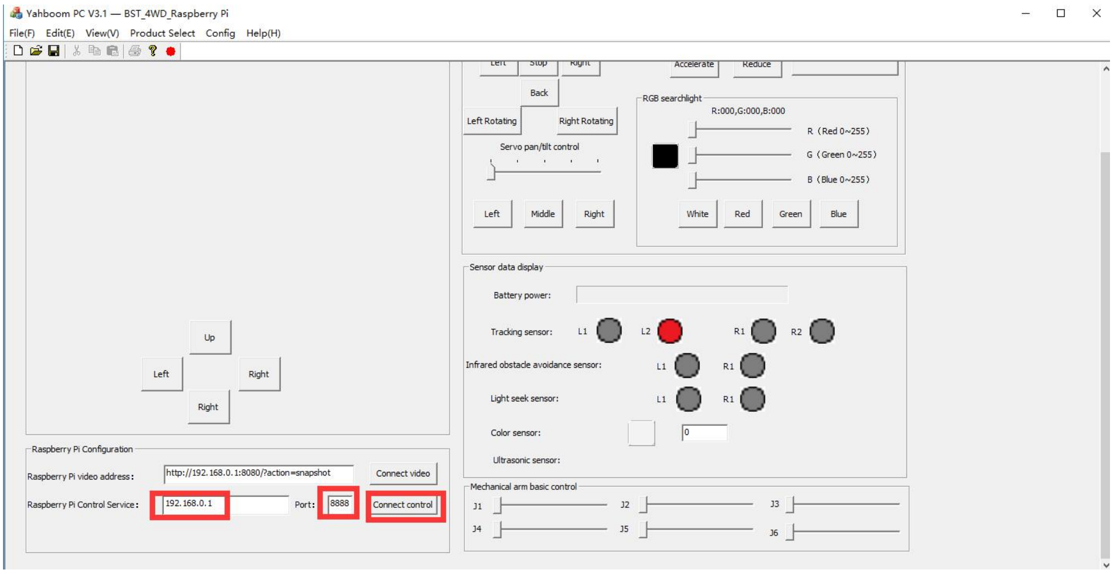
4WD Smart Robot Car

○ BST_4WD_Arduino ○ BST_4WD_51 ○ BST_4WD_STM32 ● BST_4WD_Raspberry Pi

TrikeBot Smart Robot Car

○ TrikeBot Smart Robot Car-STM32 ○ TrikeBot Smart Robot Car-Raspberry Pi

2) Click **[Connect Video]** to see the video picture captured by the camera.



Yahboom PC V3.1 — BST_4WD_Raspberry Pi

RGB searchlight R:000,G:000,B:000

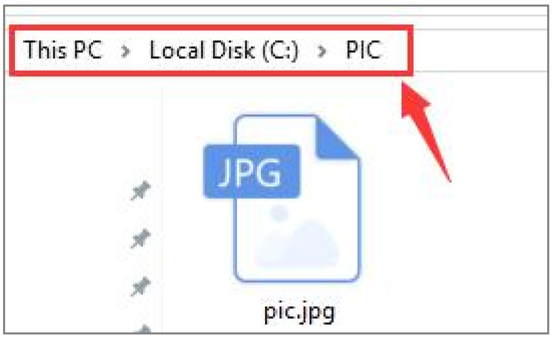
Raspberry Pi Configuration

Raspberry Pi video address: http://192.168.0.1:8080/?action=snapshot

Raspberry Pi Control Service: 192.168.0.1 Port: 8888

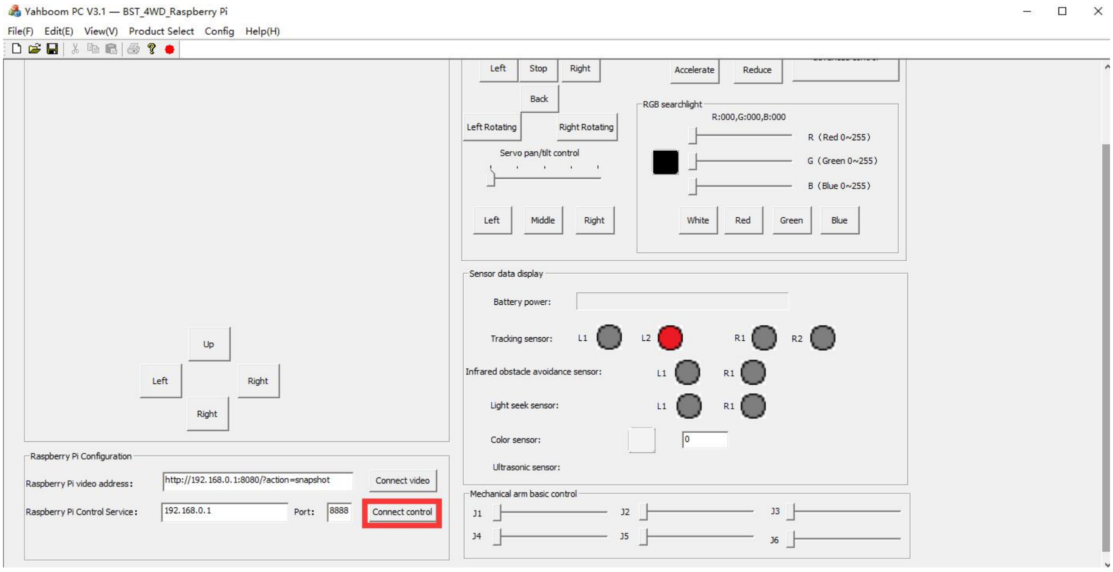
Color sensor: 0

3) Click **[Connect Video]** again to close the video picture captured by the camera. If the you cannot see the video, you needs to manually create a **PIC folder** in the **C** drive, and then create a new one **Pic.jpg file** in the folder. As shown below.



Then, you must open this file read and write permissions.

4) Click **[Connect control]** to enter remote control car mode.

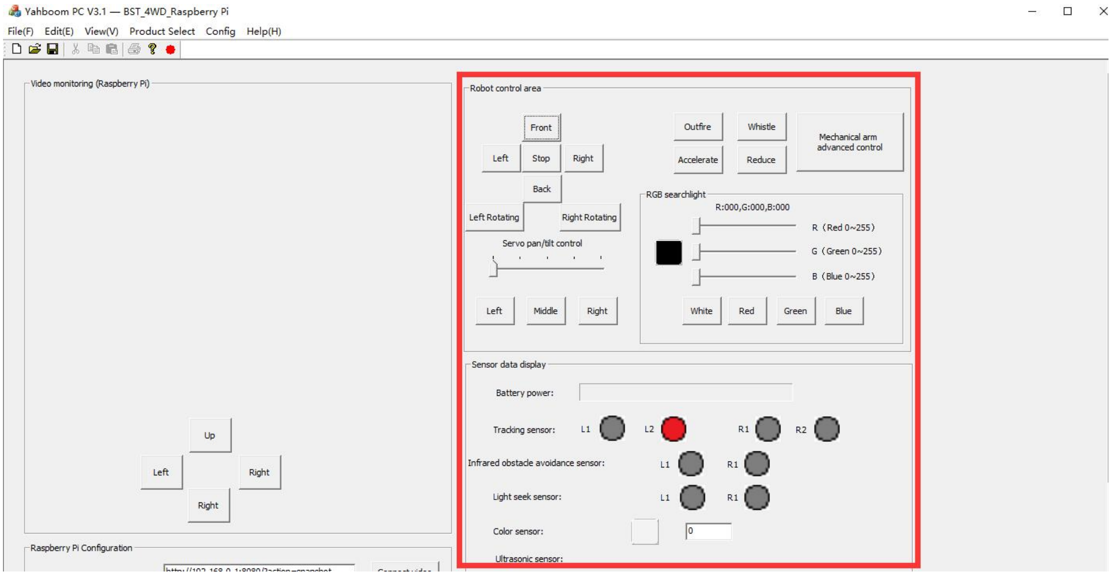


Raspberry Pi video address: http://192.168.0.1:8080/?action=snapshot

Raspberry Pi Control Service: 192.168.0.1 Port: 8888

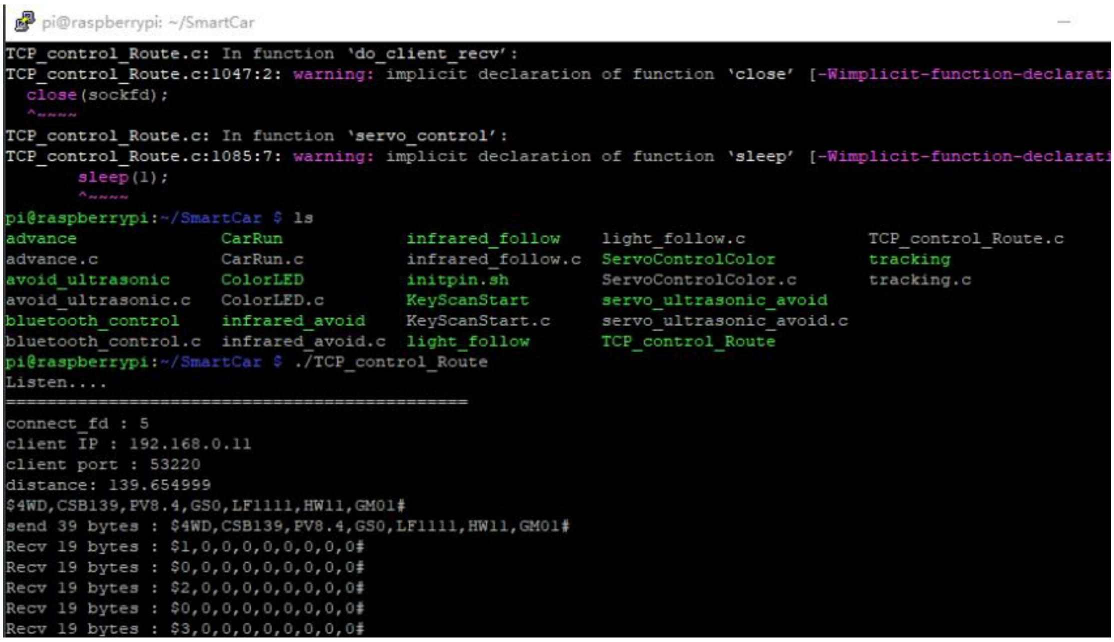
Color sensor: 0

5) You can control the car in the area on the right.



Color sensor: 0

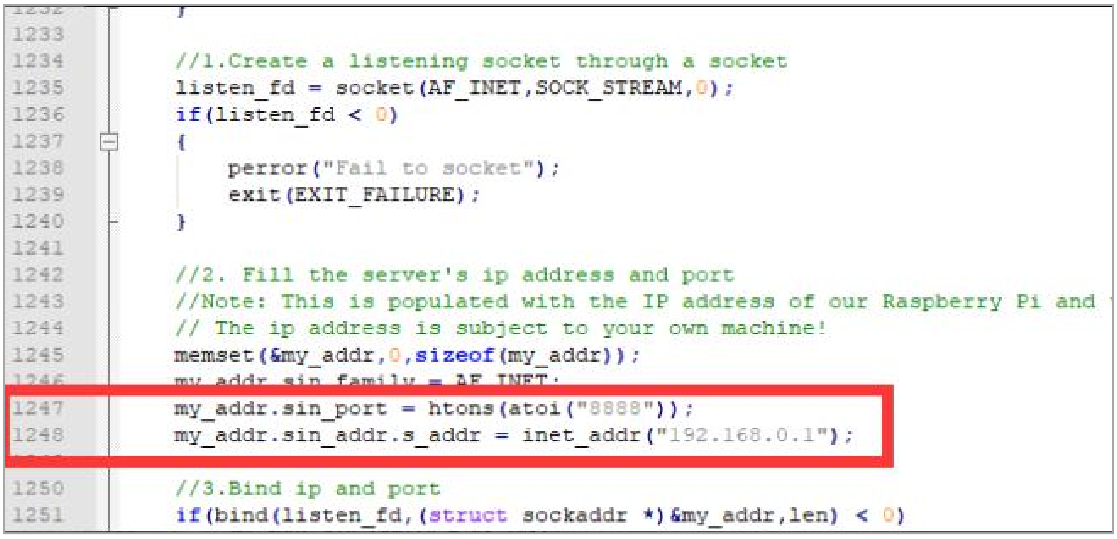
At the same time, the Raspberry Pi system will have the following tips:



pi@raspberrypi: ~/SmartCar

```
TCP_control_Route.c: In function ‘do_client_recv’:
TCP_control_Route.c:1047:2: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declarati
  close(sockfd);
  ^~~~~
TCP_control_Route.c: In function ‘servo_control’:
TCP_control_Route.c:1085:7: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declarati
       sleep(1);
       ^~~~~
pi@raspberrypi:~/SmartCar $ ls
advance              CarRun            infrared_follow    light_follow.c            TCP_control_Route.c
advance.c            CarRun.c          infrared_follow.c  ServoControlColor         tracking
avoid_ultrasonic     ColorLED          initpin.sh         ServoControlColor.c       tracking.c
avoid_ultrasonic.c   ColorLED.c        KeyScanStart       servo_ultrasonic_avoid
bluetooth_control    infrared_avoid    KeyScanStart.c     servo_ultrasonic_avoid.c
bluetooth_control.c  infrared_avoid.c  light_follow       TCP_control_Route
pi@raspberrypi:~/SmartCar $ ./TCP_control_Route
Listen....
=============================================
connect_fd : 5
client IP : 192.168.0.11
client port : 53220
distance: 139.654999
$4WD,CSB139,PV8.4,GS0,LF1111,HW11,GM01#
send 39 bytes : $4WD,CSB139,PV8.4,GS0,LF1111,HW11,GM01#
Recv 19 bytes : $1,0,0,0,0,0,0,0,0,0#
Recv 19 bytes : $0,0,0,0,0,0,0,0,0,0#
Recv 19 bytes : $2,0,0,0,0,0,0,0,0,0#
Recv 19 bytes : $0,0,0,0,0,0,0,0,0,0#
Recv 19 bytes : $3,0,0,0,0,0,0,0,0,0#
```

Note: If you want to configure the port and IP address in the code file. You can modify the code as shown below.



```
    }

    //1.Create a listening socket through a socket
    listen_fd = socket(AF_INET,SOCK_STREAM,0);
    if(listen_fd < 0)
    {
        perror("Fail to socket");
        exit(EXIT_FAILURE);
    }

    //2. Fill the server's ip address and port
    //Note: This is populated with the IP address of our Raspberry Pi and
    // The ip address is subject to your own machine!
    memset(&my_addr,0,sizeof(my_addr));
    my_addr.sin_family = AF_INET;
    my_addr.sin_port = htons(atoi("8888"));
    my_addr.sin_addr.s_addr = inet_addr("192.168.0.1");

    //3.Bind ip and port
    if(bind(listen_fd,(struct sockaddr *)&my_addr,len) < 0)
```