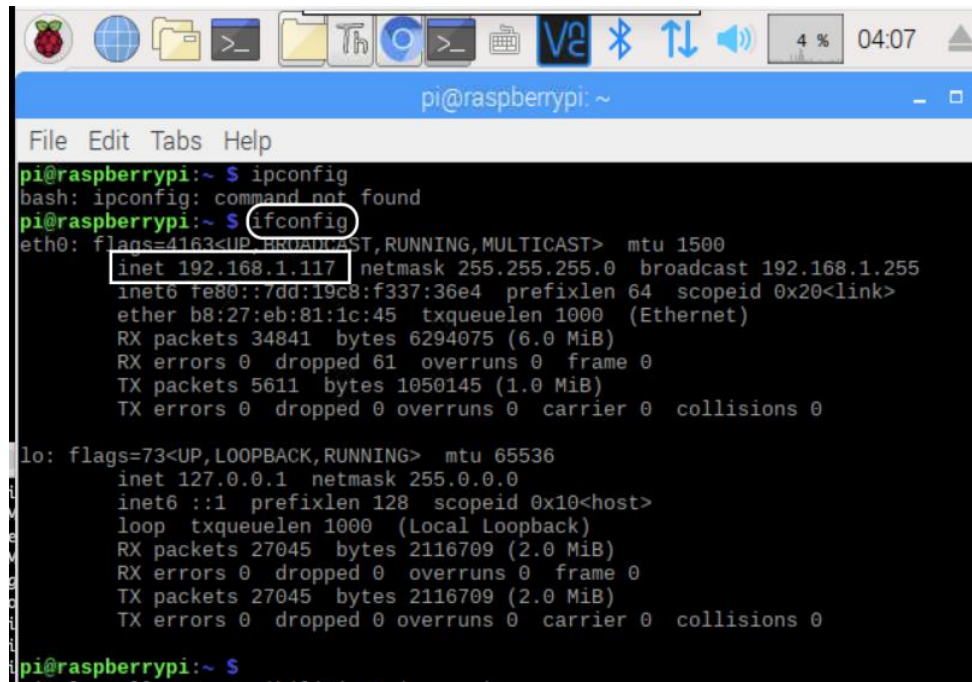


PC-side color tracking

In this chapter, we will connect the Raspberry Pi to the PC, the Raspberry Pi as the TCP/IP socket server, the PC as the client, start the OpenCV control camera on the PC. And send data to the Raspberry Pi server by `Socket.send()` function. Raspberry Pi will control servo by a simple count.

We need to input this command at the terminal :

ifconfig



```
pi@raspberrypi:~ $ ifconfig
bash: ifconfig: command not found
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.117 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::7dd:19c8:f337:36e4 prefixlen 64 scopeid 0x20<link>
    ether b8:27:eb:81:1c:45 txqueuelen 1000 (Ethernet)
    RX packets 34841 bytes 6294075 (6.0 MiB)
    RX errors 0 dropped 61 overruns 0 frame 0
    TX packets 5611 bytes 1050145 (1.0 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 27045 bytes 2116709 (2.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 27045 bytes 2116709 (2.0 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

pi@raspberrypi:~ $
```

Figure 1-1

The Raspberry pi socket server source code of the program is located at:

/home/pi/Adafruit_PCA9685/socket_servo.py

The IP address in the program is the IP address of the host.

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4     Created on Tue Nov  6 01:18:45 2018
5     * @par Copyright (C): 2010-2019, Shenzhen Yahboom Tech
6     * @file          socket_servoMotor
7     * @version       V1.0
8     * @details
9     * @par History
10
11     @author: longfusun
12 """
13 from __future__ import division
14 import Adafruit_PCA9685
15 import time
16 import socket
17
18 address = ('192.168.1.66',7783)#Local IP address
19
20 #Complete the standard socket connection, binding, monitoring, to the raspberry
21 s = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
22 s.bind(address)
23 #Initialize servo
24 pwm = Adafruit_PCA9685.PCA9685()
25 pwm.set_pwm_freq(60)
26 pwm.set_pwm(1,0,400)
27 pwm.set_pwm(0,0,500)
28 time.sleep(1)
29 #Initialize PID
30 x=0
31 y=0
32 thisError_x=0
33 lastError_x=0
34 thisError_y=0
35 lastError_y=0
36 X_P=425
37 Y_P=425
38 flag=0
39 y=0
40
41 while 1:

```

```

42  #Socket "receive" immediately after entering the steering gear
43  data,addr=s.recvfrom(2048)
44  if not data:
45      break
46  #print("got data from",addr)
47  #Socket communication data needs to be decoded first
48  x=data.decode()
49  print(x)
50  #The values sent are x and y, and in the actual "," is used as the flag.
51  strX=str(x)
52  arr=strX.split(',')
53  #String type directly into int will report an error, so first convert to float
54  intX=int(float(arr[0]))
55  intY=int(float(arr[1]))
56  print('x:',intX,'y:',intY)
57
58  #####
59  thisError_x=intX-320
60  thisError_y=intY-240
61  pwm_x=thisError_x*6+1*(thisError_x-lastError_x)
62  pwm_y=thisError_y*6+1*(thisError_y-lastError_y)
63
64  lastError_x=thisError_x
65  lastError_y=thisError_y
66  XP=pwm_x/100
67  YP=pwm_y/100
68  X_P=X_P+int(XP)
69  Y_P=Y_P+int(YP)
70  if X_P>670:
71      X_P=650
72  if X_P<0:
73      X_P=0
74  if Y_P>670:
75      Y_P=650
76  if Y_P<50:
77      Y_P=0
78
79  #####
80  print('***',X_P,Y_P)
81  pwm.set_pwm(1,0,650-X_P)
82  pwm.set_pwm(2,0,800-Y_P)
83
84  time.sleep(0.02)
85  s.close()
86
87

```

The PC-side source code of the program is located at:

```

1  from __future__ import division
2  import cv2
3  import time
4  import numpy as np
5  import socket
6
7  addr = ('192.168.1.66',7783)#Target host IP
8  #readr = ('192.168.1.110',7780)#Host IP
9  s = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
10
11  cap = cv2.VideoCapture(0)
12  cap.set(3, 640)
13  cap.set(4, 480)
14  yellow_lower=np.array([9,135,231])
15  yellow_upper=np.array([31,255,255])
16
17  x=0;
18
19  while 1:
20      ret,frame = cap.read()
21      frame=cv2.GaussianBlur(frame,(5,5),0)
22      hsv= cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
23      mask=cv2.inRange(hsv,yellow_lower,yellow_upper)
24      mask=cv2.erode(mask,None,iterations=2)
25      mask=cv2.dilate(mask,None,iterations=2)
26      mask=cv2.GaussianBlur(mask,(3,3),0)
27      res=cv2.bitwise_and(frame,frame,mask=mask)
28      cnts=cv2.findContours(mask.copy(),cv2.RETR_EXTERNAL,
29                          cv2.CHAIN_APPROX_SIMPLE)[-2]
30      if len(cnts)>0:
31          cnt=max(cnts,key=cv2.contourArea)
32          (x,y),radius=cv2.minEnclosingCircle(cnt)
33          cv2.circle(frame,(int(x),int(y)),int(radius),(255,0,255),2)
34          #Data, the (x, y) two values are combined and converted into a string type
35          # the socket does not send a string string, but a binary code formed after utf-8 encoding.
36          print('x',x);
37          data =str(x)+' '+str(y)
38          s.sendto(data.encode("utf-8"),addr)
39          cv2.imshow("capture", frame)
40          if cv2.waitKey(1)==119:
41              break
42      cap.release()
43      cv2.destroyAllWindows()
44      s.close()

```