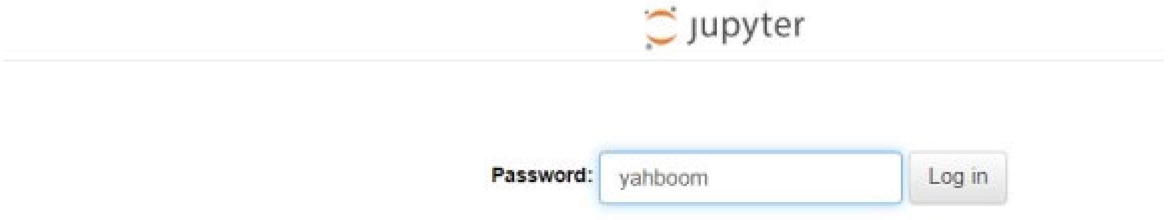
Color Tracking

1. Purpose of the experiment

This tutorial will teach you how to make the robot dog recognize red, green, blue, and yellow through a camera or color sensor, and automatically adjust its body posture to track the target. The system will calculate the target position in real time, control the robot dog to move or turn, and ensure that it always faces the recognized color object. It is suitable for visual following tasks in dynamic environments.

2. Main source code path

First, end the big program, then open the browser and enter "ip (ip is the ip of the robot dog): 8888", enter the password "yahboom" and enter



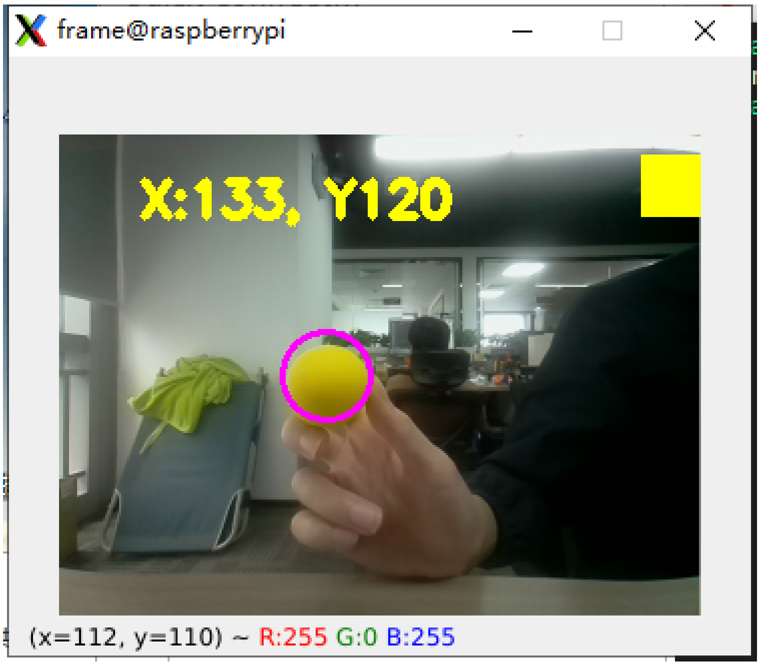
Password: yahboom

the path to ~/DOGZILLA_Lite_class/6.AI Visual Interaction Course/03.color Tracking. Open the **color_Tracking.ipynb** program and run it ， or enter it in the terminal

```
cd ~/DOGZILLA_Lite_class/6.AI Visual Interaction Course/03.color Tracking
python3 color_Tracking.py
```

3. Experimental Phenomenon

After running the source code, the body of the robot dog will follow the recognized color. Press the button on the upper right corner of the screen to switch the color. It can switch between four colors: red, yellow, blue and green.



X:133, Y120

(x=112, y=110) ~ R:255 G:0 B:255

4. Main source code analysis

```
#初始化pid Initialize pid
Px = 0.0688
Ix = 0
Dx = 0.000001
X_Middle_error = 160 #图像X轴中心 image X-axis center
X_track_PID = PID.PositionalPID(Px, Ix, Dx)

Py = 0.07
Iy = 0
Dy = 0.000001
Y_Middle_error = 120 #图像Y轴中心 image Y axis center
Y_track_PID = PID.PositionalPID(Py, Iy, Dy) #Y轴 PID参数 Y- axis PID parameters
```

Adjust the parameters of the tracking pid, which can be adjusted as needed

```
try:
    while 1:
        ret, frame = cap.read()
        frame_ = cv2.GaussianBlur(frame,(5,5),0)
        hsv = cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
        mask = cv2.inRange(hsv,color_lower,color_upper)
        mask = cv2.erode(mask,None,iterations=2)
        mask = cv2.dilate(mask,None,iterations=2)
        mask = cv2.GaussianBlur(mask,(3,3),0)
        cnts = 
cv2.findContours(mask.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)[-2]


        if g_mode == 1:
            if len(cnts) > 0:
                cnt = max (cnts, key = cv2.contourArea)
                (color_x,color_y),color_radius = cv2.minEnclosingCircle(cnt)
                if color_radius > 10:
                    cv2.circle(frame,
(int(color_x),int(color_y)),int(color_radius),(255,0,255),2)

                    #### X的方向(控制左右) X direction (control left and right)
                    X_track_PID.SystemOutput = color_x #X
                    X_track_PID.SetStepSignal(X_Middle_error)
                    X_track_PID.SetInertiaTime(0.01, 0.1)

                    x_real_value = int(X_track_PID.SystemOutput)
                    x_real_value = limit_fun(x_real_value,-15,15)

                    #### Y的方向（控制上下）   Y direction (control up and down)
                    Y_track_PID.SystemOutput = color_y #Y
                    Y_track_PID.SetStepSignal(Y_Middle_error)
                    Y_track_PID.SetInertiaTime(0.01, 0.1)
                    y_real_value = int(Y_track_PID.SystemOutput)
                    y_real_value = limit_fun(y_real_value,-11,11)

                    #print(['y'],[x_real_value])
                    #print(['p'],[-y_real_value])
                    g_dog.attitude(['p','y'],[-y_real_value,x_real_value])

            else:
                color_x = 0
                color_y = 0
                g_dog.stop()

            cv2.putText(frame, "X:%d, Y%d" % (int(color_x), int(color_y)),
(40,40), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0,255,255), 3)
            t_start = time.time()
            fps = 0


        else:
            fps = fps + 1
            mfps = fps / (time.time() - t_start)
            cv2.putText(frame, "FPS " + str(int(mfps)), (40,40),
cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0,255,255), 3)


        b,g,r = cv2.split(frame)
        img = cv2.merge((r,g,b))
        if mode==1:
            cv2.rectangle(img, (290, 10), (320, 40), red, -1)
        elif mode==2:
            cv2.rectangle(img, (290, 10), (320, 40), green, -1)
        elif mode==3:
            cv2.rectangle(img, (290, 10), (320, 40), blue, -1)
        elif mode==4:
            cv2.rectangle(img, (290, 10), (320, 40), yellow, -1)
        imgok = Image.fromarray(img)
        display.ShowImage(imgok)

        r,g,b = cv2.split(img)
        framecv = cv2.merge((b,g,r))
        #显示到电脑上 Display on computer
        cv2.imshow("frame",framecv)
        if (cv2.waitKey(1)) == ord('q'):
            break
        if button.press_b():
            g_dog.stop()
            break
        if button.press_d():
            change_color()

except:
    g_dog.reset()
    cap.release()
    cv2.destroyAllWindows()
```

As long as the above source code is the tracking control part, pressing the button in the upper right corner of the screen can switch the recognition color, and the button in the lower left corner can exit the recognition.