4. Voice control color recognition

Before running this program, you need to bind the port number of the voice board and the port number of the ROS expansion board in the host, you can refer to the previous chapter to bind; when you enter the docker container, you need to mount this voice board in order to be in the docker container and recognize the voice board.

1. Function Description

After the program starts, say "Hello, Xiaoya" to the module, and the module replies "Yes" to wake up the voice board, then use the mouse to select a color in the screen (currently recognizable colors are red, green, blue and yellow), press and hold it, and then ask. "What color is this", then it will calculate the value of HSV and answer what color it is.

2. Code reference path

After entering the docker container, the source code of this function is located at.

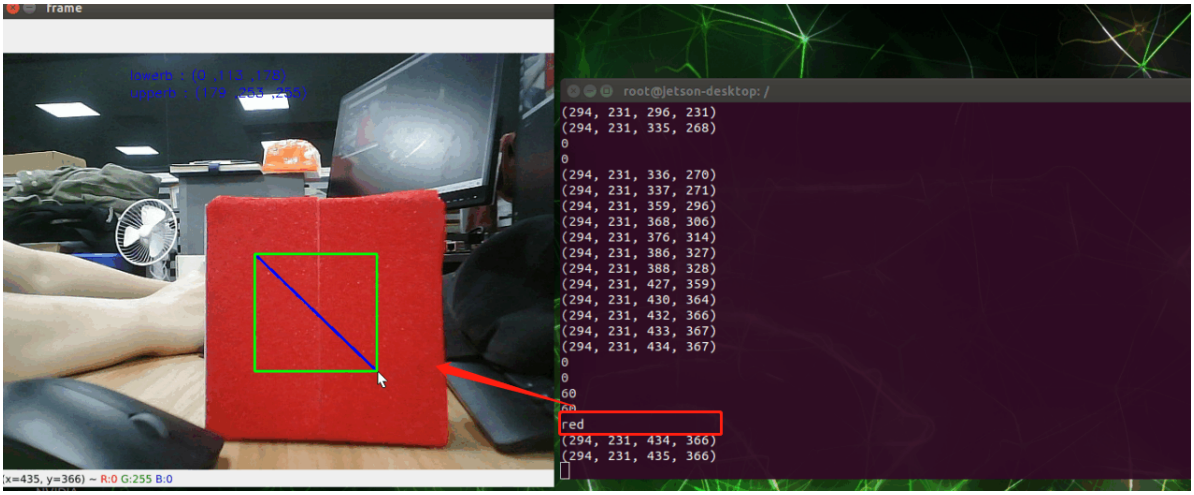
```
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_voice_ctrl/yahboomcar_voic
e_ctrl/voice_Ctrl_color_identify.py
```

3. Program startup

3.1 Startup Commands

```
cd
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_voice_ctrl/yahboomcar_voic
e_ctrl
python3 voice_Ctrl_color_identify.py
```

This part of the program does not involve the ROS part, if you want to use the framework of ros, then you can modify the program to publish the recognition results through a custom publisher, here is just to implement the process of voice color recognition.



4. Core Code Analysis

The principle of this color recognition is very simple, according to the HSV value of the selected area to determine the color, and then according to the recognition results to send the corresponding voice commands to the voice board to broadcast the recognition results, the code is as follows.

```
if self.Roi_init[0]!=self.Roi_init[2] and self.Roi_init[1]!=self.Roi_init[3]:
    HSV = cv.cvtColor(rgb_img,cv.COLOR_BGR2HSV)
    for i in range(self.Roi_init[0], self.Roi_init[2]):
        for j in range(self.Roi_init[1], self.Roi_init[3]):
            H.append(HSV[j, i][0])
            S.append(HSV[j, i][1])
            V.append(HSV[j, i][2])
    H_min = min(H); H_max = max(H)
    S_min = min(S); S_max = 253
    V_min = min(V); V_max = 255
    #print("H_max: ",H_max)
    #print("H_min: ",H_min)
    lowerb = 'lowerb : (' + str(H_min) + ' ,' + str(S_min) + ' ,' + str(V_min) +
')'
    upperb = 'upperb : (' + str(H_max) + ' ,' + str(S_max) + ' ,' + str(V_max) +
')'
    cv.putText(rgb_img, lowerb, (150, 30), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255,
0, 0), 1)
    cv.putText(rgb_img, upperb, (150, 50), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255,
0, 0), 1)
    command_result = self.spe.speech_read()
    #self.spe.void_write(56)
    if command_result !=999:
        print(command_result)
        #这里判断是否有说”这是什么颜色“
          # Here to determine if it says "What color is this?"
    if command_result == 60:
        #以下部分是判断HSV的值的范围
         # The following section is to determine the range of HSV values
        if H_min == 0 and H_max == 179 :
            #print("red")
            self.spe.void_write(61)
            print("red")
    .........
```

Note: Due to the color recognition is affected by the light intensity is relatively large, here calibration of each color of the HSV parameters are not applicable to all scenes, if the recognition of the effect is not ideal, you need to manually calibrate the parameters, refer to "Chapter 13-3, voice-controlled trolley patrol line autopilot-3.2, adjust the HSV value" course, the GUI in the HSV value, modify the corresponding color of H_min, H_max and so on. min, H_max, etc.