4.Control Muto 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 on the host machine. You can refer to the previous chapter for binding;
When entering the docker container, you need to mount the voice board to recognize the voice board in the docker container.

1. Function description

After the program is started, say "Hello, yahboom" to the module, and the module will reply "Yes" to wake up the voice board.Then use the mouse to select a color in the screen (the currently recognized colors are red, green, blue, and yellow), hold it down, and then ask, "What color is this?"Then it will calculate the HSV value and answer what color it is.

2. Code reference path

After entering the docker container, the location of the source code of this function is:

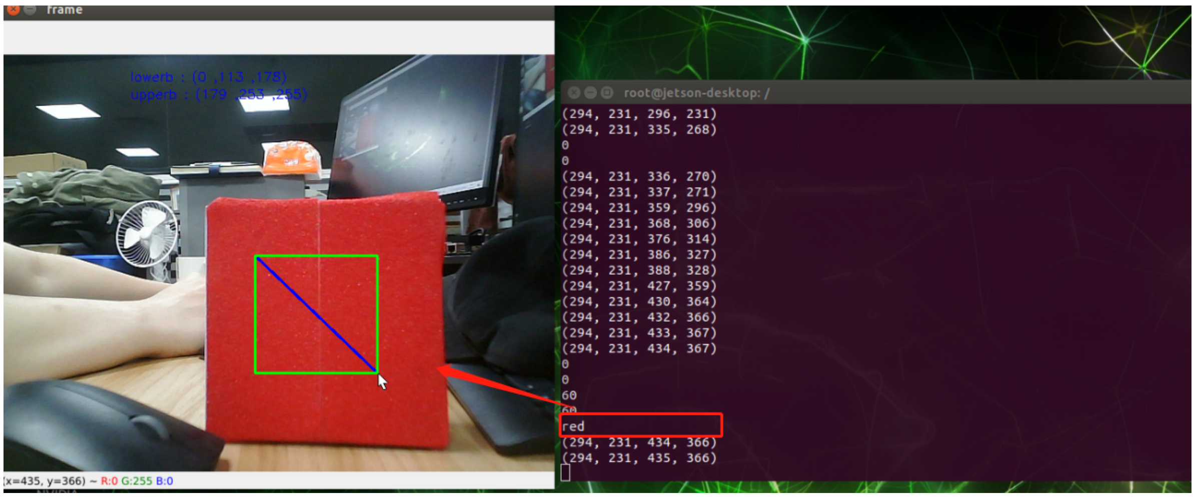
```
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_voice_ctrl/yahboomcar_voic
e_ctrl/voice_Ctrl_color_identify.py
```

3. Program startup

3.1. Start command

```
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 ros framework, you can modify the program to publish the recognition results through a custom publisher. This is just the process of realizing speech color recognition.



```
lowerb : (0 ,113 ,178)
upperb : (179 ,253 ,255)
(x=435, y=366) ~ R:0 G:255 B:0
```

```
(294, 231, 296, 231)
(294, 231, 335, 268)
0
0
(294, 231, 336, 270)
(294, 231, 337, 271)
(294, 231, 359, 296)
(294, 231, 368, 306)
(294, 231, 376, 314)
(294, 231, 386, 327)
(294, 231, 388, 328)
(294, 231, 427, 359)
(294, 231, 430, 364)
(294, 231, 432, 366)
(294, 231, 433, 367)
(294, 231, 434, 367)
0
0
60
60
red
(294, 231, 434, 366)
(294, 231, 435, 366)
```

4. Core code analysis

The principle of this color recognition is very simple. The color is judged based on the HSV value of the selected area, and then the corresponding voice command is sent to the voice board based on the recognition result, and the recognition result is broadcast. 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)
        #Determine whether it says "What color is this?"
    if command_result == 60:
        #The following part is the range for judging the value of HSV
        if H_min == 0 and H_max == 179 :
            #print("red")
            self.spe.void_write(61)
            print("red")
    .........
```

Note: Since color recognition is greatly affected by light intensity, the HSV parameters calibrated here for each color are not applicable to all scenarios. If the recognition effect is not ideal, you need to manually calibrate the parameters yourself.