4、Voice controlled car color tracking

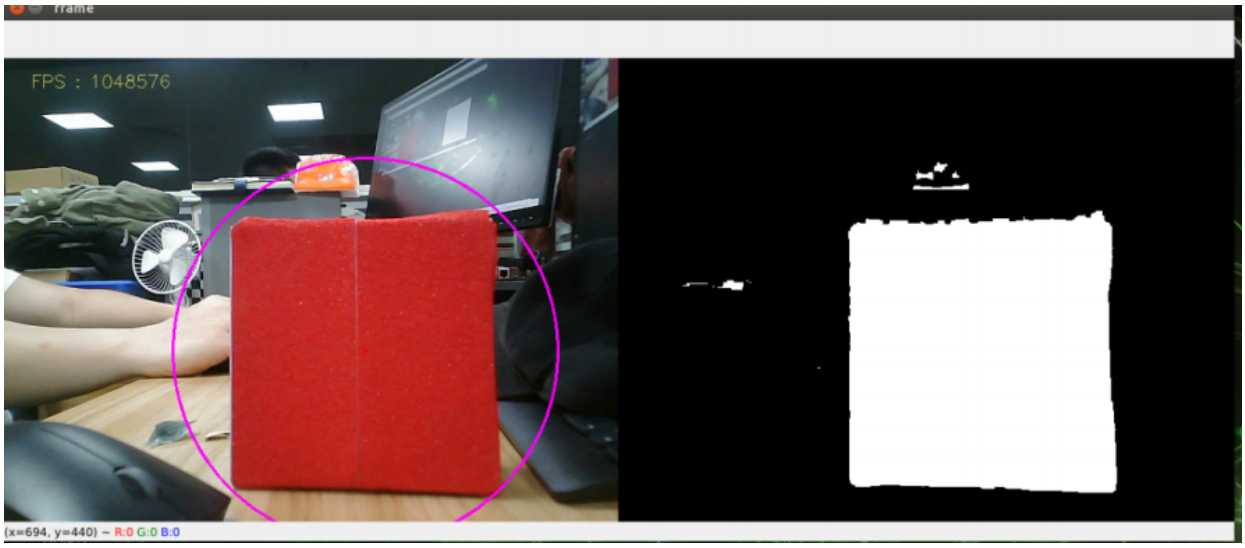
This course needs to be combined with the hardware of the Rosmaster-X3 car, and only code analysis will be done here. Firstly, let's take a look at the built-in voice commands，

functional word	Speech recognition results	Voice broadcast content
yellow following	72	OK, I found the yellow
red following	73	OK, I found the red
green following	74	OK, I found the green
follow this color	75	OK, I found this color
stop following	76	OK, it has been stoped

1、Program startup

Terminal input，

```
roslaunch yahboomcar_voice_ctrl voice_ctrl_colorTracker.launch
python ~/driver_ws/src/yahboomcar_voice_ctrl/scripts/voice_Ctrl_color_tracker.py
```



After the program starts, call "Hi Yahboom" to ROSMASTER to wake up the module. When it broadcasts "Hi , I'm here.", it indicates that the module is awakened. Taking tracking red as an example, next you can say "Sred following" to ROSMASTER, and ROSMASTER will broadcast "OK, I found the red ". Then, we release the control of ROSMASTER by pressing the R2 button on the handle, and ROSMASTER starts tracking red. If there is no remote control, you can also enter the following command through the terminal,

```
rostopic pub /JoyState std_msgs/Bool False
```

2、Core code

code path：~/driver_ws/src/yahboomcar_voice_ctrl/yahboomcar_voice_ctrl/Voice_Ctrl_colorHSV.py

Mainly parsing voice commands and loading HSVs_ The value of range

```
#Determine voice commands and load corresponding HSVs_ Range value
command_result = self.spe.speech_read()
self.spe.void_write(command_result)
if command_result == 73 :
    self.model = "color_follow_line"
    print("tracker red")
    self.hsv_range = [(20, 215, 111), (180, 253, 255)]
    self.dyn_update = True
    elif command_result == 74 :
    self.model = "color_follow_line"
    print("tracker green")
    self.hsv_range = [(44, 138, 91), (84, 255, 255)]
    self.dyn_update = True
#According to hsv_ The value of range, calculating the coordinates of the center of
the circle
rgb_img, binary, self.circle = self.color.object_follow(rgb_img,self.hsv_range)
#Publish the values of the center coordinates based on the coordinate values of the
center of the circle
if self.circle[2] != 0: threading.Thread(target=self.execute, args=(self.circle[0],
self.circle[1],self.circle[2])).start()
def execute(self, x, y, z):
    position = Position()
    position.angleX = x
    position.angleY = y
    position.distance = z
    self.pub_position.publish(position)
#With the coordinates of the center of the circle and the distance information
obtained by the depth camera, the speed is calculated based on the PID. This part of
the reference code will not be analyzed here
```