

## 4. 4. Voice control Autopilot

---

### 4. 4. Voice control Autopilot

#### 4.1. Description

#### 4.2. Steps

##### 4.2.1. function package path

##### 4.2.2. Start

##### 4.2.3. Dynamic parameter adjustment

#### 4.3.Code analysis

##### 4.3.1. Import speech recognition library and create speech recognition object

##### 4.3.2. According to the content recognized by reading the speech, modify the value of hsv\_range (process function), and then obtain the value of circle

##### 4.3.3. Release speed to chassis (execute function)

##### 4.3.4. Flowchart

##### 4.3.5. Voice module communication protocol

## 4.1. Description

Voice control robot open and close tracking red/blue/green/yellow function, and the R2 key on the handle can stop/start this function at any time.

## 4.2. Steps

### 4.2.1. function package path

```
~/yahboomcar/src/yahboomcar_voice_ctrl/
```

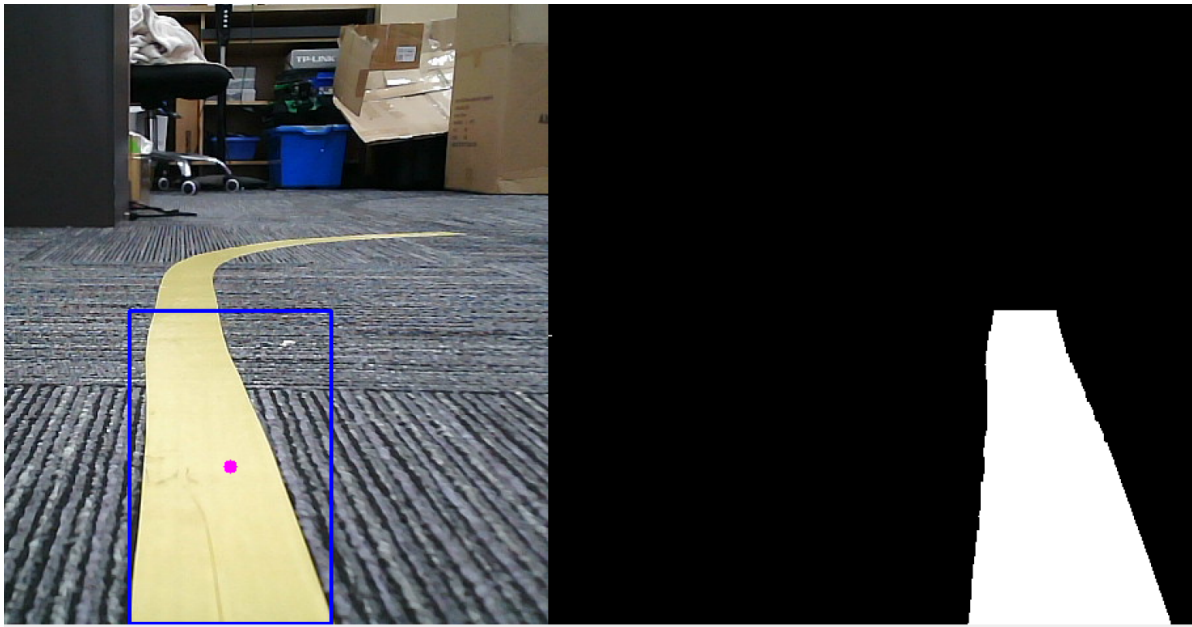
### 4.2.2. Start

```
roslaunch yahboomcar_voice_ctrl voice_ctrl_followline.launch # control
python3
yahboomcar_ws/src/yahboomcar_voice_ctrl/scripts/voice_Ctrl_follow_line.py
#Enable follow line function
```

#### \* (Take tracking red for example) \*

1. Put ROSMASTER on the yellow line, adjust the camera position, and push the camera down.
2. After the program is run, we can say "Hi Yahboom" to wake up the voice module, until it replies "Hi, i'm here", indicating that the module has been woken up.
3. We can say "Tracking the yellow line", it will reply "OK, I will track the yellow line".
4. Next, we press the R2 key on handle, then ROSMASTER starts track the yellow line. If you don't use handle, you can also start ROSMASTER by inputting the following command through the terminal.

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



If you want to cancel this color tracking function, say "Stop the tracking function", it replies "OK, tracking mode is closed". ROSMASTER will cancel this function.

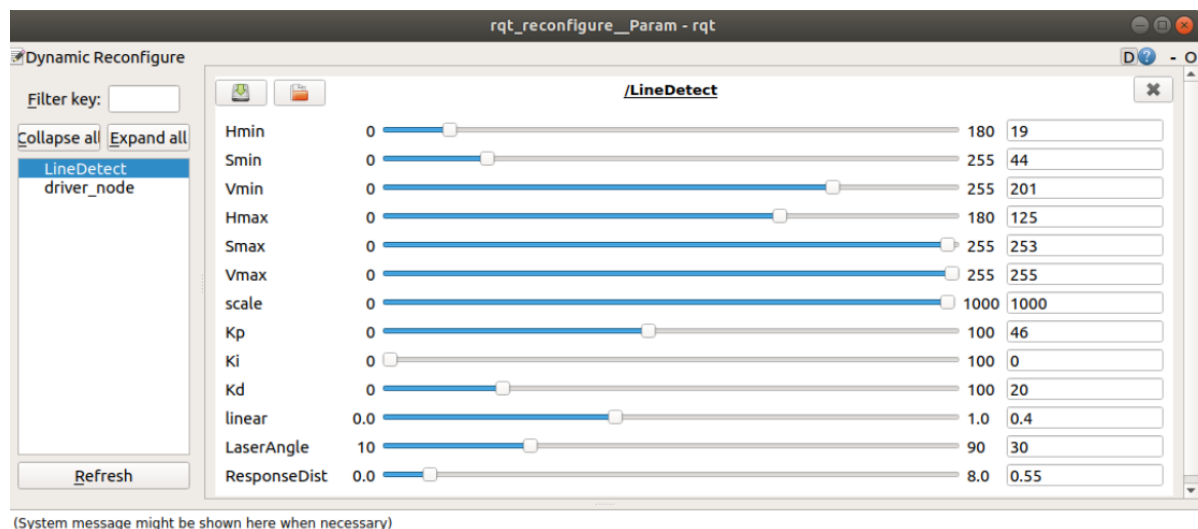
When the robot is moving, you can pause the robot by pressing the R2 key on handle again. Or input the following command to temporarily stop the robot.

```
rostopic pub /JoyState std_msgs/Bool True
```

### 4.2.3. Dynamic parameter adjustment

```
roslaunch rqt_reconfigure rqt_reconfigure
```

Open the dynamic adjustment parameters, select the LineDetect column, then you can adjust the parameters inside, adjust the parameters, manually modify them to voice\_Ctrl\_follow\_line.py, restart the code to use the adjusted parameters.



## 4.3.Code analysis

### 4.3.1. Import speech recognition library and create speech recognition object

```
from Speech_Lib import Speech
self.spe = Speech()
```

### 4.3.2. According to the content recognized by reading the speech, modify the value of hsv\_range (process function), and then obtain the value of circle

```
self.command_result = self.spe.speech_read()
self.spe.void_write(self.command_result)

if self.command_result == 23 :
    self.model = "color_follow_line"
    print("red follow line")
    self.hsv_range = [(0, 106, 175), (180, 255, 255)]

elif self.command_result == 24 :
    self.model = "color_follow_line"
    print("green follow line")
    self.hsv_range = [(55, 105, 136), (95, 255, 255)]

elif self.command_result == 25 :
    self.model = "color_follow_line"
    print("bule follow line")
    self.hsv_range = [(55, 134, 218), (125, 253, 255)]

elif self.command_result == 26 :
    self.model = "color_follow_line"
    print("yellow follow line")
    self.hsv_range = [(17, 55, 187), (81, 255, 255)]
rgb_img, binary, self.circle = self.color.line_follow(rgb_img, self.hsv_range)
```

**Note:** The HSV value can be modified according to the actual situation. Since the camera is more sensitive to light, it may appear that the HSV value here may be different, and the line tracking effect is not very good. The user can adjust the maximum and minimum values of HSV with dynamic parameters, modify the max and min values of the calibrated color HSV into the above code, and use the calibrated values after restarting the code.

### 4.3.3. Release speed to chassis (execute function)

```
if color_radius == 0: self.ros_ctrl.pub_cmdVel.publish(Twist())
else:
    twist = Twist()
    b = Bool()
    [z_Pid, _] = self.PID_controller.update([(point_x - 320)/16, 0])
    if self.img_flip == True: twist.angular.z = +z_Pid
    else: twist.angular.z = -z_Pid
    twist.linear.x = self.linear
    if self.warning > 10:
```

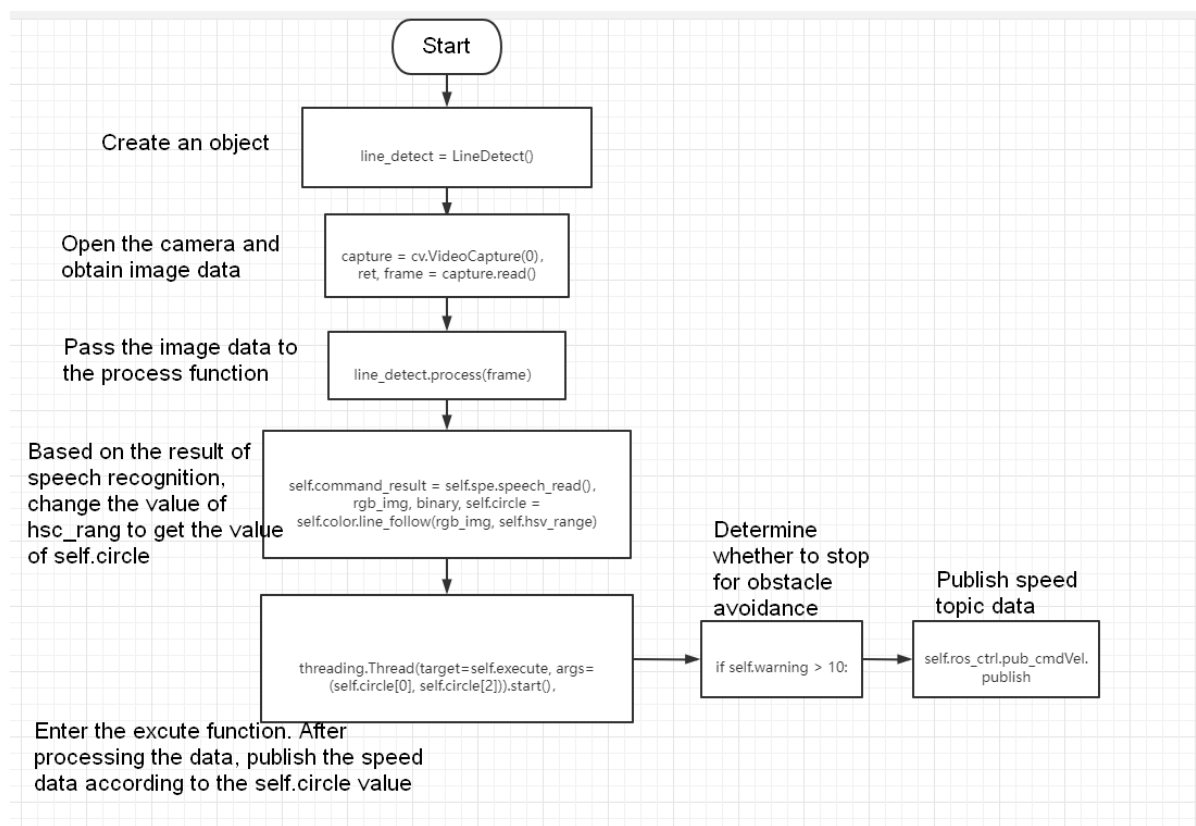
```

        rospy.loginfo("Obstacles ahead !!!")
        self.ros_ctrl.pub_cmdVel.publish(Twist())
        self.Buzzer_state = True
        b.data = True
        self.pub_Buzzer.publish(b)
    else:
        if self.Buzzer_state == True:
            b.data = False
            for i in range(3): self.pub_Buzzer.publish(b)
            self.Buzzer_state = False
        self.ros_ctrl.pub_cmdVel.publish(twist)

```

According to the obtained value of `self.circle`, it is passed to execute as an actual parameter, data is processed, it is judged whether to avoid obstacles, and finally the speed topic data is released.

#### 4.3.4. Flowchart



Code path:

```
~/yahboomcar/src/yahboomcar_voice_ctrl/scripts/voice_ctrl_follow_line.py
```

#### 4.3.5. Voice module communication protocol

Voice command	Speech Recognition Module Results	Voice broadcast content
Stop the tracking function	22	OK, tracking mode is closed
Start track red line	23	OK, I will track the red line
Start track green line	24	OK, I will track the green line
Start track blue line	25	OK, I will track the blue line
Start track yellow line	26	OK, I will track the yellow line