6. Patrol line autopilot

1. Program Function Description

After the program starts, adjust the tilt angle of the camera, break the camera down, so that the camera can see the line, then click on the image window, press the r key to enter the color selection mode; then in the area of the line in the screen, box out the color of the required patrol line, after releasing the mouse will automatically load the processed image; finally, press the space bar to open the patrol line function. In the process of running, the car will stop when it encounters obstacles and the buzzer will sound; after opening the joystick control program, the R2 key on the joystick can pause the movement of the car.

2. Program 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_linefollow/yahboomcar_line
follow
```

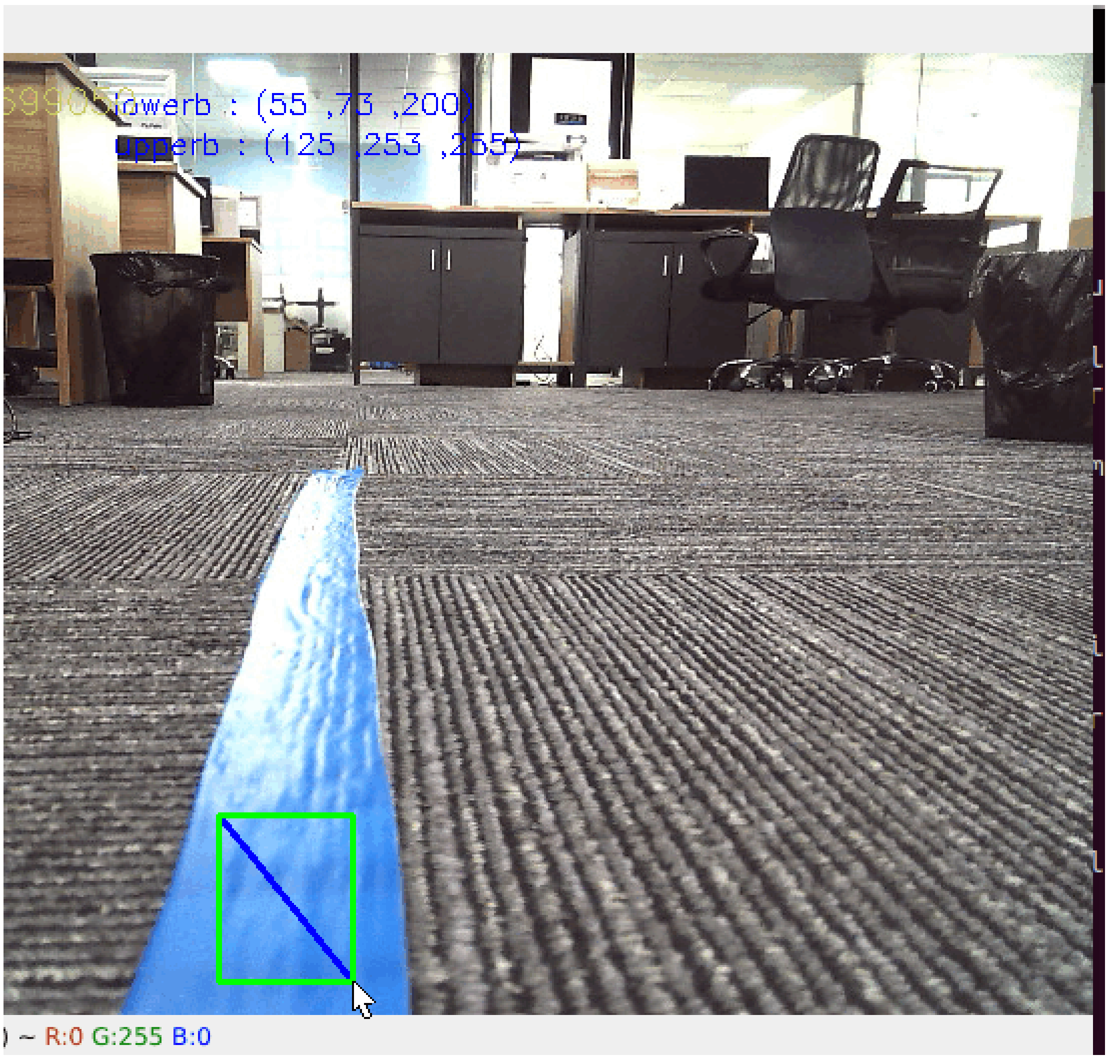
3. Program startup

3.1 Startup command

After entering the docker container, according to the actual model and radar model, terminal input, the

```
#启动小车底盘 #Start the trolley chassis
ros2 run yahboomcar_bringup Ackman_driver_R2
#启动A1雷达 # Activate A1 lidar
ros2 launch sllidar_ros2 sllidar_launch.py
#启动4ROS雷达# Activate 4ROS lidar
ros2 launch ydlidar_ros2_driver ydlidar_launch.py
#启动手柄控制节点 #Start the joystick control node
ros2 run yahboomcar_ctrl yahboom_joy_R2
ros2 run joy joy_node
#启动小车巡线程序 A1雷达
#Start the trolley patrol program A1 lidar
ros2 run yahboomcar_linefollow follow_line_a1_R2
#启动小车巡线程序 4ROS雷达
#Start the trolley patrol program 4ROS lidar
ros2 run yahboomcar_linefollow follow_line_4ROS_R2
```

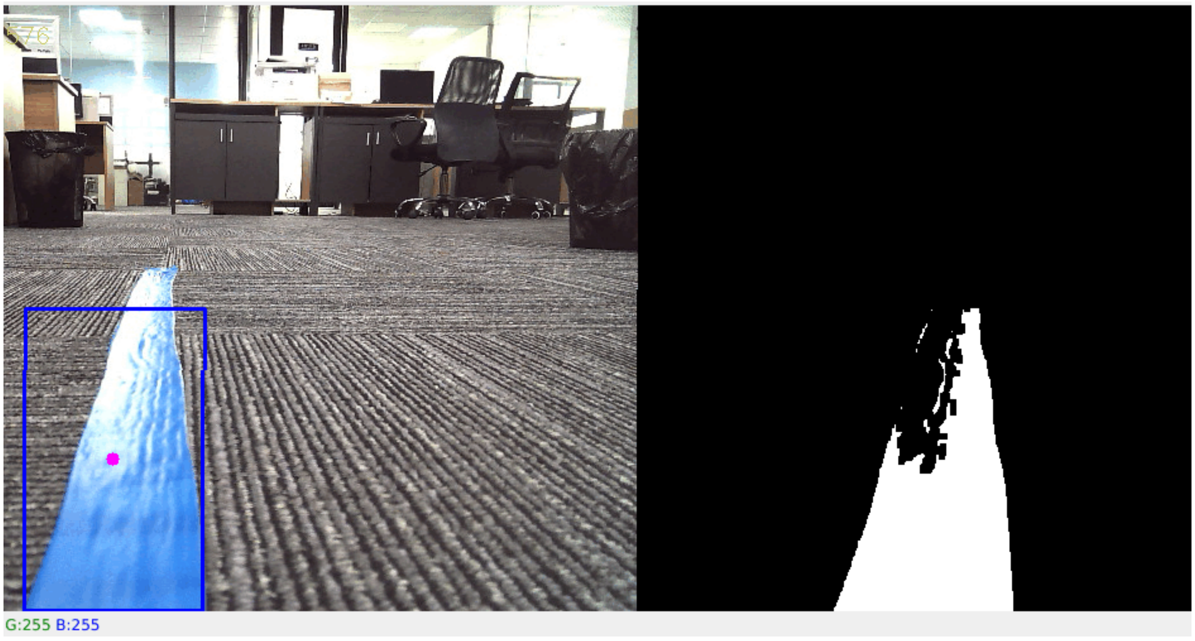
Take the Patrol Blue Line, for example.



```
lowerb : (55 ,73 ,200)
upperb : (125 ,253 ,255)
```

```
) ~ R:0 G:255 B:0
```

After pressing the r key, select the blue line area as shown above, release the mouse after selecting the



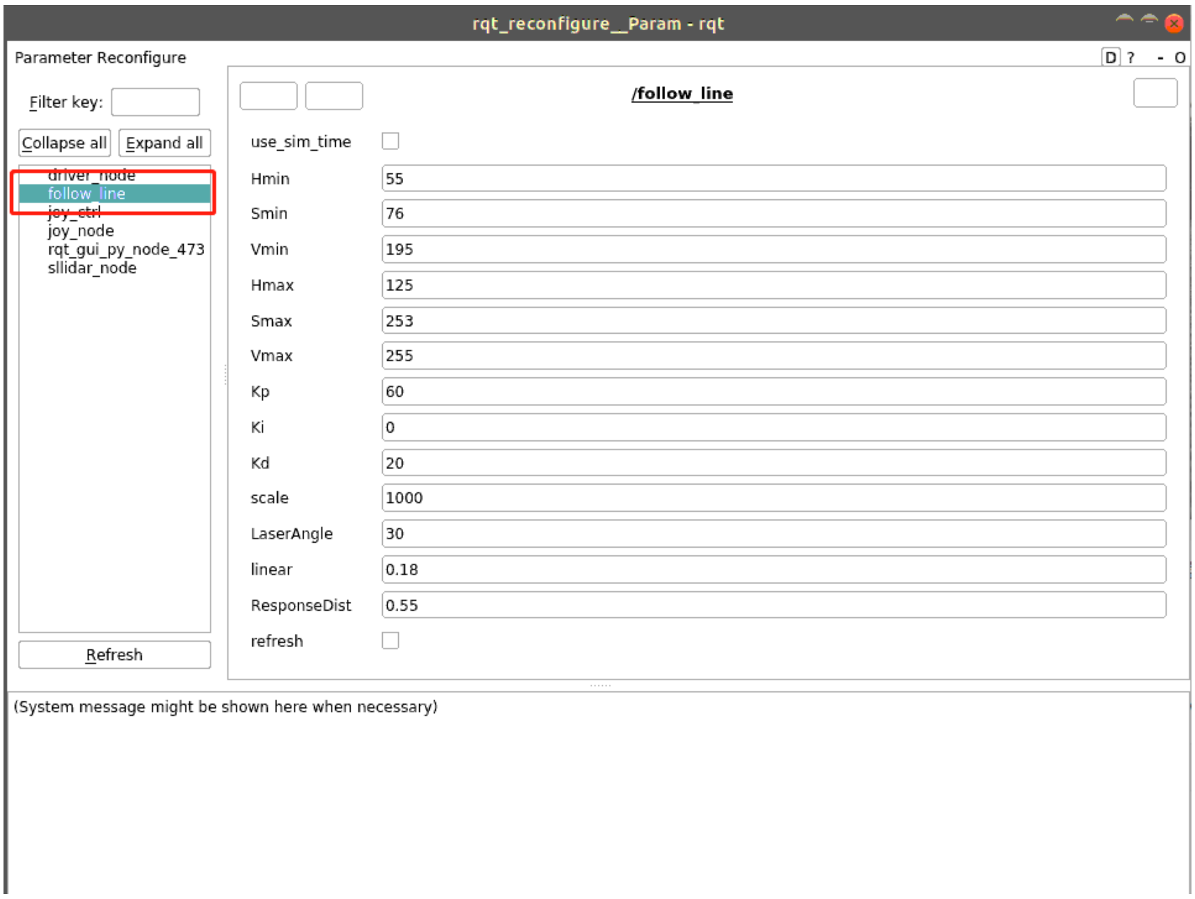
```
G:255 B:255
```

As shown in the picture above, the right side shows the processed image, it will show the blue line part. Then press the space bar to start calculating the speed, and the trolley patrols the line and drives automatically.

3.2 Dynamic Parameter Adjustment

You can adjust the relevant parameters through the dynamic parameterizer, docker terminal input.

```
ros2 run rqt_reconfigure rqt_reconfigure
```



driver_node
follow_line
joy_ctrl
joy_node
rqt_gui_py_node_473
sllidar_node

/follow_line

use_sim_time	☐
Hmin	55
Smin	76
Vmin	195
Hmax	125
Smax	253
Vmax	255
Kp	60
Ki	0
Kd	20
scale	1000
LaserAngle	30
linear	0.18
ResponseDist	0.55
refresh	☐

(System message might be shown here when necessary)

The adjustable parameters are.

parameters	instruction
Kp	P value of PID
Ki	I value of PID
Kd	D value of PID
scale	PID adjustment ratio coefficient
LaserAngle	Angle of lidar detection
linear	Linear velocity magnitude
ResponseDist	Obstacle avoidance detection distance
refresh	Refresh Parameters button

4. Core Code

Let's first sort out the principle of the implementation of line patrolling by

- Calculate the offset between the center of the line and the center of the image.
- Calculating the value of the angular velocity according to the coordinate offset.
- The velocity is issued to drive the trolley.

Calculate the center coordinates.

```
#计算hsv值 #Calculate hsv values
rgb_img, self.hsv_range = self.color.Roi_hsv(rgb_img, self.Roi_init)
#计算self.circle，计算出X的坐标、半径值。半径值为0说明没有检测到线，则发布停车信息
# Calculate self.circle, calculate X coordinate, radius value. A radius value of 0 means that no line is detected, then post a parking message
rgb_img, binary, self.circle = self.color.line_follow(rgb_img, self.hsv_range)
```

Calculating the value of the angular velocity of

```
#320是中心点的X坐标的值，通过得到的图像的X值与320的偏差，可以计算出“我现在距离中心有多远”，然后计算角速度的值
#320 is the value of the X-coordinate of the center point. By the deviation of the X-value of the obtained image from 320, we can calculate "how far am I from the center", and then calculate the value of the angular velocity.
[z_Pid, _] = self.PID_controller.update([(point_x - 320)*1.0/16, 0])
```