Autonomous driving line patrol

Note: The ROS_DOMAIN_ID of the Raspberry Pi and the microROS control board need to be consistent. You can check [MicroROS Control Board Parameter Configuration] to set the microROS control board ROS_DOMAIN_ID. Check the tutorial [Connect MicroROS Agent] to determine whether the IDs are consistent.

1、Program function description

After the program is started, adjust the pitch angle of the camera and move the camera downward so that the camera can see the line. Then click on the image window and press the r key to enter the color selection mode; then frame the line area in the picture. Select the color of the required line tracing, and the processed image will be automatically loaded after releasing the mouse; finally, press the space bar to turn on the line tracing function. When the car encounters an obstacle during operation, it will stop and the buzzer will sound.

2、Program code reference path

After entering the docker container, the location of the source code of this function is as follows.

```
/root/yahboomcar_ws/src/yahboomcar_astra/yahboomcar_astra/
```

3、Program start

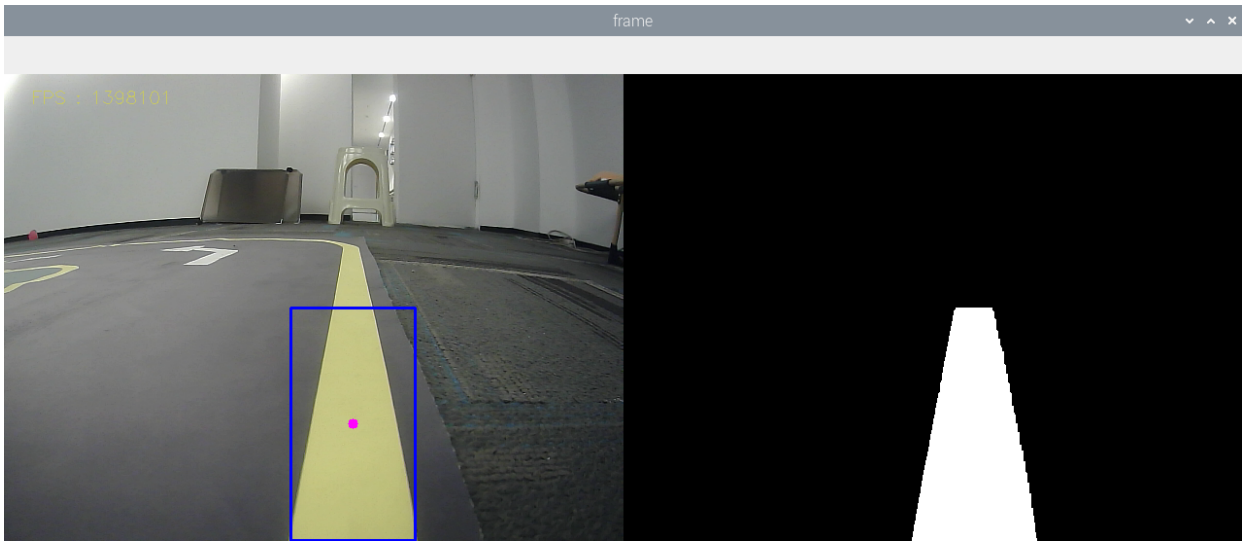
3.1、Start command

After entering the docker container, enter the following command in the terminal.

```
ros2 run yahboomcar_astra  follow_line
```

Take the yellow line patrol as an example.

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

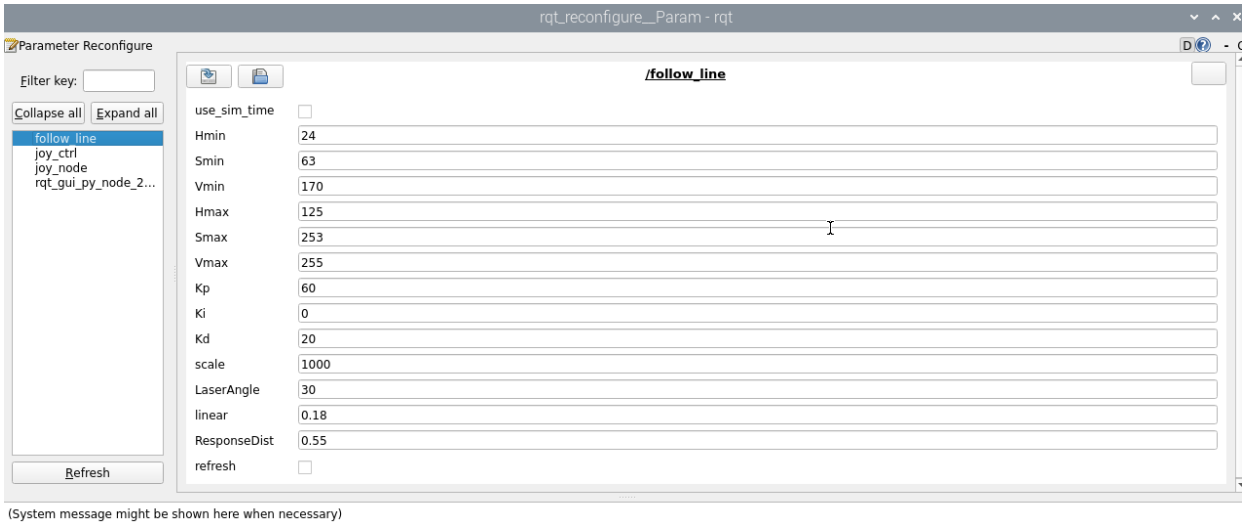


As shown in the picture above, what is shown on the right is the processed image, which will show the yellow line part. Then press the space bar to start calculating the speed, and the car will follow the line and drive automatically.

3.2、Dynamic parameter adjustment

Relevant parameters can be adjusted through the dynamic parameterizer and docker terminal input.

```
ros2 run rqt_reconfigure rqt_reconfigure
```



The adjustable parameters are as follows.

parameter	Parameter Description
Kp	P value of PID
Ki	I value of PID
Kd	D value of PID
scale	PID adjustment proportion coefficient
LaserAngle	Radar detection angle
linear	Linear speed
ResponseDist	Obstacle avoidance detection distance
refresh	Refresh parameters button

4、core code

The following is the implementation principle of line patrol

- Calculate the offset between the center coordinates of the patrol line and the center of the image.
- Calculate the value of angular velocity based on the coordinate offset.

- Release speed to drive the car.

Calculate the center coordinates,

```
#Calculate hsv value
rgb_img, self.hsv_range = self.color.Roi_hsv(rgb_img, self.Roi_init)
#Calculate self.circle and calculate the X coordinate and radius value. A radius
value of 0 indicates that no line is detected and parking information is released.
rgb_img, binary, self.circle = self.color.line_follow(rgb_img, self.hsv_range)
```

Calculate the value of angular velocity,

```
#320 is the value of the X coordinate of the center point. Through the deviation of
the X value of the obtained image from 320, we can calculate "how far away from the
center I am now", and then calculate the value of the angular velocity
[z_Pid, _] = self.PID_controller.update([(point_x - 320)*1.0/16, 0])
```