5. Astra autonomous driving

Function package: ~/yahboomcar_ws/src/yahboomcar_linefollw

5.1. Introduction

The Yahboom mobile robot's depth camera is capable of autonomous driving. It can recognize multiple colors at any time, independently store the currently recognized colors, and follow the detected and recognized colors. During the tracking process, it can also achieve real-time obstacle avoidance.

The Yahboom mobile robot can also realize the function of real-time control of HSV. By adjusting the high and low thresholds of HSV, it filters out interfering colors, so that the tracking route can be ideally identified in complex environments. If the color selection effect is not ideal, If so, you need to move the car to different environments and calibrate it at this time, so that it can recognize the colors we need in complex environments.

5.1.1, HSV introduction

HSV (Hue, Saturation, Value) is a color space created by A. R. Smith in 1978 based on the intuitive characteristics of color, also known as the Hexcone Model.

The parameters of color in this model are: hue (H), saturation (S), and lightness (V).

H: 0 — 180

S: 0 — 255

V: 0 — 255

Here some reds are classified into the purple range:

	black	grey	white	red		orange	yellow	green	light blue	blue	Purple
H_min	0	0	0	0	156	11	26	35	78	100	125
H_max	180	180	180	10	180	25	34	77	99	124	155
S_min	0	0	0	43		43	43	43	43	43	43
S_max	255	43	30	255		255	255	255	255	255	255
V_min	0	46	221	46		46	46	46	46	46	46
V_max	46	220	255	255		255	255	255	255	255	255

5.1.2, HSV hexagonal pyramid

- **Hue H**

Represents color information, that is, the position of the spectral color. This parameter is represented by an angle, with a value ranging from 0° to 360°, starting from red and counting in counterclockwise direction. Red is 0°, green is 120°, and blue is 240°. Their complementary colors are: yellow is 60°, cyan is 180°, and purple is 300°.

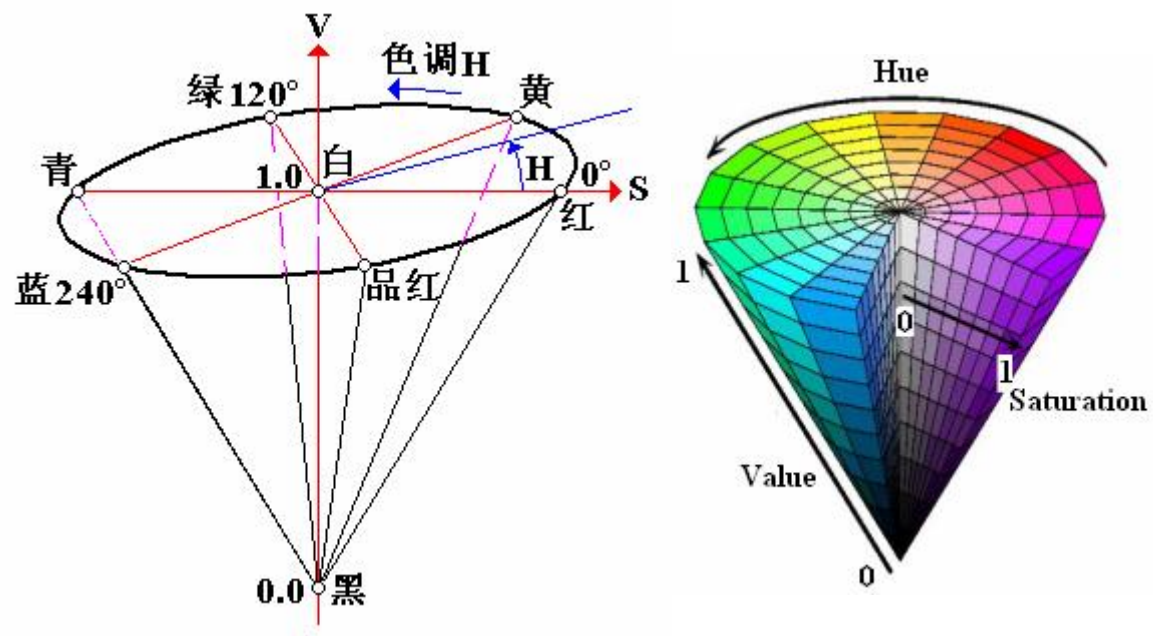
- **Saturation S**

Saturation S is expressed as the ratio between the purity of the selected color and the maximum purity of that color. When S=0, there is only grayscale. 120 degrees apart. Complimentary colors are 180 degrees apart. A color can be thought of as the result of mixing a certain spectral color with white. The greater the proportion of spectral colors, the closer the color is to spectral colors, and the higher the saturation of the color. The saturation is high and the color is deep and vivid. The white light component of the spectral color is 0, and the saturation reaches the highest level. Usually the value range is 0% ~ 100%. The larger the value, the more saturated the color.

- **LightnessV**

Brightness represents the brightness of a color. For light source color, the brightness value is related to the brightness of the luminous body; for object color, this value is related to the transmittance or reflectance of the object. Usually the value range is 0% (black) to 100% (white). One thing to note: there is no direct relationship between it and light intensity.

The three-dimensional representation of the HSV model evolves from the RGB cube. If you imagine looking from the white vertices of the RGB along the diagonal of the cube to the black vertices, you can see the hexagonal shape of the cube. The hexagonal borders represent color, the horizontal axis represents purity, and lightness is measured along the vertical axis.



5.2. Operation steps

5.2.1. Start

Note: [R2] on the remote control handle has the [pause/start] function for this gameplay. Different models will have different parameter ranges, but the principle is the same; take the X3 McLunner as an example.

There are two starting methods, choose one. The demonstration case is method 2; before starting, place the robot to the starting position so that the depth camera faces downward as much as possible.

```
#Raspberry Pi 5 master needs to enter docker first, please perform this step
#If running the script into docker fails, please refer to ROS/07, Docker tutorial
~/run_docker.sh
```

method one

robot side

```
roslaunch yahboomcar_linefollw follow_line.launch VideoSwitch:=true
img_flip:=false
```

Method 2

Can be controlled remotely for easy operation.

robot side

```
roslaunch yahboomcar_linefollw follow_line.launch VideoSwitch:=false
img_flip:=false
```

virtual machine

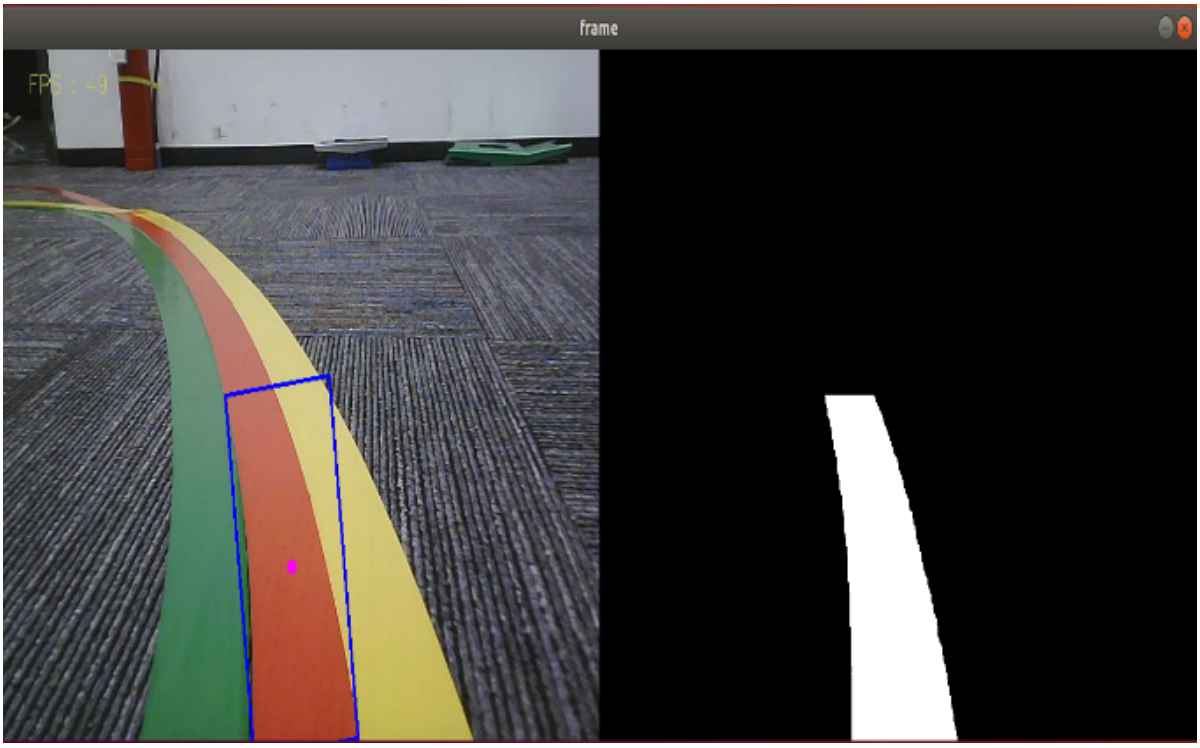
```
roslaunch yahboomcar_linefollw line.launch
```

- VideoSwitch parameter: whether to use the camera function package to start.
- img_flip parameter: whether to flip the screen horizontally, the default is false.

Set parameters according to needs, or modify the launch file directly, so there is no need to attach parameters when starting.

5.2.2. Identification

After startup, the system defaults to [Target Detection Mode], as shown below on the left:



Keyboard key control:

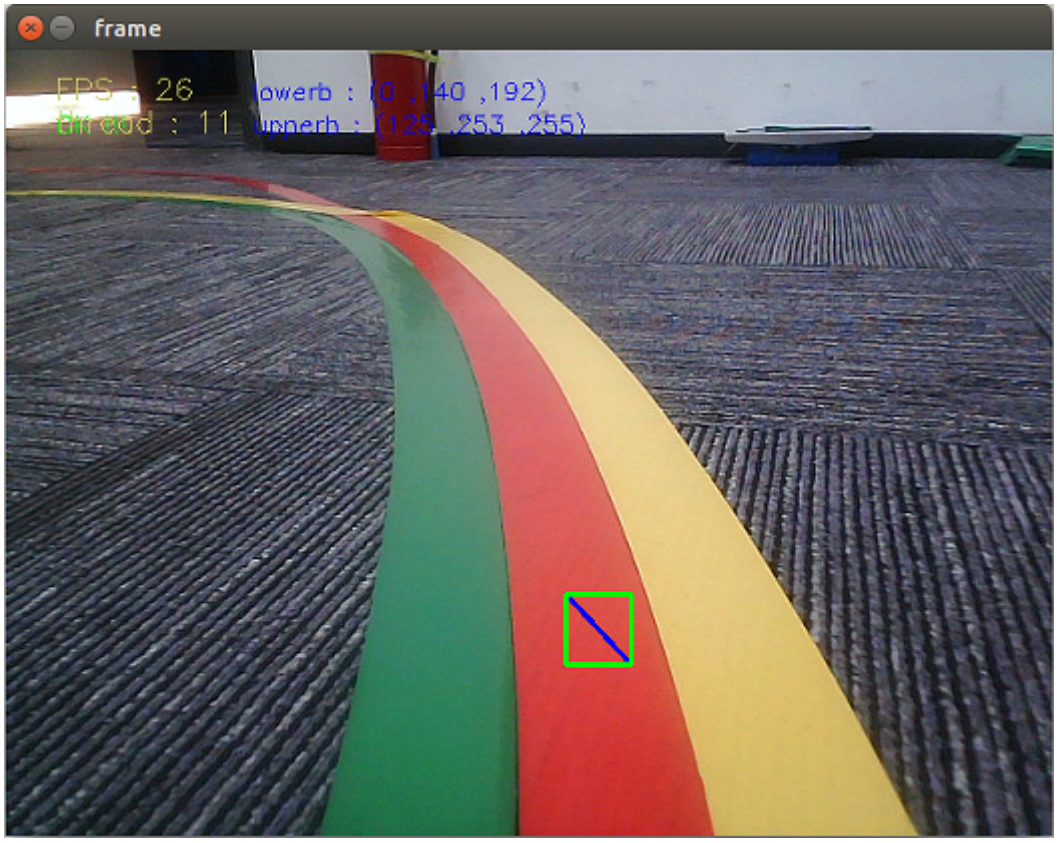
【r】: Color selection mode, you can use the mouse to select the area of the color to be recognized (cannot exceed the area range).

【i】: Target detection mode. The color image on the left (Color) and the binary image on the right (Binary).

【q】: Exit the program.

【Spacebar】: Follow the track.

In the color selection mode, you can use the mouse to select the area of the color to be recognized (cannot exceed the area range), as shown in the figure below, release it to start recognition.

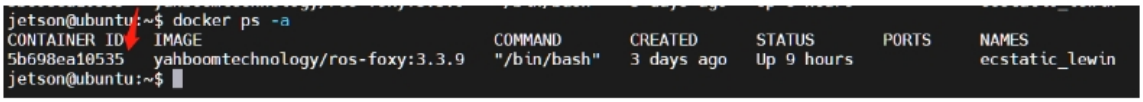


5.2.3. Color calibration

<PI5 needs to open another terminal to enter the same docker container

1. In the above steps, a docker container has been opened. You can open another terminal on the host (car) to view:

```
docker ps -a
```



2. Now enter the docker container in the newly opened terminal:

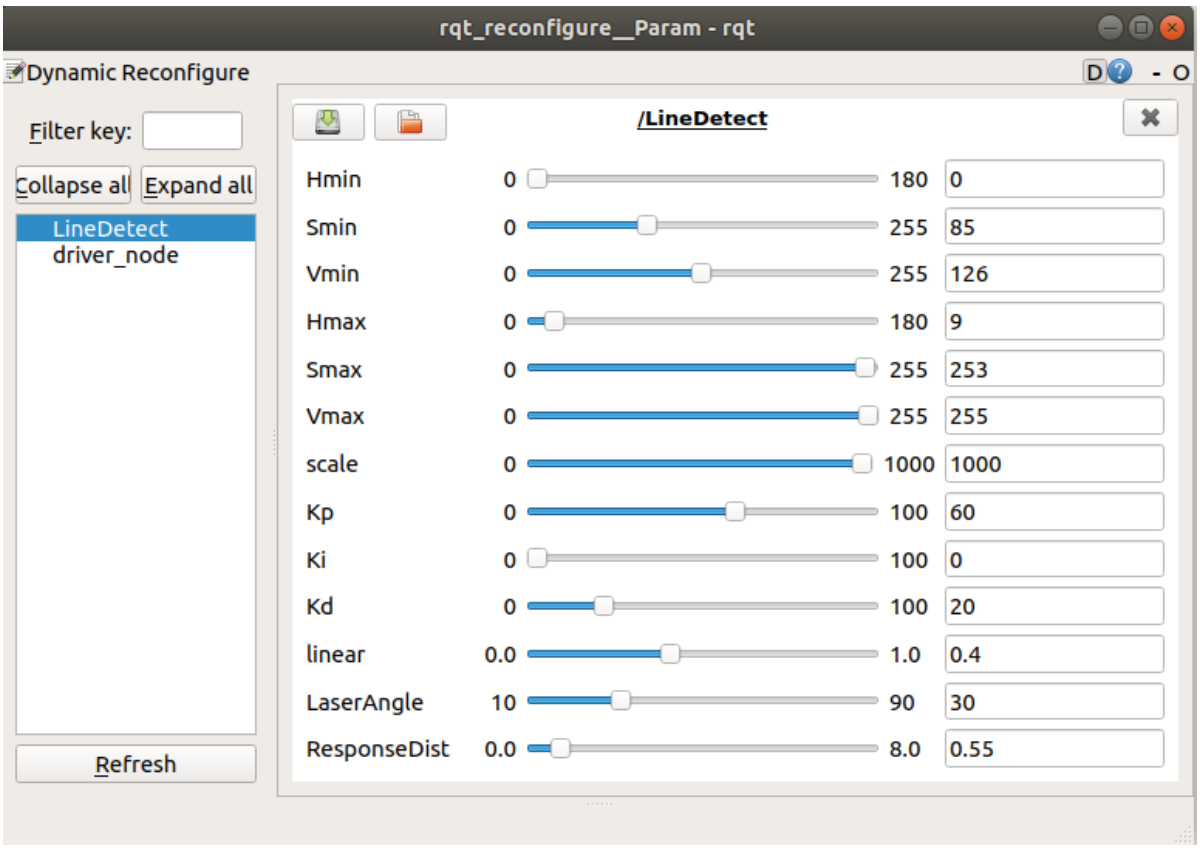
```
docker exec -it 5b698ea10535 /bin/bash
```

After successfully entering the container, you can open countless terminals to enter the container.

Dynamic parameter debugging tool

```
rosrun rqt_reconfigure rqt_reconfigure
```

Set the mode to [Target Detection Mode] and start the dynamic parameter debugging tool.



Select the [LineDetect] node. Generally, you only need to adjust [Hmin], [Smin], [Vmin], and [Hmax]. These four parameters can be well identified. The slide bar is always in a dragging state and data will not be transferred to the system until it is released; you can also select a row and then slide the mouse wheel.

Parameter analysis:

[Kp], [Ki], [Kd]: PID control during car driving.

[scale]: PID scaling.

[linear]: Car running speed; range [0, 1.0], unit: meters; set as required.

[LaserAngle]: Lidar effective angle; range [0, 180], unit: degree; set as required.

[ResponseDist]: lidar response distance; range [0.15, 8.0], unit: meters; set as required.

- Parameter modification

When the parameters are adjusted to the optimal state, the corresponding parameters are modified into the file, and no adjustment is required when using again.

According to the optimal parameters of the [rqt_reconfigure] debugging tool, enter the [scripts] folder of the [yahboomcar_linefollw] function package and modify the parameters corresponding to the [follow_line.py] file, as shown below

```
class LineDetect:
    def __init__(self):
        rospy.on_shutdown(self.cancel)
        rospy.init_node("LineDetect", anonymous=False)
        ... ...
        self.scale = 1000
        self.FollowLinePID = (60, 0, 20)
        self.linear = 0.4
        self.LaserAngle = 30
        self.ResponseDist = 0.55
        self.PID_init()
        ... ...
```

[rqt_reconfigure] Modification of the initial value of the debugging tool

```
gen.add("Hmin", int_t, 0, "Hmin in HSV", 0, 0, 180)
gen.add("Smin", int_t, 0, "Smin in HSV", 85, 0, 255)
gen.add("Vmin", int_t, 0, "Vmin in HSV", 126, 0, 255)
gen.add("Hmax", int_t, 0, "Hmax in HSV", 9, 0, 180)
gen.add("Smax", int_t, 0, "Smax in HSV", 253, 0, 255)
gen.add("Vmax", int_t, 0, "Vmax in HSV", 255, 0, 255)
gen.add("scale", int_t, 0, "scale", 1000, 0, 1000)
gen.add("Kp", int_t, 0, "Kp in PID", 60, 0, 100)
gen.add("Ki", int_t, 0, "Ki in PID", 0, 0, 100)
gen.add("Kd", int_t, 0, "Kd in PID", 20, 0, 100)
gen.add("linear", double_t, 0, "linear", 0.4, 0, 1.0)
gen.add("LaserAngle", int_t, 0, "LaserAngle", 30, 10, 90)
gen.add("ResponseDist", double_t, 0, "ResponseDist", 0.55, 0, 8)
exit(gen.generate(PACKAGE, "LineDetect", "LineDetectPID"))
```

Enter the [cfg] folder of the [yahboomcar_linefollw] function package and modify the initial values of the parameters corresponding to the [LineDetectPID.cfg] file. The color [HSV] adjustment parameters do not need to be modified. The system will automatically generate the [LineFollowHSV.text] file, which will be automatically read when the system starts.

```
gen.add("Kp", int_t, 0, "Kp in PID", 60, 0, 100)
```

Take the above article as an example to analyze

Parameters	Analysis	Corresponding parameters
name	The name of the parameter	"Kp"
type	parameter data type	int_t
level	a bitmask passed to the callback	0
description	A description parameter	"Kp in PID"
default	Initial value for node startup	60
min	parameter minimum value	0
max	parameter maximum value	10.0

Note: After modification, you must recompile and update the environment to be effective.

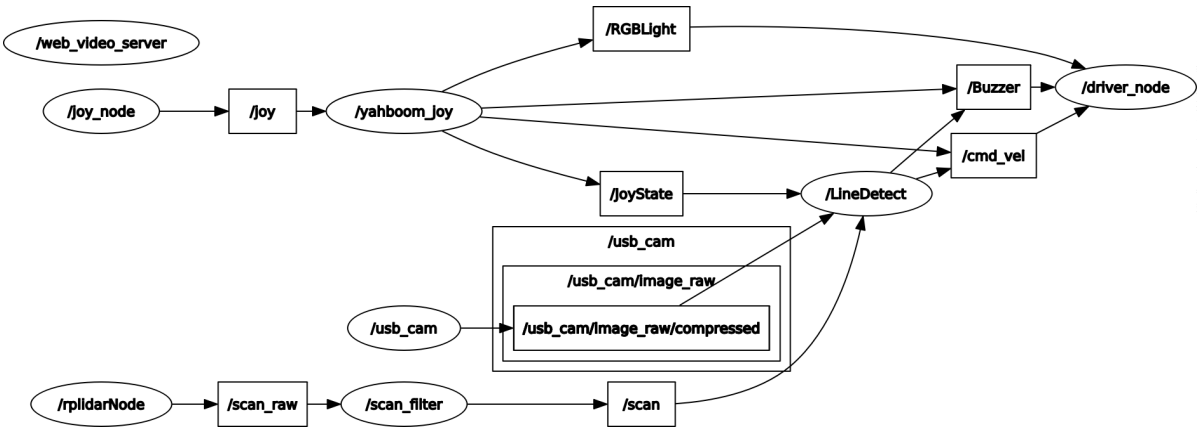
```
cd ~/yahboomcar_ws
catkin_make
source devel/setup.bash
```

5.2.4. Tracking driving

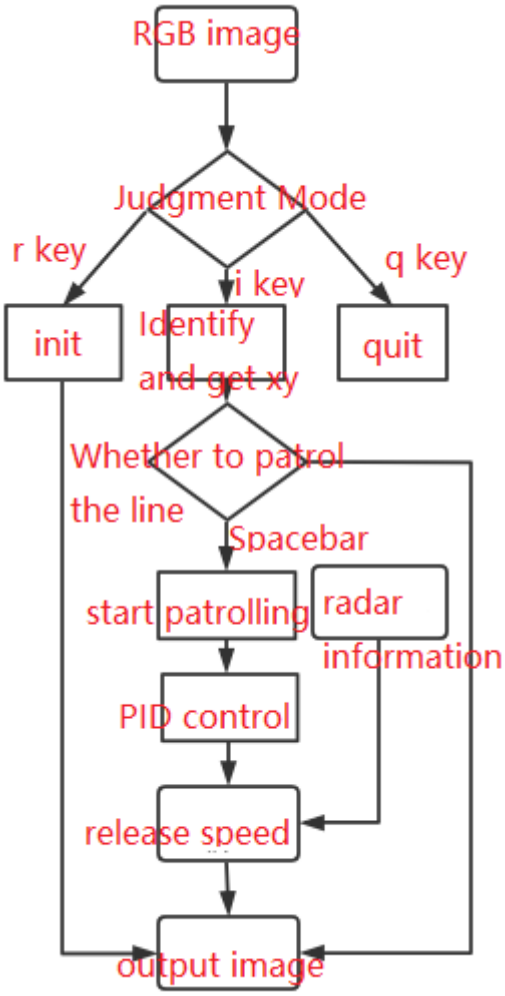
After identifying that there is no problem, click the [space bar] on the keyboard to execute the tracking program.

Node view

```
rqt_graph
```



【LineDetect】Node analysis



- Subscribe to lidar
- Subscribe to images
- Subscription handle
- Publish speed information
- Post buzzer messages