# 4. Astra autonomous driving



Function package: /home/yahboom/YBAMR-COBOT-EDU-00001/src/yahboom_navrobo_linefollw

## 4.1. Introduction

The Yahboom mobile robot uses a deep camera to autonomously drive. It can recognize multiple colors at any time and store the currently recognized color autonomously. It can follow the detected and recognized color and avoid obstacles in real time during the tracking process. If there is an obstacle, the screen will print "Obstacles ahead !!!"

The Yahboom mobile robot can also realize the function of real-time HSV control. By adjusting the high and low thresholds of HSV, the interfering colors are filtered out, so that the tracking route can be ideally recognized in complex environments. If the color picking effect is not ideal, the car needs to be moved to different environments for calibration to achieve the recognition of the color we need in complex environments.

### 4.1.1, 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 color parameters in this model are: hue (H), saturation (S), and brightness (V).

H: 0 — 180

S: 0 — 255

V: 0 — 255

Here, part of the red is classified as purple:

| | 黑 | 灰 | 白 | 红 | | 橙 | 黄 | 绿 | 青 | 蓝 | 紫 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

### 4.1.2, HSV Hexagonal Pyramid

- **Hue H**

Indicates color information, that is, the position of the spectral color. This parameter is expressed as an angle, ranging from 0° to 360°. Starting from red and counting counterclockwise, red is 0°, green is 120°, and blue is 240°. Their complementary colors are: yellow is 60°, cyan is 180°, and purple is 300°.

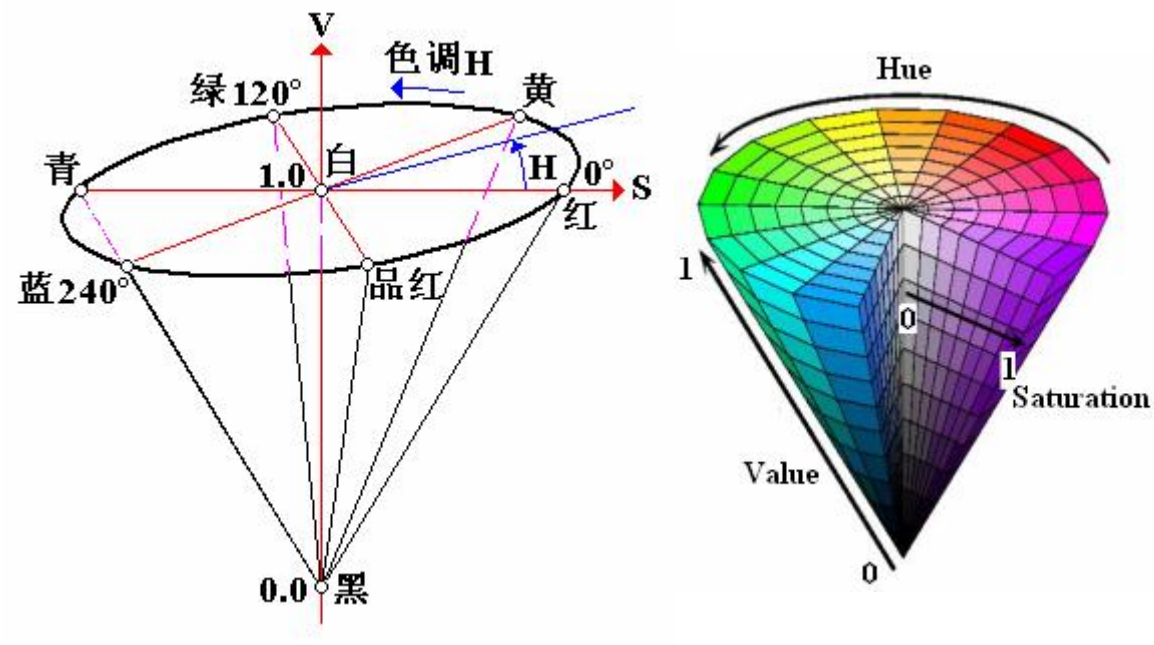
- **Saturation S**

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

- **Brightness V**

Brightness indicates how bright a color is. For light source color, the brightness value is related to the brightness of the light source; for object color, this value is related to the transmittance or reflectance of the object. The value usually ranges from 0% (black) to 100% (white). One thing to note is that there is no direct relationship between it and light intensity.

The three-dimensional representation of the HSV model evolved from the RGB cube. Imagine observing from the white vertex along the diagonal of the RGB cube to the black vertex, you can see the hexagonal shape of the cube. The hexagonal boundary represents the color, the horizontal axis represents the purity, and the brightness is measured along the vertical axis.




## 4.2, Operation steps

### 4.2.1, Start

**Note: The [SWB] mid-range of the aircraft model remote control has the [Emergency Stop] function of this gameplay**

- To start control, you need to first turn the SWB button to the upper gear (control command mode) to release the remote control

Before starting, place the robot to the starting position so that the depth camera is as downward as possible

**Please do this step before running the program**

```
sudo supervisorctl stop ChassisServer #Turn off the self-starting chassis
service

#Indoor version NAVROBO-Perform this step more
sudo supervisorctl start LaserServer #start/stop Switch radar service
/home/yahboom/YBAMR-COBOT-EDU-00001/start/OBColorViewer #Release color stream
video100
#[info][722318][Pipeline.cpp:251] Start streams done!
#[info][722318][Pipeline.cpp:234] Pipeline start done!
#[warning][722318][Pipeline.cpp:327] Wait for frame timeout, you can try to
increase the wait time! current timeout=100
```

Start

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

- The car has a close-range anti-collision function, so it is necessary to ensure that the radar is started normally. If you run `rostopic echo /scan`

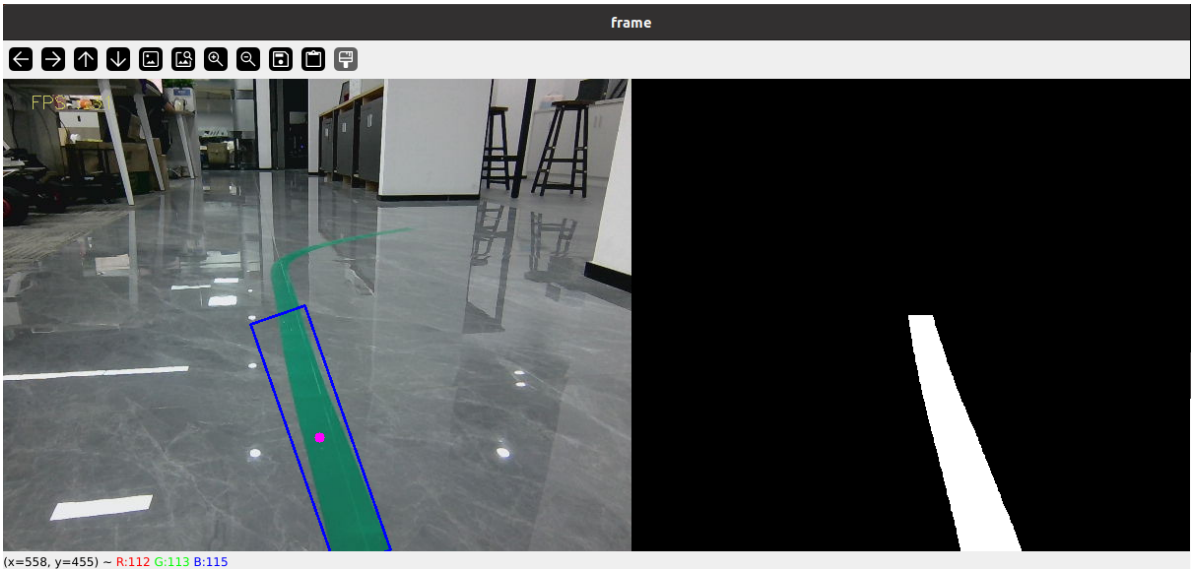
If the print is empty and the data cannot be obtained, the startup is abnormal. Please restart the radar service command.

- 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 directly modify the launch file, so that no additional parameters are required when starting.

### 4.2.2, Identification

After startup, the system defaults to [Target Detection Mode], as shown in the lower left figure:




Keyboard button control:

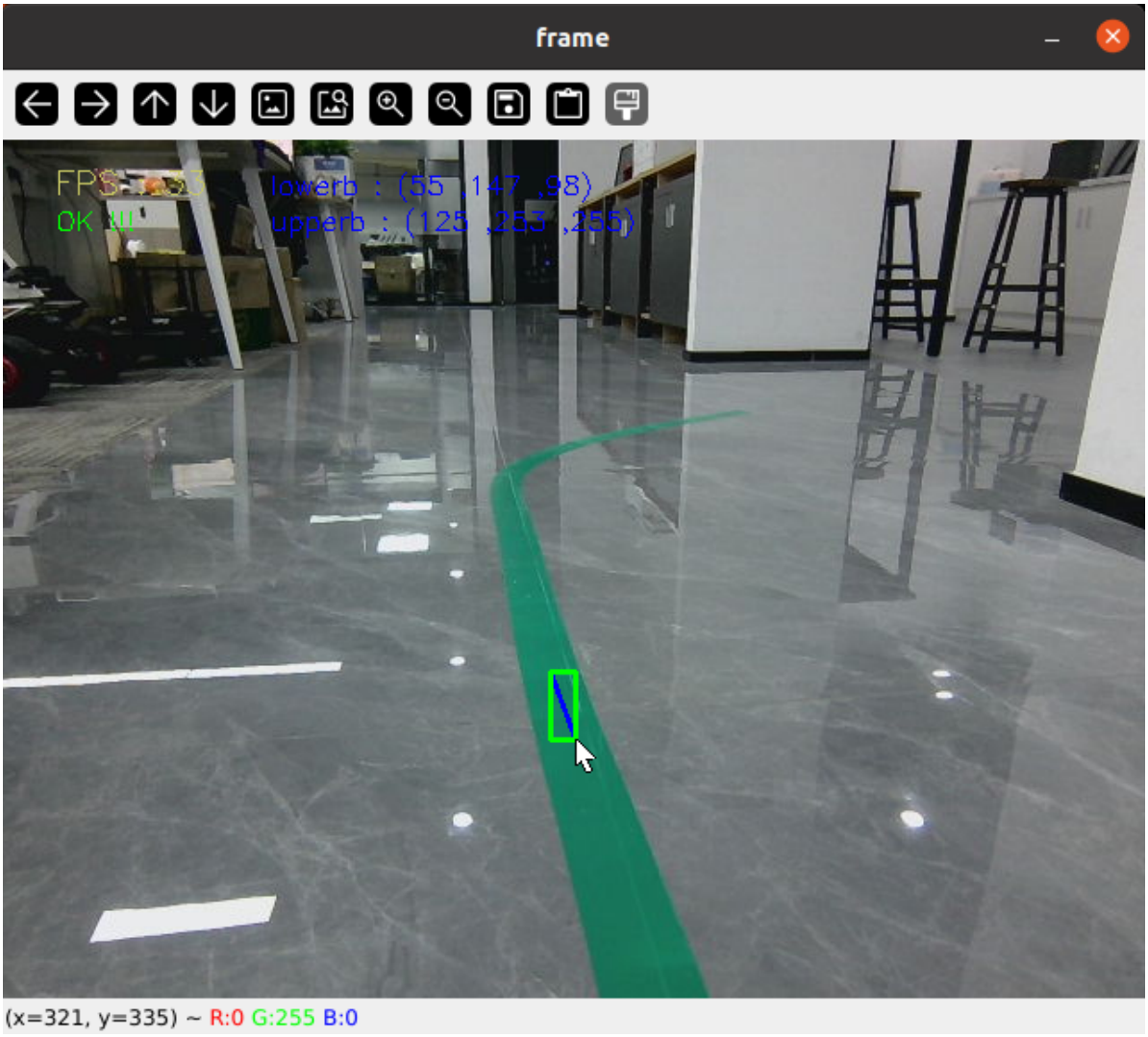
[r]: Color selection mode, you can use the mouse to select the area of the color to be identified (cannot exceed the area range).

[i]: Target detection mode. Color image on the left, binary image on the right.

[q]: Exit the program.

[Space bar]: Follow the track.

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

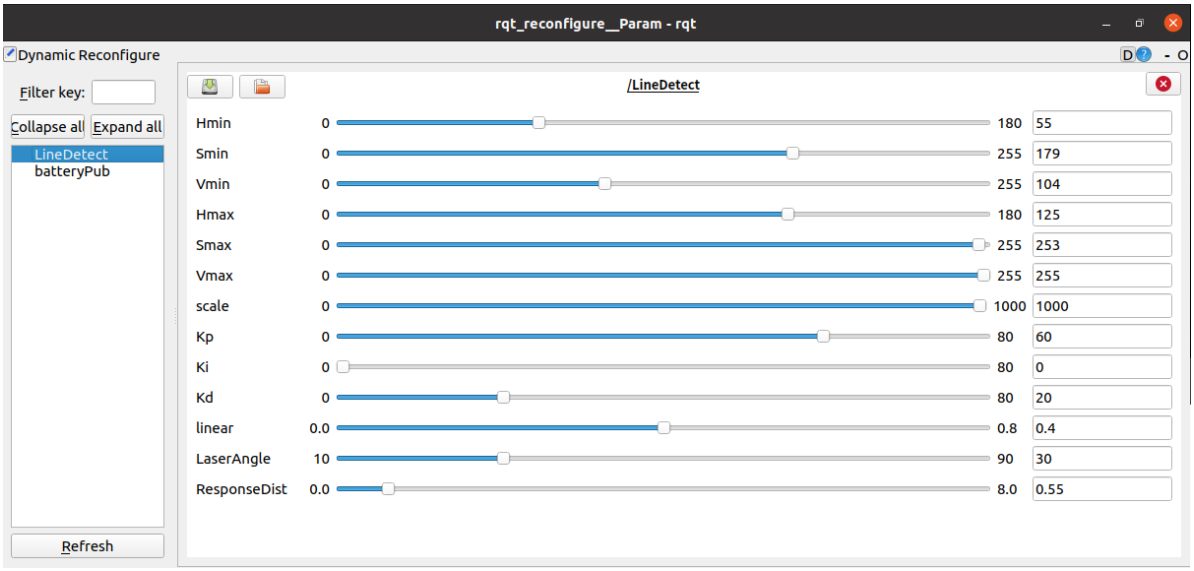



### 4.2.3, Color Calibration

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 recognized. The slider is always in the dragging state, and no data will be transferred to the system. You can release it before you can do so. You can also select a row and then slide the mouse wheel.

Parameter analysis:

[Kp], [Ki], [Kd]: PID control during the car's driving process.

[scale]: PID scaling.

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

[LaserAngle]: Laser radar effective angle; range [0, 90], unit: degree; set as required.

[ResponseDist]: Laser radar response distance; range [0.55, 5.0], unit: meter; set as required.

- Parameter modification

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

According to the optimal parameters of the [rqt_reconfigure] debugging tool, enter the [scripts] folder of the [yahboom_navrobo_linefollw] function package and modify the corresponding parameters of the [follow_line_4ROS.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] Debug tool initial value modification

```
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", in t_t, 0,
"Kp in PID", 60, 0, 80) gen.add("Ki", ••int_t, 0, "Ki in PID", 0, 0, 80)
gen.add("Kd", int_t, 0, "Kd in PID", 20, 0, 80) gen.add("linear", double_t, 0,
"linear", 0.4, 0, 0.8) 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 [yahboom_navrobo_linefollw] function package and modify the initial values of the corresponding parameters in 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, 80)
```

Analyze the above as an example

| Parameter | Analysis | Corresponding parameter |
| --- | --- | --- |
| name | Name of the parameter | "Kp" |
| type | Parameter data type | int_t |
| level | A bit mask passed to the callback | 0 |
| description | A description parameter | "Kp in PID" |
| default | Initial value of the node startup | 60 |
| min | Minimum parameter value | 0 |
| max | Maximum parameter value | 80 |

**Note: After the modification is completed, the environment must be recompiled and updated to be effective.**

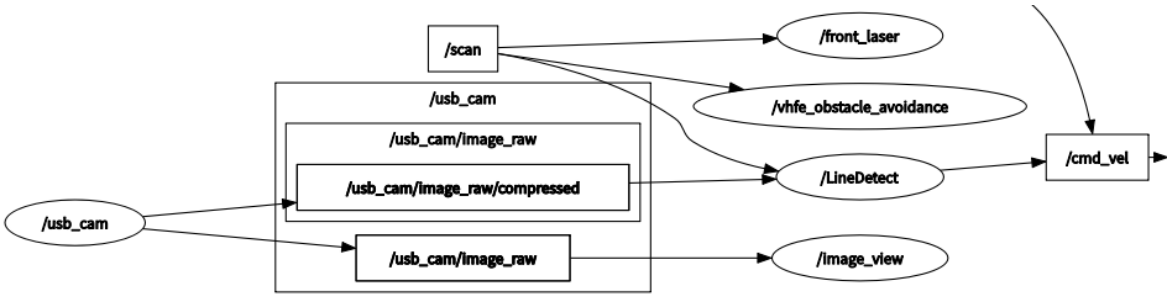
```
cd ~/YBAMR-COBOT-EDU-00001/
catkin build yahboom_navrobo_linefollw
source install/setup.bash
```

## 4.2.4, Follow the path

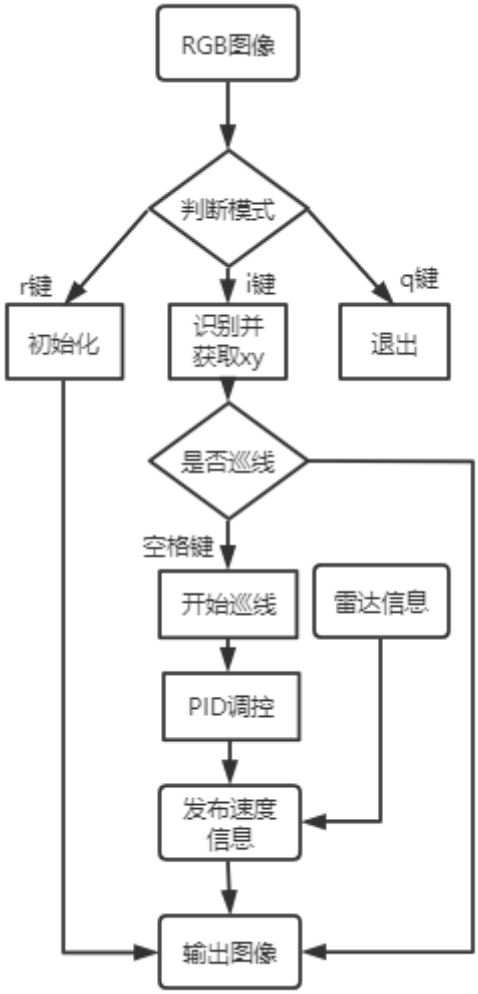
After the recognition is correct, click the keyboard [Spacebar】, execute the tracking program.

Node View

```
rqt_graph
```




【LineDetect】Node Analysis




- Subscribe to LiDAR
- Subscribe to Image
- Publish Speed Information