2. Lidar obstacle avoidance

Introduction to radar obstacle avoidance gameplay:

- Set the laser radar detection angle and response distance
- After turning on the car, the car will drive in a straight line if there are no obstacles
- Determine the position of the obstacle in the car (left front, right front, front)
- React according to the position of the obstacle in the car (turn left, turn right, turn left in a big circle, turn right in a big circle)

2.1. How to use

Note: The [SWB] mid-range of the model aircraft remote control has the [emergency stop] function of this gameplay. Due to the problem of movement, this section does not support the Ackerman model. The parameter range will change for different models, but the principle is the same. Please put the model aircraft remote control in a convenient place for control, and pay attention to safety when playing! ! !

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

Turn off the self-starting chassis service

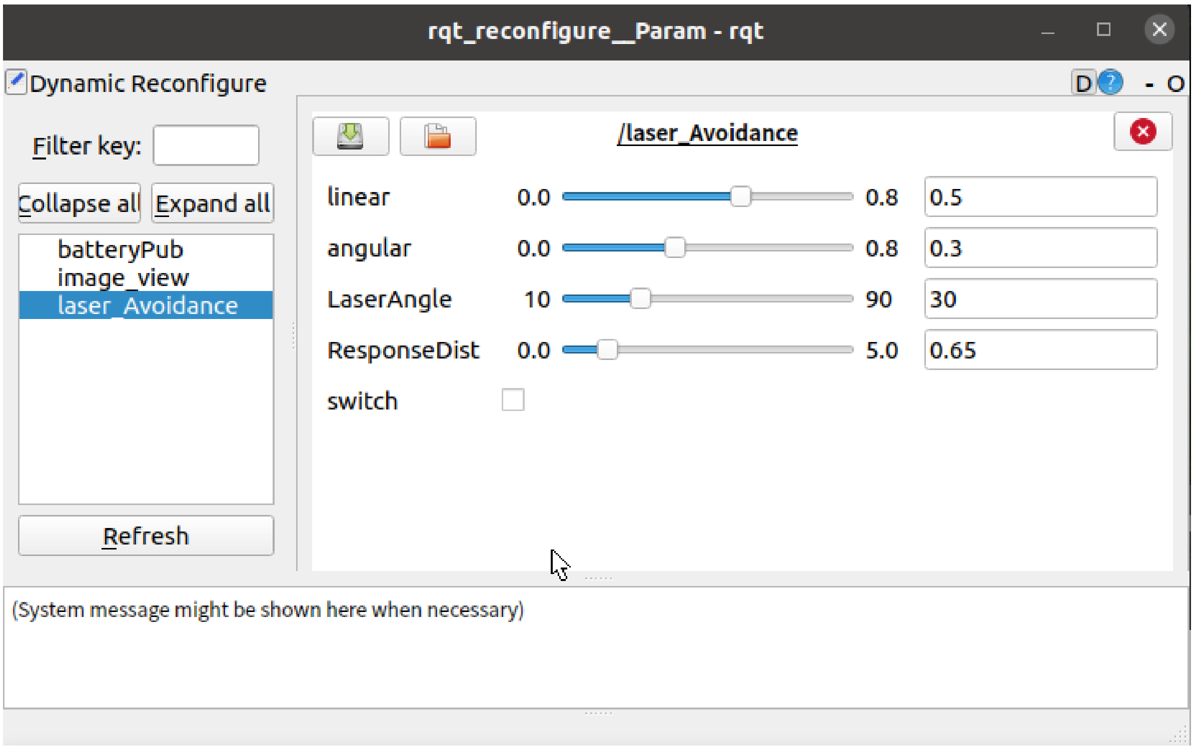
```
sudo supervisorctl stop ChassisServer
```

One-button start (robot side), after executing the command, the car starts to move.

```
sudo supervisorctl restart LaserServer #start/stop switch radar service (indoor version)
roslaunch yahboom_navrobo_laser laser_Avoidance.launch
```

Dynamic debugging parameters

```
rosrun rqt_reconfigure rqt_reconfigure
```



linear 0.0 0.8 0.5
angular 0.0 0.8 0.3
LaserAngle 10 90 30
ResponseDist 0.0 5.0 0.65
switch □

(System message might be shown here when necessary)

Parameter analysis:

Parameter	Range	Analysis
【linear】	【0.0，0.8】	Car linear speed
【angular】	【0.0，0.8】	Car angular speed
【LaserAngle】	【10，90】	Laser radar detection angle (left and right angles)
【ResponseDist】	【0.0, 5.0】	Car response distance
【switch】	【False, True】	Car movement 【Start/Pause】

The box in front of 【switch】, click the value of 【switch】 to True, the car stops. 【switch】 defaults to False, the car moves.

- 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_laser] function package and modify the corresponding parameters of the [laser_Avoidance.py] file as shown below

```
class laserAvoid:
def __init__(self):
rospy.on_shutdown(self.cancel)
... ...
self.linear = 0.5
self.angular = 0.3
self.ResponseDist = 0.80
self.LaserAngle = 30 # 10~90
```

[rqt_reconfigure] Debug Tool Initial Value Modification

```
gen.add("linear", double_t, 0, "linear in PID", 0.5, 0, 0.8)
gen.add("angular", double_t, 0, "angular in PID", 0.3, 0, 0.8)
gen.add("LaserAngle", int_t, 0, "LaserAngle", 30, 10, 90)
gen.add("ResponseDist", double_t, 0, "ResponseDist", 0.80, 0, 5)
gen.add("switch", bool_t, 0, "switch in rosbot", False)
```

Enter the [cfg] folder of the [yahboom_navrobo_laser] function package and modify the initial values of the corresponding parameters in the [laserAvoidancePID.cfg] file.

```
gen.add("linear", double_t, 0, "linear in PID", 0.5, 0, 0.8)
```

Analyze the above example

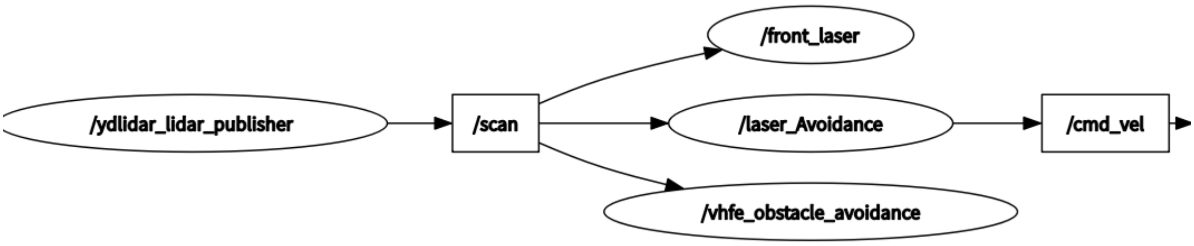
Parameter	Analysis	Corresponding parameter
name	Name of the parameter	"linear"
type	Parameter data type	double_t
level	A bit mask passed to the callback	0
description	A description parameter	"linear in PID"
default	Initial value of the node startup	0.5
min	Minimum value of the parameter	0
max	Maximum value of the parameter	0.8

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

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

Node view

```
rqt_graph
```



[laser_Aviodance] Node analysis

- Subscribe to laser radar data
- Publish car speed

2.2, source code analysis

launch file

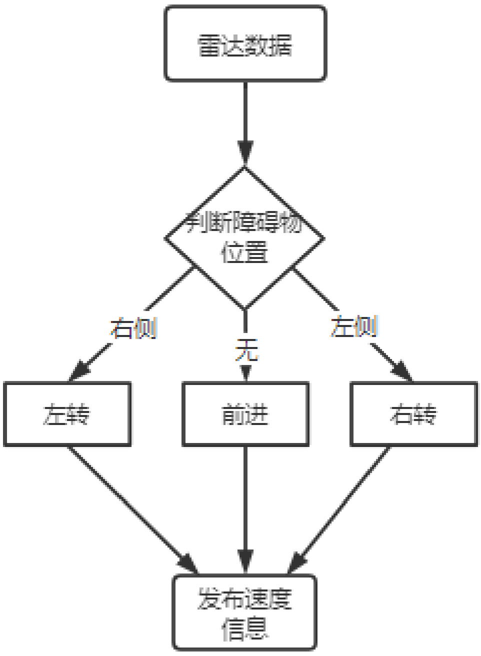
- laser_Avoidance.launch

```
<launch>
<!-- Chassis driver -->
<include file="$(find scout_bringup)/launch/scout_mini_robot_bringup.launch"/>
<!-- Start the lidar obstacle avoidance node -->
<!-- Activate lidar obstacle avoidance node -->
<node name='laser_Avoidance' pkg="yahboom_navrobo_laser"
type="laser_Avoidance.py" required="true" output="screen"/>
</launch>
```

laser_Avoidance.py source code parameter analysis:

Parameter	Default value	Judgment
self.front_warning	Default is 0	When the value is greater than 10, it means there is an obstacle in front.
self.Left_warning	Default is 0	When the value is greater than 10, it means there is an obstacle in the left front.
self.Right_warning	Default is 0	When the value is greater than 10, it means there is an obstacle in the right front.

Program design flow chart:



The car moves forward autonomously and avoids surrounding obstacles. For specific program design, see the source code of [laser_Avoidance.py].