3. Lidar guard

Introduction to radar guard gameplay:

- Set the laser radar detection angle and response distance.
- After turning on the car, the car faces the target closest to the car.
- When the distance between the target and the car is less than the response distance, the screen will keep printing obstacles!! and sound an alarm until there is no target within the response distance.
- The car angular velocity PID can be adjusted to achieve the best car rotation effect.

3.1. Usage

Note: The [SWB] mid-range of the aircraft model remote control has the [emergency stop] function of this gameplay. Please put the aircraft model remote control in a convenient place for control. 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

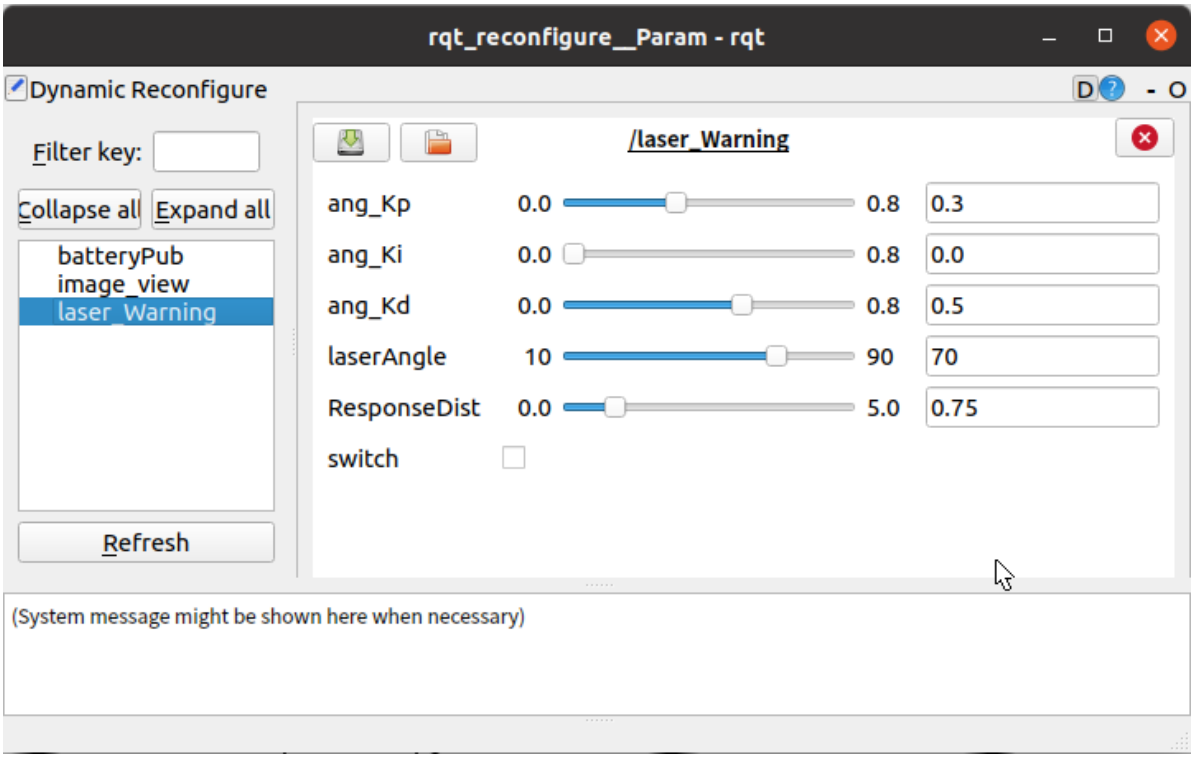
```
sudo supervisorctl stop ChassisServer
```

One-click start, 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_Warning.launch
```

Dynamic debugging parameters

```
rosrun rqt_reconfigure rqt_reconfigure
```



Parameter analysis:

Parameter	Range	Analysis
【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】

[ang_Kp], [ang_Ki], [ang_Kd]: PID debugging of the car's angular velocity.

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_Warning.py] file as shown below

```
class laserWarning:
def __init__(self):
rospy.on_shutdown(self.cancel)
... ...
self.ang_pid = SinglePID(0.3, 0.0, 0.5)
Server(laserWarningPIDConfig, self.dynamic_reconfigure_callback)
self.laserAngle = 70
self.ResponseDist = 0.75
```

[rqt_reconfigure] Debug Tool Initial Value Modification

```
gen.add("ang_Kp", double_t, 0, "Kp in PID", 0.3, 0, 0.8)
gen.add("ang_Ki", double_t, 0, "Ki in PID", 0.0, 0, 0.8)
gen.add("ang_Kd", double_t, 0, "Kd in PID", 0.5, 0, 0.8)
gen.add("laserAngle", int_t, 0, "laserAngle", 70, 10, 90)
gen.add("ResponseDist", double_t, 0, "ResponseDist", 0.75, 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 [laserWarningPID.cfg] file.

```
gen.add("ang_Kp", double_t, 0, "Kp in PID", 0.3, 0, 0.8)
```

Analyze the above example

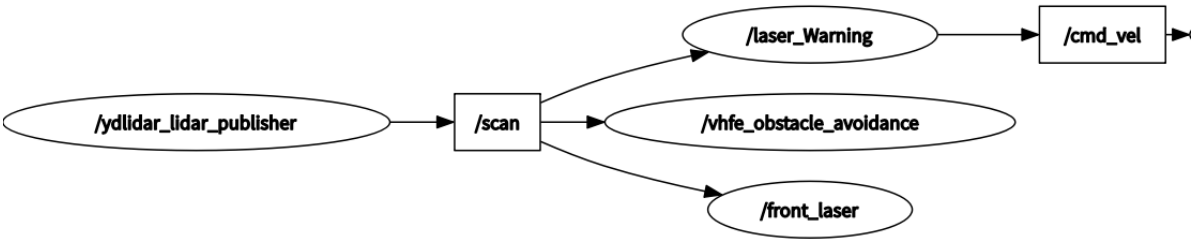
Parameter	Analysis	Corresponding parameter
name	Name of the parameter	"ang_Kp"
type	Parameter data type	double_t
level	A bit mask passed to the callback	0
description	A description parameter	"Kp in PID"
default	Initial value of the node startup	0.3
min	Minimum value of the parameter	0
max	Maximum value of the parameter	0.8

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

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

Node view

```
rqt_graph
```



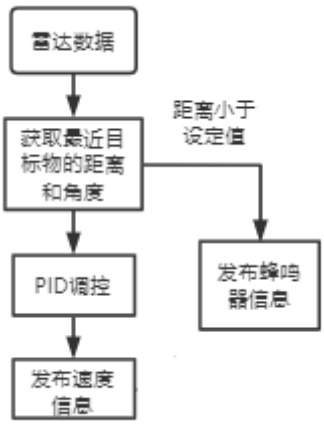
3.2, source code analysis

launch file

- laser_Warning.launch

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

laser_Warning.py source code flow chart:



According to the position where the target appears, the car rotates autonomously to face the target; when the target approaches a certain distance, the buzzer alarms.