4. Radar following

Introduction to radar following gameplay:

- Set lidar detection angle and distance.
- After turning on the car, the car will follow the target closest to the car and keep a certain distance.
- When there is an obstacle in front of the car, the buzzer keeps sounding and stops retreating until there is no obstacle.
- The PID can adjust the linear speed and angular speed of the car to make the car follow the best effect.

4.1. How to use

Note: [R2] on the remote control handle has the [pause/start] function for this gameplay. There will be slight differences depending on the model, but the principle is the same; take the X3 Mai Lun car as an example.

```
#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
```

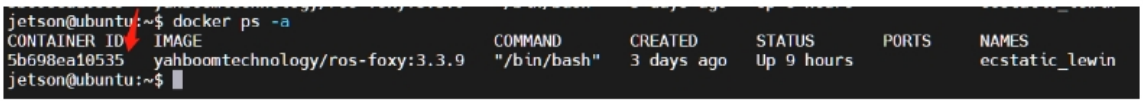
Start with one click and the car will start moving after executing the command.

```
roslaunch yahboomcar_laser laser_Tracker.launch
```

<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
```



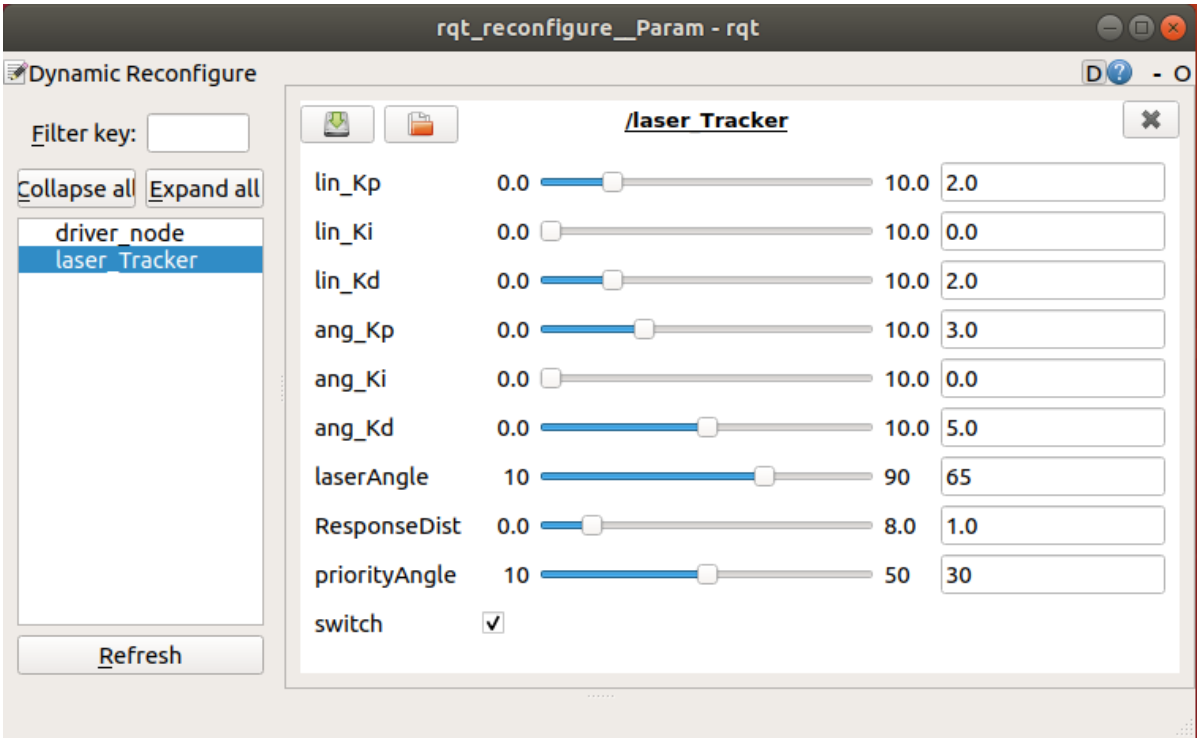
2. Now enter the docker container in the newly opened terminal:

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

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

Dynamic debugging parameters

```
rosrun rqt_reconfigure rqt_reconfigure
```



Parameter analysis:

Parameters	Range	Parse
[laserAngle]	[10, 90]	Lidar detection angle (left and right angles)
[ResponseDist]	[0.0, 8.0]	Car following distance
[priorityAngle]	[10, 50]	The car gives priority to the following range (left and right angles)
[switch]	[False, True]	The car starts to pause

[lin_Kp], [lin_Ki], [lin_Kd]: Car line speed PID debugging.

[ang_Kp], [ang_Ki], [ang_Kd]: PID debugging of car angular speed.

In the box in front of [switch], click the value of [switch] to True, and the car will stop. [switch] Default is False, the car moves.

The parameter [priorityAngle] cannot be larger than [laserAngle], otherwise it will be meaningless.

- 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_laser] function package and modify the parameters corresponding to the [laser_Tracker.py] file, as shown below

```
class laserTracker:
     def __init__(self):
         rospy.on_shutdown(self.cancel)
         ... ...
         self.lin_pid = SinglePID(2.0, 0.0, 2.0)
         self.ang_pid = SinglePID(3.0, 0.0, 5.0)
         self.ResponseDist = rospy.get_param('~targetDist', 1.0)
         Server(laserTrackerPIDConfig, self.dynamic_reconfigure_callback)
         self.laserAngle = 65
         self.priorityAngle = 30 # 40
```

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

```
gen.add("lin_Kp", double_t, 0, "Kp in PID", 2.0, 0, 10)
gen.add("lin_Ki", double_t, 0, "Ki in PID", 0.0, 0, 10)
gen.add("lin_Kd", double_t, 0, "Kd in PID", 2.0, 0, 10)
gen.add("ang_Kp", double_t, 0, "Kp in PID", 3.0, 0, 10)
gen.add("ang_Ki", double_t, 0, "Ki in PID", 0.0, 0, 10)
gen.add("ang_Kd", double_t, 0, "Kd in PID", 5.0, 0, 10)
gen.add("laserAngle", int_t, 0, "laserAngle", 65, 10, 90)
gen.add("ResponseDist", double_t, 0, "ResponseDist", 1.0, 0, 8)
gen.add("priorityAngle", int_t, 0, "priorityAngle", 30, 10, 50)
gen.add("switch", bool_t, 0, "switch in rosbot", False)
```

Enter the [cfg] folder of the [yahboomcar_laser] function package and modify the initial values of the parameters corresponding to the [laserTrackerPID.cfg] file.

```
gen.add("lin_Kp", double_t, 0, "Kp in PID", 2.0, 0, 10)
```

Take the above article as an example to analyze

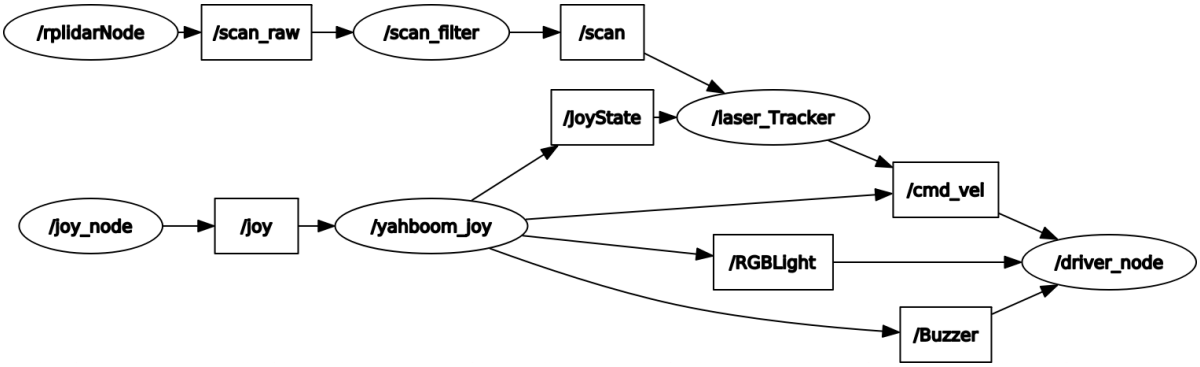
Parameters	Analysis	Corresponding parameters
name	The name of the parameter	"lin_Kp"
type	parameter data type	double_t
level	a bitmask passed to the callback	0
description	A description parameter	"Kp in PID"
default	Initial value for node startup	2.0
min	parameter minimum value	0
max	parameter maximum value	10

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

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

Node view

```
rqt_graph
```



4.2. Source code analysis

launch file

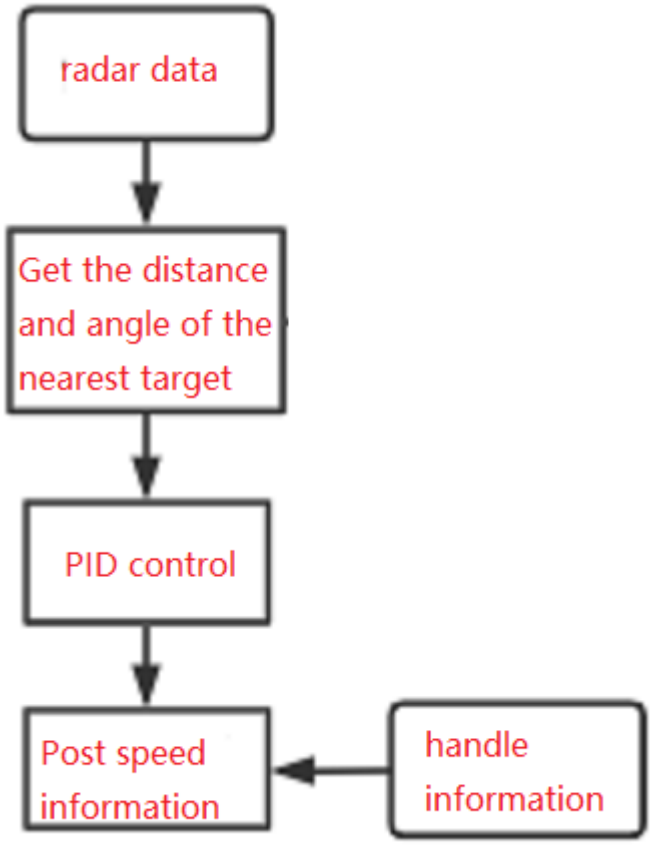
-base.launch

```
<launch>
     <!-- Start lidar node -->
     <!-- Activate the lidar node -->
     <include file="$(find rplidar_ros)/launch/rplidar.launch"/>
     <!-- Start the car chassis drive node-->
     <!-- Start the car chassis drive node -->
     <include file="$(find yahboomcar_bringup)/launch/yahboomcar.launch"/>
     <!-- Handle control node -->
     <!-- Handle control node -->
     <include file="$(find yahboomcar_ctrl)/launch/yahboom_joy.launch"/>
</launch>
```

- laser_Avoidance.launch

```
<launch>
     <!-- Launch base.launch file -->
     <!-- Launch the base.launch file -->
     <include file="$(find yahboomcar_laser)/launch/base.launch"/>
     <!-- Start lidar following node -->
     <!-- Activate lidar follow node -->
     <node name='laser_Tracker' pkg="yahboomcar_laser" type="laser_Tracker.py"
required="true" output="screen"/>
</launch>
```

laser_Tracker.py source code flow chart:



According to the location where the target appears, the car moves autonomously to face the target and always maintains a certain distance.