Lidar guard

Note: The virtual machine needs to be in the same LAN as the car, and the ROS_DOMAIN_ID needs to be consistent. You can check [Must read before use] to set the IP and ROS_DOMAIN_ID on the board.

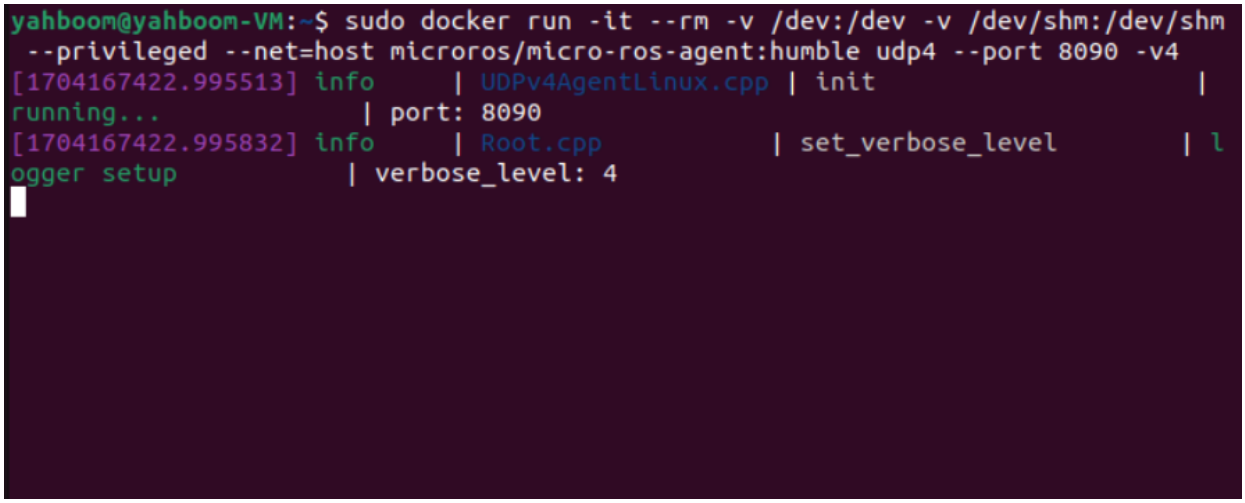
1. Program function description

The car connects to the agent and runs the program. The radar on the car scans the nearest object within the set range and tracks the object through rotation. If the object is closer to the radar than the set distance, the buzzer on the car will Sounds as a warning. Parameters such as the radar detection range and obstacle avoidance detection distance can be adjusted through the dynamic parameter adjuster.

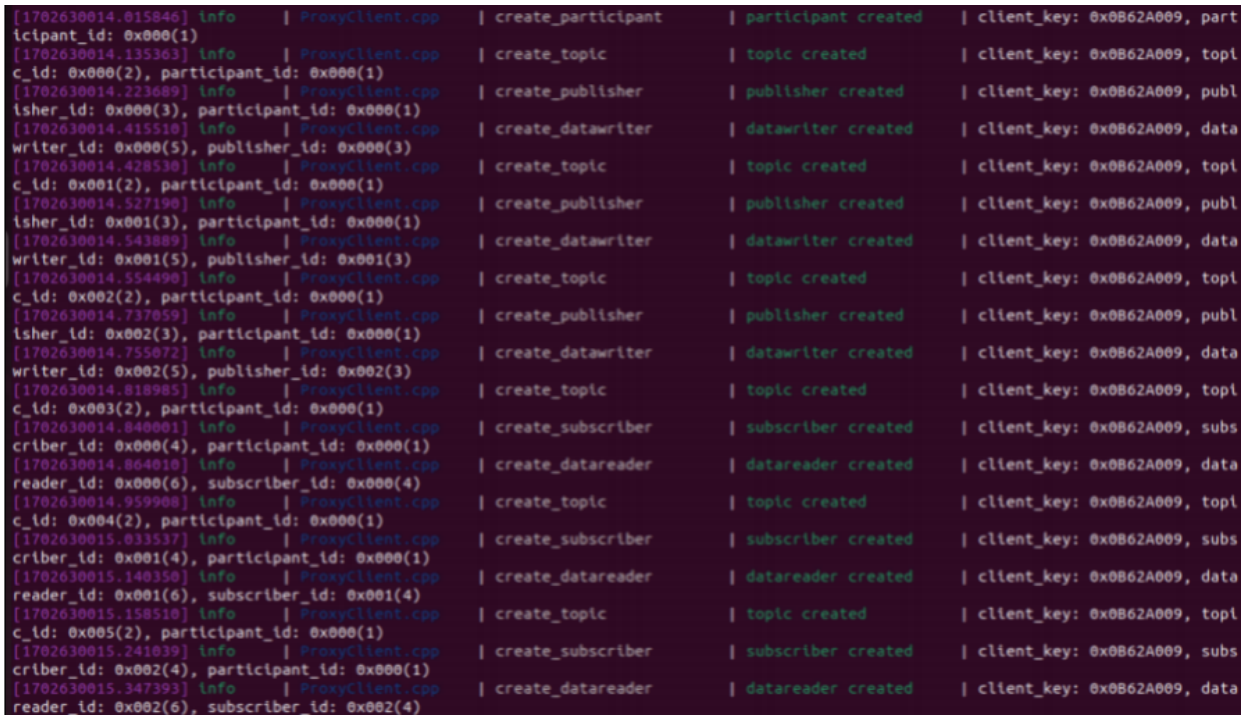
2. Start and connect to the agent

Taking the supporting virtual machine as an example, enter the following command to start the agent:

```
sudo docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --privileged --net=host
microros/micro-ros-agent:humble udp4 --port 8090 -v4
```



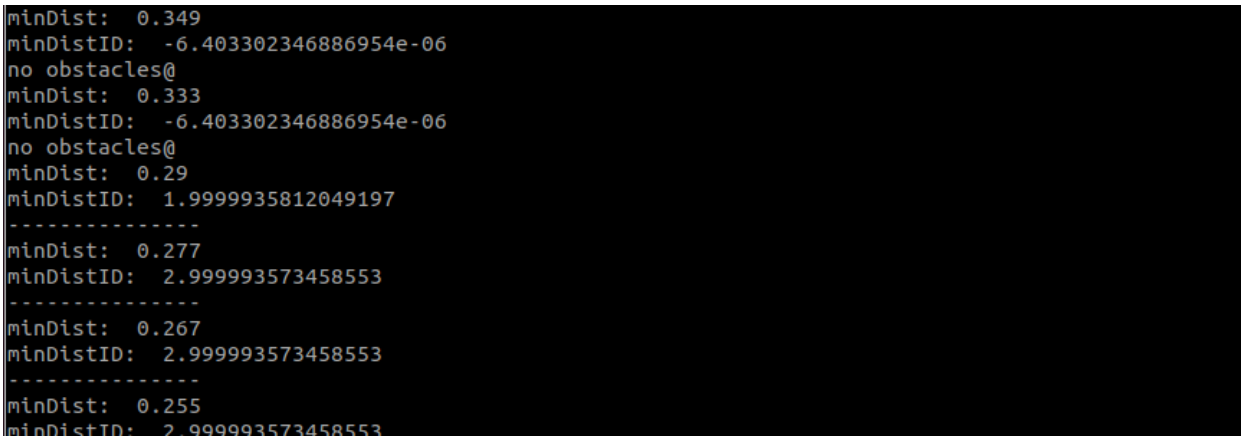
Then, turn on the car switch and wait for the car to connect to the agent. The connection is successful, as shown in the figure below.



3. Start the program

Enter the following command in the terminal to start:

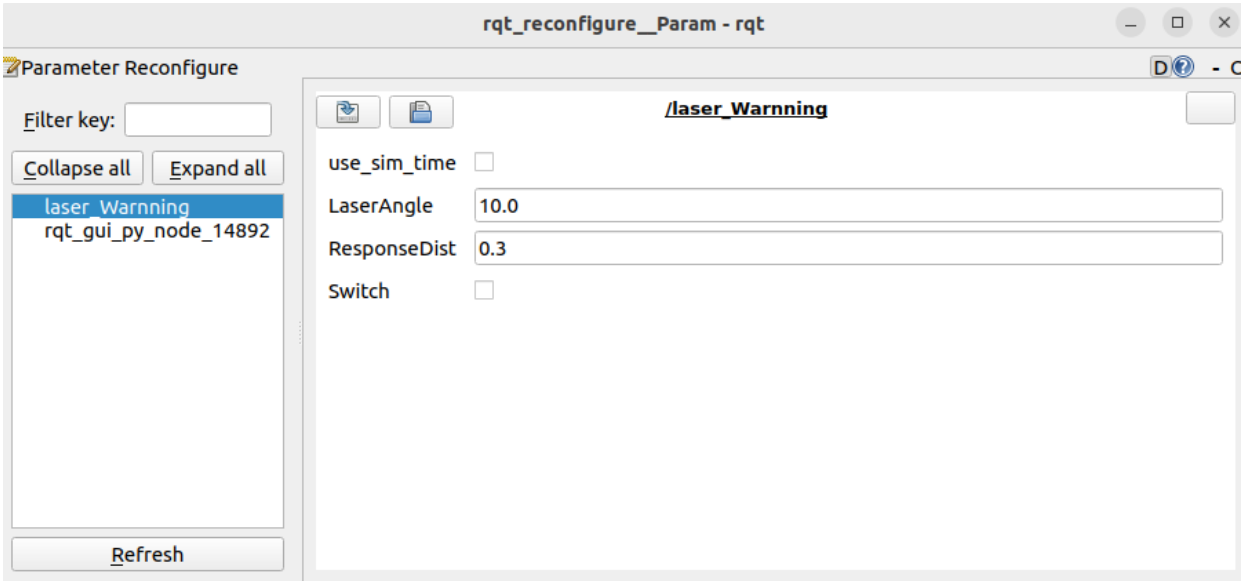
```
ros2 run yahboomcar_laser laser_Warning
```



After the program is started, it will search for the nearest object within the radar scanning range, move the object slowly, and the car will follow the object through rotation.

As shown in the picture above, if the set range is not exceeded, [no obstacles@] will be printed. If an obstacle appears, [----------] will be printed and the buzzer will sound. Some parameters can be set through the dynamic parameter adjuster and terminal input.

```
ros2 run rqt_reconfigure rqt_reconfigure
```



Note: There may not be the above nodes when you first open it. You can see all nodes after clicking Refresh. The displayed laser_Warning is the node of the radar guard.

Description of the above parameters:

- LaserAngle：Radar detection angle
- ResponseDist：tracking distance
- Switch：Game switch

After modifying the above parameters, you need to click on the blank space to transfer the parameters into the program.

4. Code analysis

Source code reference path (taking the supporting virtual machine as an example):

```
/home/yahboom/yahboomcar_ws/src/yahboomcar_laser/yahboomcar_laser
```

laser_Warning.py, the core code is as follows,

```
#Create a radar subscriber to subscribe to radar data and remote control data and a
speed publisher to publish speed data and a buzzer publisher to publish buzzer
control data.
self.sub_laser = self.create_subscription(LaserScan,"/scan",self.registerScan,1)
self.sub_JoyState = self.create_subscription(Bool,'/JoyState',
self.JoyStateCallback,1)
self.pub_vel = self.create_publisher(Twist,'/cmd_vel',1)
self.pub_Buzzer = self.create_publisher(UInt16,'/beep',1)
#Radar callback function: processes subscribed radar data
ranges = np.array(scan_data.ranges)
minDistList = []
minDistIDList = []
for i in range(len(ranges)):
    angle = (scan_data.angle_min + scan_data.angle_increment * i) * RAD2DEG
```

```
#Find the nearest object within the radar detection range according to the set radar
detection angle
if angle > 180: angle = angle - 360
if  abs(angle) < self.LaserAngle and ranges[i] > 0:
    minDistList.append(ranges[i])
    minDistIDList.append(angle)
if len(minDistList) != 0:
    minDist = min(minDistList)
    minDistID = minDistIDList[minDistList.index(minDist)]
else:
    return
#Based on the position deviation from the tracking object, the angular velocity is
calculated so that the front of the car is aligned with the object.
angle_pid_compute = self.ang_pid.pid_compute(minDistID/48, 0)
if abs(angle_pid_compute) < 0.1:
    velocity.angular.z = 0.0
else:
    velocity.angular.z = angle_pid_compute
self.pub_vel.publish(velocity)
#Determine whether the radar detection distance of the smallest object is less than
the set range, and then make a judgment whether the buzzer needs to sound.
if minDist <= self.ResponseDist:
    print("---------------")
    b = UInt16()
    b.data = 1
    self.pub_Buzzer.publish(b)

else:
    print("no obstacles@")
    b = UInt16()
    b.data = 0
    self.pub_Buzzer.publish(UInt16())
```