5、Lidar follow

Note: This course takes Rosmaster-X3 as an example. Users need to modify according to their own motion model. The course only explains the implementation method.
Function package path:~/oradar_ ws/src/yahboomcar_ laser

Introduction to radar following game:

- Set the lidar detection angle and distance.
- After the car is started, the car follows the nearest target and keeps a certain distance.
- The PID that can adjust the linear speed and angular speed of the car makes the car follow the best effect.

5.1、Usage method

Input following command in terminal.

```
roslaunch yahboomcar_laser laser_Tracker.launch
```

Input following command in terminal to dynamic debugging parameters

```
rosrun rqt_reconfigure rqt_reconfigure
```

View node picture.

```
rqt_graph
```

5.2、Source code analysis

5.2.1、launch file

laser_Tracker.launch

```
<launch>
<!-- Start base.launch file -->
<include file="$(find transbot_laser)/launch/base.launch">
</include>
<!-- Start the laser lidar avoiding -->
<node name='laser_Avoidance' pkg="transbot_laser" type="laser_Tracker.py"
required="true" output="screen"/>
</launch>
```

The base.launch file is used to start the car chassis and radar, mainly to check the laser_ Tracker.py,

Code path: ~/oradar_ws/src/yahboomcar_laser/scripts

The core code is as follows, which is mainly used to enter the callback function after receiving the radar information.

```
def registerScan(self, scan_data):
        if not isinstance(scan_data, LaserScan): return
        # 记录激光扫描并发布最近物体的位置（或指向某点）
        # Record the laser scan and publish the position of the nearest object
(or point to a point)
        ranges = np.array(scan_data.ranges)
        offset = 0.5
        frontDistList = []
        frontDistIDList = []
        minDistList = []
        minDistIDList = []
        # 按距离排序以检查从较近的点到较远的点是否是真实的东西
        # if we already have a last scan to compare to:
        for i in range(len(ranges)):
            angle = (scan_data.angle_min + scan_data.angle_increment * i) *
RAD2DEG
            # if angle > 90: print "i: {},angle: {},dist: {}".format(i, angle,
scan_data.ranges[i])
            # 通过清除不需要的扇区的数据来保留有效的数据
            if 270-self.priorityAngle<angle<270self.priorityAngle:
                if ranges[i] < (self.ResponseDist + offset):
                    frontDistList.append(ranges[i])
                    frontDistIDList.append(angle)
            elif 270-self.laserAngle < angle < 270-self.priorityAngle:
                minDistList.append(ranges[i])
                minDistIDList.append(angle)
            elif 270+self.priorityAngle<angle<270+self.laserAngle:
                minDistList.append(ranges[i])
                minDistIDList.append(angle)
        # 找到最小距离和最小距离对应的ID
        # Find the minimum distance and the ID corresponding to the minimum
distance
        if len(frontDistIDList) != 0:
            minDist = min(frontDistList)
            minDistID = frontDistIDList[frontDistList.index(minDist)]
        else:
            minDist = min(minDistList)
            minDistID = minDistIDList[minDistList.index(minDist)]
        # rospy.loginfo('minDist: {}, minDistID: {}'.format(minDist, minDistID))
        if self.ros_ctrl.Joy_active or self.switch == True:
            if self.Moving == True:
                self.ros_ctrl.pub_vel.publish(Twist())
                self.Moving = not self.Moving
            return
        self.Moving = True
        velocity = Twist()
        if abs(minDist - self.ResponseDist) < 0.1: minDist = self.ResponseDist
        velocity.linear.x = -self.lin_pid.pid_compute(self.ResponseDist,
minDist)
        ang_pid_compute = self.ang_pid.pid_compute((180 - abs(minDistID)) / 72,
0)
        if minDistID > 0: velocity.angular.z = -ang_pid_compute
        else: velocity.angular.z = ang_pid_compute
        if ang_pid_compute < 0.02: velocity.angular.z = 0
        self.ros_ctrl.pub_vel.publish(velocity)
```

Among them, self.priorityAngle is the angle of priority to follow. This value cannot be larger than self.laserAngle. Otherwise, it is meaningless. After calculating the minimum ID, the speed data is released to the bottom through PID calculation, so that the car follows the nearest object. The angle judgment here also needs to be modified according to the position of the 0 ° angle of the lidar.