3.Lidar avoiding

Note: Section 2-10 takes the transbot crawler as an example. Users need to modify it according to their own motion model. These courses are only used as running demos.

ROS package path：~/ydlidar_ws/src/transbot_laser

Introduction of lidar obstacle avoidance:

- Set lidar detection angle and response distance
- After turning on the robot, the trolley drives in a straight line without obstacles
- Based on the robot, determine the direction of the obstacle (front left, front right, straight ahead)
- Let the robot react according to the position of the obstacle (turn left, turn right, turn left for long time, turn right for long time)

3.1 Instructions

Note: The [R2] of the handle remote controller can [Pause/Open] for all functions of robot car

Start up，terminal input,

```
roslaunch transbot_laser laser_Avoidance.launch
```

Dynamic debugging parameters，terminal input,

```
rosrun rqt_reconfigure rqt_reconfigure
```

Parameter analysis:

Parameter	** Range**	Analysis
【linear】	【0.0，0.45】	Linear speed of robot
【angular】	【0.0，2.0】	Angular speed of robot
【LaserAngle】	【10，85】	Lidar detection angle (angle of left and right side)
【ResponseDist】	【0.0，8.0】	Robot response distance
【switch】	【False，True】	Robot movement【start/pause】

[Switch] Click the box in front of [switch], the value of [switch] is True, and the car will stop.
[Switch] The default is False, and the car moves.

View node

```
rqt_graph
```

3.2、Source code analysis

3.2.1、launch file

- laser_Avoidance.launch

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

- base.launch

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

- py code：~/ydlidar_ws/src/transbot_laser/scripts/laser_Avoidance.py

  The core code part, this part is mainly to judge whether there are real things on the left, front and right

```
def registerScan(self, scan_data):
        if not isinstance(scan_data, LaserScan): return
        # Record the laser scan and publish the position of the nearest object
(or point to a point)
        ranges = np.array(scan_data.ranges)
        self.Right_warning = 0
        self.Left_warning = 0
        self.front_warning = 0
        # 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 -10 > angle > -10-self.LaserAngle:
                if ranges[i] < self.ResponseDist:
                    self.Right_warning += 1
                    #print(angle)
            if 10+self.LaserAngle > angle > 10:
                if ranges[i] < self.ResponseDist:
                    self.Left_warning += 1
                    #print(angle)
            if abs(angle) < 10:
                if ranges[i] <= self.ResponseDist:
                    self.front_warning += 1
                    #print(angle)
```