2、雷达避障

功能包路径：~/rplidar_ws/src/transbot_laser

雷达避障玩法介绍：

- 设定激光雷达检测角度和响应距离
- 开启小车后，没有障碍物的情况下，小车直线行驶
- 判断障碍物显现在小车的方位（左前方、右前方、正前方）
- 根据障碍物出现在小车的方位，作出反应（左转、右转、左转大圈、右转大圈）

2.1、使用方法

注意：遥控手柄的【R2】具备所有玩法的【暂停/开启】的功能。

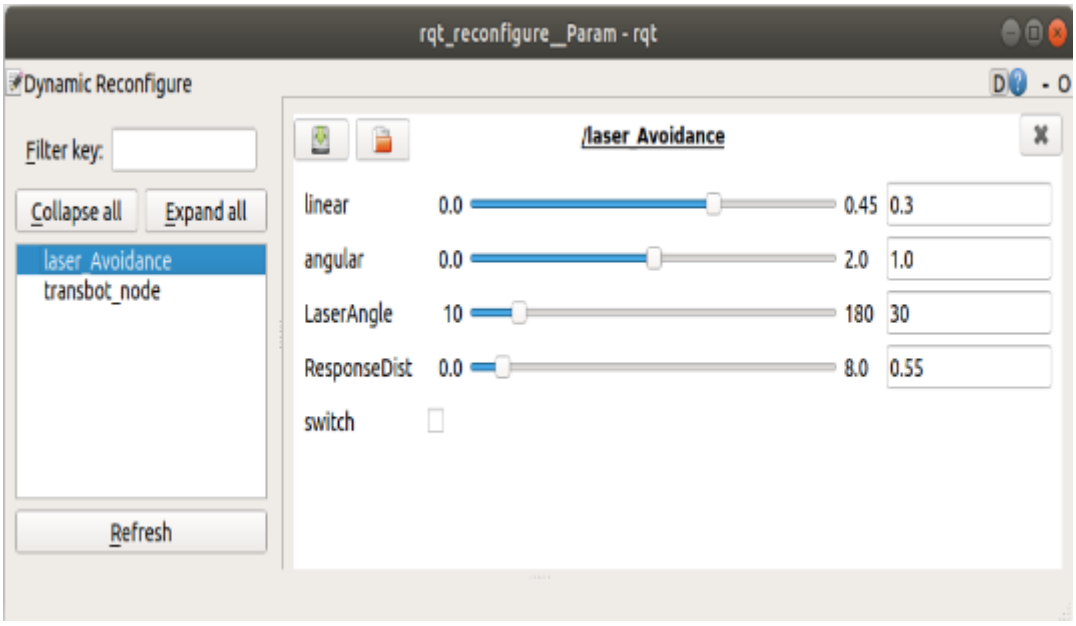
一键启动

```
roslaunch transbot_laser laser_Avoidance.launch lidar_type:=a1
```

lidar_type参数：使用激光雷达的型号：[a1,a2,a3,s1,s2]。

动态调试参数

```
rosrun rqt_reconfigure rqt_reconfigure
```



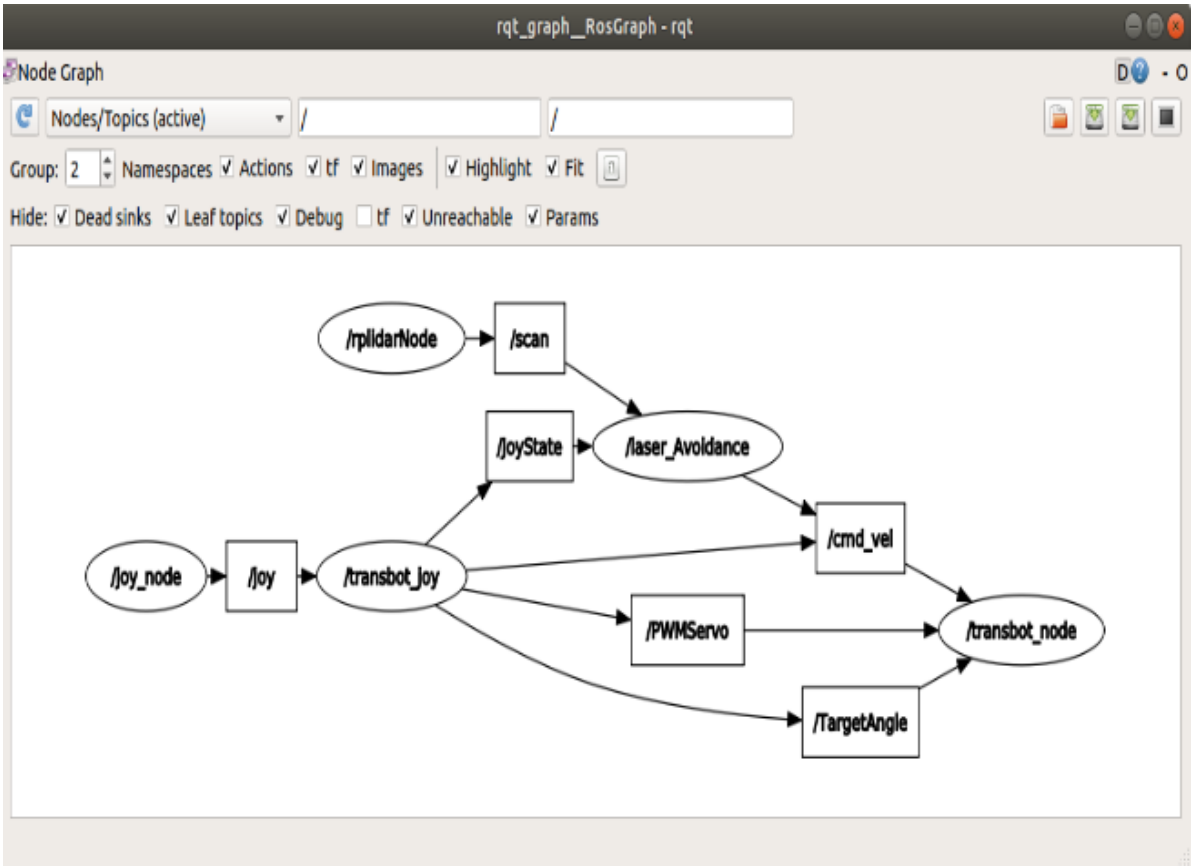
参数解析：

参数	范围	解析
【linear】	【0.0，0.45】	小车线速度
【angular】	【0.0，2.0】	小车角速度
【LaserAngle】	【10，180】	激光雷达检测角度（左右一侧角度）
【ResponseDist】	【0.0，8.0】	小车响应距离
【switch】	【False，True】	小车运动【开始/暂停】

【switch】前面的方框，点击【switch】的值为True，小车停止。【switch】默认为False，小车运动。

节点查看

```
rqt_graph
```



2.2、源码解析

launch文件

- base.launch

```
<launch>
    <!-- 启动激光雷达节点-->
    <arg name="lidar_type" default="a1" doc="lidar_type type [a1,a2,a3,s1,s2]"/>
    <include file="$(find rplidar_ros)/launch/rplidar.launch">
        <arg name="lidar_type" value="$(arg lidar_type)"/>
    </include>
    <!-- 动态调试工具节点-->
```

```
<!--    <node pkg="rqt_reconfigure" type="rqt_reconfigure" name="rqt_reconfigure"
output="screen"/>-->
    <!-- 启动小车底盘驱动节点-->
    <node pkg="transbot_bringup" type="transbot_driver.py" name="transbot_node"
required="true" output="screen">
        <param name="imu" value="/transbot/imu"/>
        <param name="vel" value="/transbot/get_vel"/>
    </node>
    <!-- 手柄控制节点 -->
    <include file="$(find transbot_ctrl)/launch/transbot_joy.launch"/>
</launch>
```

- laser_Avoidance.launch

```
<launch>
    <!-- 启动base.launch文件 -->
    <arg name="lidar_type" default="a1" doc="lidar_type type [a1,a2,a3,s1,s2]"/>
    <include file="$(find transbot_laser)/launch/base.launch">
        <arg name="lidar_type" value="$(arg lidar_type)"/>
    </include>
    <!-- 启动激光雷达避障节点 -->
    <node name='laser_Avoidance' pkg="transbot_laser" type="laser_Avoidance.py"
required="true" output="screen"/>
</launch>
```

py源码：~/rplidar_ws/src/transbot_laser/scripts/laser_Avoidance.py

```
        if self.front_warning > 10 and self.Left_warning > 10 and
self.Right_warning > 10:
            # print ('1、右转')
            ... ...
        elif self.front_warning > 10 and self.Left_warning <= 10 and
self.Right_warning > 10:
            # print ('2、左转')
            ... ...
            if self.Left_warning > 10 and self.Right_warning <= 10:
                # print ('3、右转')
                ... ...
        elif self.front_warning > 10 and self.Left_warning > 10 and
self.Right_warning <= 10:
            # print ('4、右转')
            ... ...
            if self.Right_warning <= 10 and self.Left_warning > 10:
                # print ('5、左转')
                ... ...
        elif self.front_warning > 10 and self.Left_warning < 10 and
self.Right_warning < 10:
            # print ('6、右转')
            ... ...
        elif self.front_warning < 10 and self.Left_warning > 10 and
self.Right_warning > 10:
            # print ('7、右转')
            ... ...
        elif self.front_warning < 10 and self.Left_warning > 10 and
self.Right_warning <= 10:
            # print ('8、右转')
```

```
            ... ...
        elif self.front_warning < 10 and self.Left_warning <= 10 and
self.Right_warning > 10:
            # print ('9、左转')
            ... ...
        else:
            # print ('10、前进')
            ... ...
```

根据障碍物，出现的位置，设置小车不同的响应。

源码参数解析：

参数	默认值	判断
self.front_warning	默认为0	当数值大于10，说明前方有障碍物。
self.Left_warning	默认为0	当数值大于10，说明左前方有障碍物。
self.Right_warning	默认为0	当数值大于10，说明右前方有障碍物。