节点：cloud_tf

1.功能：

将原始点云转化为 base_link 坐标下的点云，前提是要建立好 tf 关系

2 发布：

pub_cloud = nh->advertise<sensor_msgs::PointCloud2>("/cloud_tf", 10);（坐标变换后的点云数据）

3.订阅：

ros::Subscriber cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/rslidar_points", 10, PointCloudCallback);

将 ROS 消息中的点云数据提取并转换为 PCL 库中的点云对象 pcl::fromROSMsg(*msg, cloud),

在点云中移除具有无穷（NaN）值的点 cloud = remove_infinite_points(cloud);

应用一个 PassThrough 滤波器，限制在指定的 XYZ 范围内的点 cloud = Cloud_PassThrough(cloud, -max_xy_range, max_xy_range, -max_xy_range, max_xy_range, min_z, max_z);

对点云应用一个变换 cloud= CloudTf(cloud,"base_link");

将处理后的点云转化为 ros 消息后 pcl::toROSMsg(cloud, msgx); 并发布 pub_cloud.publish(msgx)

4.主函数：

声明订阅和发布的消息。

5.其他函数：

CloudTf(cloud,"base_link")函数功能：PointCloud CloudTf(PointCloud cp_src, string frame_id)

移除无效点，并对每个点进行坐标变换到 base_link 下

节点：cloud_segmentation

1 功能：

将点云转换为前后左右四个区域点云

2.发布：

pub_cloud_front = nh->advertise<sensor_msgs::PointCloud2>("agv_front_cloud", 10);

pub_cloud_back = nh->advertise<sensor_msgs::PointCloud2>("agv_back_cloud", 10);

pub_cloud_left = nh->advertise<sensor_msgs::PointCloud2>("agv_left_cloud", 10);

pub_cloud_right = nh->advertise<sensor_msgs::PointCloud2>("agv_right_cloud", 10);

3.订阅：

cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/cloud_tf", 10, PointCloudCallback);

将 ROS 消息中的点云数据提取并转换为 PCL 库中的点云对象，设定距离，将点云数据进行范围划分，再转化为 ros 消息并发布 pub_cloud_front.publish(msgx); pub_cloud_back.publish(msgx); pub_cloud_left.publish(msgx); pub_cloud_right.publish(msgx);

4.主函数：

声明订阅和发布的消息。

节点：cloud_segmentation

1.功能：

本节点输入为前后左右四路激光点云,输出前后左右障碍物最近距离

2.发布：

ros::Publisher pub_obs_dis = nh->advertise<std_msgs::Float32>("obs_dis", 10);

3.订阅：

cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/cloud_segmentation/agv_front_cloud", 10, &PointCloudCallback);

cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/cloud_segmentation/agv_back_cloud", 10, &PointCloudCallback);

cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/cloud_segmentation/agv_left_cloud", 10, &PointCloudCallback);

cloud_sub=nh->subscribe<sensor_msgs::PointCloud2>("/cloud_segmentation/agv_right_cloud", 10, &PointCloudCallback);

将消息转化为ROS消息，如果点云数量小于10就作废，并利用库中已有函数获取三维点云中的最大最小值 pcl::getMinMax3D(cloud, cloud_min, cloud_max); 根据朝向得到点云数据中最小值，并减去车的长度和车的宽度 如 if(dir=="front") obs_dis=fabs(cloud_min.x)-0.5*agv_length;

4.主函数：

声明订阅和发布的消息。滤波障碍物距离 obs_dis，发布障碍物消息 pub_obs_dis.publish(msg);

调试过程问题：

1.obs_dis 的数据类型发生了变化，因此进行了部分修改，其中调试时涉及到了.h 文件的问题

2.坐标系变换进行了修改，在处理障碍物距离的时候 坐标系变化与 turntable 发生了冲突，进行了修改。