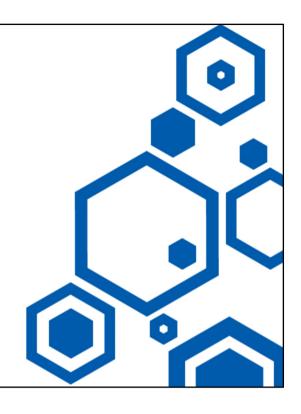


主讲人 贾清源



# 纲要



▶第一部分:地面滤除

▶第二部分:聚类

这一章主要是两个任务,第一个任务是完成地面拟合,第二个任务是在去除地面的基础上做一个聚类

# 地面滤除



### 【RANSAC平面拟合】

- 随机选取三个点,根据它们拟合一个平面
- 计算所有点到平面的距离,根据设定的阈值判断是否属于内点,并记录内点数、平面参数
- 将当前内点数量与之前最好的相比,记录它们之间最优值
- 直到达到最大迭代次数或者达到设置内点率

```
      40
      def ground_segmentation(data):

      41
      # 作业1

      42
      # 屏蔽开始

      43
      n = len(data)

      44
      #设置参数

      45
      iter_num = 100 #迭代次数

      46
      sigma = 0.3 #点到平面的差值

      47
      P = 0.99 #准确率

      48
      outlier_ratio = 0.5 #e

      49
      #记录最好的以合平面参数: Ax + By + Cz + D = 0

      50
      best_idx = []

      51
      best_inliner = (1-outlier_ratio)*n

      52
      best_A, best_B, best_C, best_D = 0, 0, 0, 0
```

做地面去除的话,推荐的算法就是RANSAC平面拟合。第一步是随机选取三个点,然后根据他们拟合一个平面。第二步是计算所有点到平面的一个距离,根据设定的阈值去判断是否属于内点,然后并记录下来这个点数和平面参数。第三步就是将内点数量跟之前最好的一次相比,取一个最优值。第四步就是重复迭代前面三个步骤,然后直到取到一个最大迭代次数,或者达到设置的内点率

# 地面滤除



### 【RANSAC平面拟合】

- 随机选取三个点,根据它们拟合一个平面
- 计算所有点到平面的距离,根据设定的阈值判断是否属于内点,并记录内点数、平面参数
- 将当前内点数量与之前最好的相比,记录它们之间最优值
- 直到达到最大迭代次数或者达到设置内点率

## 聚类



#### 【DBSCAN聚类】

不知道聚类数量,基于密度聚类

- 设置DBSCAN的距离阈值dis,以及min\_sample, 对传入的点构建Kdtree
- 先找出初始核心点,将其填入到core\_set,作为循环开始

```
def clustering(data):
# 作业2
# 屏蔽开始
dis = 0.5
min_sample = 5
n = len(data)

#构建kdtree
leaf_size = 8
kdtree = neighbors.KDTree(data, leaf_size)

#初始化对象集合、未访问集合、聚类个数、聚类索引
core_set = set()
unvisit_set = set(range(n))
k = 0
cluster_index = np.zeros(n, dtype=int)

#適过kdtree与min_sample条件判断所有核心点
nearest_idx = kdtree.query_radius(data, dis)
for i in range(n):
    if len(nearest_idx[i]) >= min_sample:
        core_set.add(i)
```

做聚类的话,这里推荐就是用DBSCAN聚类。因为去除地面之后,并不知道物体的数量,没法设置K参数。DBSCAN的优势就在于此,它是基于点密度来做的聚类。他会把挨着的、比较挨紧凑的一些物体分成一个类别

DBSCAN一般需要两个参数,一个是距离阈值,一个是最小采样点数。在实现过程中,因为要不断求每个点的邻域,所以我们用KDtree来建立一下索引。

## 聚类



从core\_set中随机抽取点,未访问集合中删去它,访问集合中添加它。

- 从访问集合中取第一个点,判断是否为核心点,是的话将其临近点与未访问集合的交集加入到访问集合,未访问集合删去它们,不断循环直到访问集合为空
- 用前一次的未访问集合减去当前的未访问 集合得到第k类聚类,在core\_set中去掉该 类核心点,属于第k类。
- 不断循环,直到core sets为空。

最后将未访问集合设为噪声点,类别为-1

可视化用open3d做一下,matplotlib显示点云数据量太卡了

DBSCAN聚类的算法流程写在左边,大家可以直接看一下。然后还有一点就是最后做可视化的时候,代码里的matplotlib显示点云数据量会比较大比较卡,所以建议大家是用第一章作业中的open3d函数来实现这个可视化





