点云聚类移除外点实时性测试报告

by Xian Zhang, zhangxian@zerozero.cn

1 目的

测试点云聚类计算的效率。

2 方法

假设点云数据有20000个点或一定数量的点，依次对每个点进行聚类分析，最终得到的聚类是有多少，PC上耗时多少。

3 实施

3.1 PC配置

Thinkpad E490，配置如下：

```
cat /proc/cpuinfo     //Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz
free -m               //内存:   total 7821 MB
getconf LONG_BIT      //CPU型号：64位
cat /proc/cpuinfo | grep "physical id" | sort | uniq | wc -l        //CPU 1个
cat /proc/cpuinfo | grep "cpu cores" | wc -l                        //每个CPU 8个
core
```

3.2 聚类理论综述

1 方法调研：参考官网[1]，重点模块如下图

图1 - PCL库主要模块

根据官网提供的内容，所有模块共有15个：

【1】过滤器 Filter：

- 用途：噪音点消除；
- 起因：测量误差；
- 原理：计算点到自身以及它邻域中心的平均距离，若点到邻域的平均距离超出一定范围，即全局平均距离和标准差的规定范围，则视为外点。

【2】特征 Feature：

- 用途：结合点云数据，估计某种特殊数据结构和机制的3D特征，例如法线，曲率等；
- 起因：点的邻域范围内的点信息有助于描述几何外形或图案；
- 原理：估算曲面法线和曲率的方法是计算k邻点图块的特征向量和特征值，最小特征值对应的特征向量即法线，曲率可由特征值得到。

【3】关键点 Keypoint：

- 用途：利用紧凑，稳定，独特的点对数据进行描述性或代表性表示；

- 起因：点云中关键点的数量将远小于云中的总点数，它可以由明确的检测标准检测到；

【4】配准 Registration：

- 用途：将点通过配准的方式整合到全局一致的模型中；
- 原理：确定数据集之间的对应点，并找到一个最小化相应点之间的距离（对齐误差）的变换。这个过程是反复的，因为对应关系受到数据集的相对位置和方向的影响。一旦对齐错误低于给定的阈值，就说配准完成；

【5】Kd树 K-Dimension Tree：

- 用途：方便信息检索，如点或特征描述符组之间的对应关系，或定义一个或多个点周围的局部邻域，使用FLANN；
- 结构：每个节点都为k维点的二叉树，可以是完全二叉树，也可以是非完全二叉树。每个叶节点只能二分，且先左后右。

【6】八叉树 Octree：

- 用途：高效存储数据点云数据，减少内存分配，便于信息搜索；
- 结构：每个八叉树根节点只有一个，叶节点有8个或0个。

【7】分割 Segmentation：

- 用途：对点云数据进行聚类分析，划分成不同的簇，便于区分目标的不同组成；

【8】样本共识 Sample Consensus：

- 用途：根据点云数据进行拟合，例如根据数据，用RANSAC拟合线，圆柱，立方体模型；

【9】表面 Surface：

- 用途：利用3D扫描，重建物体的表面；
- 算法：网格Meshing划分，快速方法是三角化Triangulation原始点；相对慢速方法是网格平滑和提填孔。

【10】测距图像 Range Image：

- 用途：测距图像即深度图像，图像中的像素代表它距离传感器原点的距离，测距图像由双目和ToF相机测量产生，有了相机内参，可将距离图像转换为点云。

【11】输入输出 I/O：

- 用途：点云数据的读取和写入，PCD文件

【12】可视化 Visualization：

- 用途：在3D点云数据上，根据算法计算，展示原型或结果展示

【13】公共库 Common：

- 用途：给PCL其他库提供支持，如规范的数据结构和普适的方法

【14】搜索 Search：

- 用途：提供一些搜索算法
- 算法：Kd树，八叉树，Brute Force，其他方法

【15】二进制 Binaries：

- 用途：对PCL库的常用工具提供快速参考

例如：`pcl_viewer`：可视化PCD（点云数据）文件的快速方法；`convert_pcd_ascii_binary`：将PCD（点云数据）文件从ASCII转换为二进制；`convert_pcd_ascii_binary`：将PCD（点云数据）文件从ASCII转换为二进制；`pcd2vtk`：将PCD（点云数据）文件转换为VTK格式；`mesh2pcd`：使用光线追踪操作将CAD模型转换为PCD（点云数据）文件等等。

2 非官网聚类方法：

- 搜索参考[2]，简称DBSCAN聚类；原理参考[3]；代码参考[4]
- 参考gitub搜索[5]，例如使用 openMP，结合ROS等，还有adaptive clustering

3 总结：涉及到点云聚类分析的只有**【1】过滤器**和**【7】分割**，后面将依次

- 考察官网方法：过滤器和分割，算2万个点的处理耗时；
- 考察非官网方法：DBSCAN方法，算2万个点耗时。

忽略openMP和结合ROS的方法，因飞机可能没有那样的环境；忽略adaptive clustering，点赞只有9 star.

3.3 实际运行效果

时间复杂度的计算在官网上没有官方说明，只能根据算法原理和官网demo去推测大概范围。例如，点云数据有N个点，计算每个查询点到领域内其他N-1个点的距离，则时间复杂度为 $O(N^2)$ ，类似冒泡排序的时间复杂度[8]。如果这里举例有错，指出即可。

【1】过滤器： 利用过滤操作剔除模型外点，**强调外点（噪音点，离群点，远点等）的移除效果**。

- **直通过滤器**：对坐标 X, Y, Z 轴的指定范围进行裁剪，不在坐标轴范围内的点，一律剔除。也可以选择保留范围内的点或者范围外的点。例如创建直通滤波器的对象，设置滤波字段名为 Z 轴方向，可接受范围是 $(0.0, 1.0)$ ，即将点云中所有点的 Z 轴坐标不在该范围内的点过滤或保留；

  图2 2D直通滤波

  如上图，以 X 轴的50为分界线，凡是大于 50的全部是外点，同样也可以 Z 轴为准，凡是 Z 轴的值在0-1之间的属于内点，否则低于0或大于1全是外点。由于遍历每个点，根据字段范围处理，故时间复杂度估计为 $O(N)$ ；
- **体素格过滤器**：在每个体素（即小三维立方体）内，用体素中所有点的质心（同重心）来近似显示体素中其他点。该体素内所有点就可以用一个质心点或重心点来表示，对所有体素处理后就可得到过滤后的点云；体素格滤波器主要作用是对点云数据进行**降采样 Downsampling**，即减少点云中的数据，但大体点云包含的形状结构和特征信息和原数据一致，具体原理需要看源码，估计时间复杂度是 $O(N^2)$ ；
- **投影模型过滤器**：投影点指对三维空间中的点在其某个平面 $(e.g., oxy, oyz, oxz_{平面})$ 的投影；参数模型指几何表面模型，例如线条，圆圈，平面，球面，曲面等，但该几何表面由一系列系数制约，例如某参数模型是 $ax + by + cz = 0$，显然系数 a, b, c 制约着该平面模型的空间位置。由于模型拟合是RANSAC，例如先选择两点解析直线方程，再遍历其他点是否满足该方程形式，最后根据阈值判断是内点还是外点，故故时间复杂度类似 $O(N)$ ；
- **提取点云索引**：索引即点云中不同点的标签或下标，方便点云分类提取操作的。例如，点云操作过程中经常需要提取点云子集，包括一些滤波算法也会提取点云的索引，然后根据索引提取点云子集。提取点云索引，需要使用**模型分割 Model Segmentation**，即找到支撑模型（例如线条，圆圈，平面，柱面，球面等）的点云数据，剔除离群点。 **原理**是RANSAC，采取用户定义的**距离阈值 Distance Threshold**，时间复杂度类似 $O(N)$；
- **条件过滤器**：原理是第一步设置查询点的半径（例如0.8），同时设置一个阈值条件，例如minNeighbor(2)；第二步，根据第一个半径条件搜索查询点的领域点，然后判断领域点到查询点距离是否大于或等于阈值条件。如果是则属于内点，否则属于外点。条件除了距离，也可类似直通滤波，设置字段名，根据内置的**比较算子**， 例如 GT(great than), EQ(equal), LT(less than),

GE(greater than or equal)等设定条件，然后执行过滤操作；时间复杂度类似 $O(N^2)$；

- **截取框过滤器**：定义一个2D/3D的凸/凹框（e.g. 某平面2D矩形框），根据框的顶点位置，判断点云是否在框之内（根据顶点坐标范围），过滤不在该框的外点，官网介绍参考[6]，由于需要遍历每个点，判断每个点的字段范围，时间复杂度估计是$O(N)$；
- **统计过滤器**：第一步遍历点云每个点，根据人为设定的K值选择每个查询点的K个近邻点，计算其距离上的均值和标准差；第二步根据每个查询点的距离均值和标准差形成正态分布，设定距离阈值；第三步再次遍历每个查询点到邻近点的距离，如果距离在阈值之外，则判断为外点[7]；时间复杂度类似$O(N^2)$。

图2 - 滤波过滤模型外点

【2】分割：利用滤波和分割算法，二者偶尔有重合。但分割强调不同表面有所不同的效果，即点云数据的聚类分析。剔除外点只是分割算法的局部操作，且个别分割算法完全用不到滤波操作。

算法介绍如下：如果需要参考连接，联系笔者即可。

- 平面模型分割 Plane Model Segmentation，算法是**RANSAC**；
- 圆柱模型分割 Cylinder Model Segmentation，算法是**RANSAC**（官网）；
- 欧几里得聚类提取 Euclidean Cluster Extraction，算法是距离阈值；
- 区域生长分割 Region growing segmentation，算法是利用法线，曲率，欧式距离计算进行点云聚类分析；
- 基于彩色信息生长分割 Color-based region growing segmentation，算法同区域生长分割类似，但用颜色信息代替法线，应用欧式距离计算进行点云聚类分析（官网）；
- 基于最小切割的分割 Min-cut Based Segmentation，算法是图论中的最小分割算法；
- 有条件的欧几里得聚类 Conditional Euclidean Clustering，算法同区域生长分割类似，是它的拓展版本；
- 基于法线不同（也叫法线差异）分割 Difference of Normals Based Segmentation，算法思想借鉴高斯不同**DOG**，涉及不同尺度下如何估计法线，正确选取支持半径，进而分割；
- 点云聚类为超体元（也叫超体聚类） Clustering of Pointclouds into Supervoxels，算法思想是对局部特征进行总结，如纹理，颜色，材质等自动分割成一块，归入最小晶粒 **MOV**，然后指定粒子距离 **Rvoxel**和晶核距离 **Rseed**，进而实现晶体生长，分割；
- 渐进型形态学过滤器分割（也叫地面点云分割） Identifying ground returns using ProgressiveMorphologicalFilter segementation，算法本身用于处理高空获取的激光雷达数据，把地面与非地面的物体分割，来获取地貌**3d**地图。涉及法向量，栅格高度差，水平面校准等激光雷达地面点云分割方法；
- 基于模型外点移除过滤点云 Filtering a PointCloud using ModelOutlierRemoval，算法是根据模型和点之间的距离阈值，移除模型中的离群点和**NaN**。

图3 - 分割算法实际操作效率

4 总结

凡是非官方提出的方法不建议不考虑，例如机器学习聚类分析算法，如KNN, K-Means, DBSCAN等；主流参考上述。

5 参考文献

[1] http://pointclouds.org/documentation/

[2] https://blog.csdn.net/weixin_42718092/article/details/86221246

[3] https://www.cnblogs.com/bonelee/p/8692336.html

[4] https://github.com/buresu/DBSCAN/blob/master/dbscan.cpp

[5] https://github.com/search?l=C%2B%2B&q=point+cloud+clustering&type=Repositories

[6] http://docs.pointclouds.org/trunk/classpcl_1_1_crop_hull.html

[7] https://blog.csdn.net/qq_22170875/article/details/84994029

[8] https://www.cnblogs.com/wuxiangli/p/6399266.html