基于 VG-DBSCAN 算法的大场景散乱点云去噪

赵凯[1*]，徐友春[2**]，李永乐[2]，王任栋[1]

[1]陆军军事交通学院，天津 300161；

[2]军事交通运输研究所，天津 300161

摘要 针对城市环境下三维激光雷达(LiDAR)点云数据密度不均匀、离群噪点多而不利于后期点云帧间匹配的问题，提出一种应用于城市环境下大规模散乱 LiDAR 点云的离群噪点滤除算法。该算法对传统的基于密度的噪声应用空间聚类(DBSCAN)算法进行改进，通过对三维点云进行体素栅格划分，创建了一个由栅格单元组成的集合，以此大幅减小每个对象在数据空间中邻域的搜索范围。改进后的算法能够快速发现各个聚类，使目标点云与离群点分离，从而剔除点云中的离群噪点。实验结果表明：所提算法能够实时处理点云数据，在保证点云三维几何特征的同时能有效识别并滤除点云中的离群噪点，降低点云规模，加快点云后续处理的效率，使帧间匹配的精确度提高了 2 倍，且匹配耗时仅为去噪处理前的 1/3。

关键词 遥感；激光雷达；点云去噪；密度聚类

中图分类号 TP242 **文献标识码** A **doi**：10.3788/AOS201838.1028001

Large-Scale Scattered Point-Cloud Denoising Based on VG-DBSCAN Algorithm

Zhao Kai[1*], Xu Youchun[2**], Li Yongle[2], Wang Rendong[1]

[1]*Army Military Transportation University, Tianjin* 300161, *China*;

[2]*Institute of Military Transportation, Tianjin* 300161, *China*

Abstract Non-uniform 3D light detection and ranging (LiDAR) point-cloud data with outlier noises are not conducive to interframe point-cloud-matching in urban environments. Thus, an outlier noise filtering algorithm for large-scale scattered LiDAR point-cloud in urban environments is proposed. This algorithm improves the traditional density-based spatial clustering of applications with noise (DBSCAN) algorithm by applying voxel-grid partitioning to the three-dimensional point-cloud to create a set of grid cells, which greatly reduces the search scope of each object's neighborhood in the data-space range. The improved algorithm can quickly find each cluster, which separates the target point-cloud from the outliers, thus eliminating the outlier noise in the point-cloud. The experimental results show that the proposed algorithm can process point-cloud data in real-time, ensure three-dimensional geometric features of point-cloud, effectively recognize and filter out outlier noise, reduce the scale of point-cloud, and speed up the subsequent processing efficiency of the point-cloud. Using this algorithm, the accuracy of matching between the frames is doubled, and the matching time is only one-third of the time before denoising.

Key words remote sensing; LiDAR; point-cloud denoising; density clustering

OCIS codes 280.3640; 200.4560; 150.6910; 150.5670

1 引 言

城市道路环境以其复杂、多变、规模大等特点成为当前自动驾驶领域研究的热点和难点[1-5]。在城市三维环境下，多线激光雷达(LiDAR)扫描通常会得到密度不均匀的点云数据集，并且传感器的局限性、采集设备的固有噪声、场景中物体表面的反射特性等会不可避免地使点云数据产生大量的离群点。噪声点的存在会严重影响后续点云的特征提取和特征匹配。因此，有必要对原始点云进行滤波操作，以获得适合进一步处理的精确点云。Fleishman 等[6]对双边滤波进行改进，并拓展到三维网格模型上，提

收稿日期：2018-04-20；**修回日期**：2018-05-10；**录用日期**：2018-05-16

基金项目：国家重点研发计划(2016YFB0101001-6)

* **E-mail**：zhkai929@126.com；** **E-mail**：xu56419@126.com

出针对网格降噪的双边滤波算法，该算法通过对影响域和空间域赋予不同的权值函数来进行各向异性的去噪，具有良好的降噪效果，且能够保持点模型的几何特征，但不能很好地对离群点加以处理，易削去模型中最尖锐的部分。聂建辉等[7]在分析点云离群点数据产生的原因后引入数据挖掘分类算法，将离群点分为远离群点和近离群点两类，并采用基于三维区域增长的方法识别远离群点，使用基于曲面变化度的局部离群系数方法识别近离群点，该算法可以有效识别离群点数据并进行去噪。李仁忠等[8]提出一种基于方法库的点云模型去噪与精简算法，依据噪声点到模型主体的距离，利用统计滤波结合半径滤波去除大尺度噪声。苏本跃等[9]用 K-means 聚类算法对点云进行聚类，根据点到聚类中心的欧氏距离和邻近点曲率变化判断每个类中的点是否为噪声点，进而去除大量外部噪声点。然而上述方法均不适用于大规模散乱点云（如城市环境）的去噪处理。

鉴于上述算法的局限性，本文提出了一种针对大规模散乱点云的去噪算法。该算法首先对三维 LiDAR 点云进行体素栅格划分，然后确定核心点、搜索栅格邻域、划分每个栅格的簇类，最终将目标点云聚类，使目标点云与离群点云分离，从而达到滤除离群噪点的目的。该算法在去除离群噪点和冗余信息、降低点云数量规模的同时还能保留点云的特征信息，不损坏点云的细节信息，提高了后续点云匹配的精确度，且缩短了匹配耗时。

2　基于密度的带噪声数据空间聚类算法

基于密度的带噪声数据空间聚类算法（DBSCAN）是第一个基于密度的聚类算法，由 Ester 等[10]于 1996 年提出。该算法将簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在存在噪声的数据中发现任意形状的聚类。该算法的显著优点是聚类速度快，且能有效处理噪声点。

DBSCAN 算法的核心思想总结如下：

该算法有两个参数，即 Eps 和 MinPts，其中 Eps 表示目标数据点搜索邻域的半径，MinPts 表示最小邻域点数。在图 1 中，如果 c 点的 Eps 邻域至少有 MinPts 个点，则 c 被称为核心点。搜索区域中的点将重复聚类，直到仅剩下 Eps 邻域内点数小于 MinPts 的数据点。点 b 在点 p 的 Eps 邻域内，但点 b 的 Eps 邻域内的点数小于 MinPts，因此，b 点被标记为边界点（b 不是核心点，但落在某个核心点的 Eps 邻域内）。点 o 的邻域点数少于 MinPts，因此不能成为新的簇。

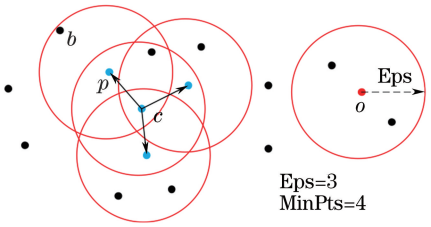

图 1　DBSCAN 算法的核心思想

Fig. 1　Core idea of DBSCAN algorithm

3　基于体素栅格的快速密度聚类算法

密度聚类算法可以通过选择合适的阈值将目标点云与离群点分离，达到去除离群点的目的。在大规模散乱点云数据的处理中，由于噪声的数目、分布均未知，所以 DBSCAN 算法能够较好地应用在此类问题中。但是在 DBSCAN 算法和基于相对密度的聚类方法中，每个对象点必须与数据集中众多的其他对象点进行比较。为了降低执行时间成本，本研究对 DBSCAN 算法进行了改进，提出了一种基于体素栅格的快速密度聚类（VG-DBSCAN）算法，利用该算法对点云进行滤除离群噪点的处理。

VG-DBSCAN 算法将三维点云数据按照维度划分为以体素栅格为单元的多个相邻的区间，创建一个由栅格单元组成的集合，以此来大幅减小每个对象在数据空间中邻域的搜索范围，只要考察当前对象的空间相邻栅格单元就能实现其邻域的搜索，快速发现各个聚类。VG-DBSCAN 算法的具体步骤如下。

步骤 1：划分体素栅格。如图 2(a)所示，构建一个三维体素栅格，并将三维 LiDAR 点云数据的每个数据点划分给它所在的栅格单元。每个单元栅格的对角线长度是 Eps，这意味着栅格单元的边长为 $\mathrm{Eps}/3^{1/2}$，如图 2(b)所示。如此设定单元格大小的

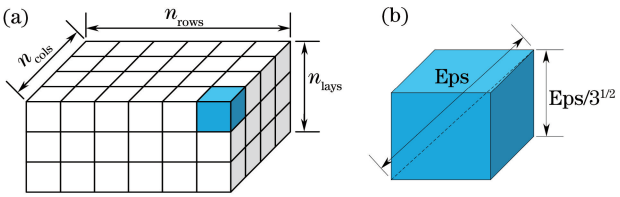

图 2　划分三维体素栅格。

(a)三维体素栅格；(b)体素单元格

Fig. 2　Dividing a three-dimensional voxel grid.

(a) Three-dimensional voxel grid; (b) voxel cell

意义在于如果栅格内的点数大于 MinPts，则能够直接将所有这些包含于该体素栅格单元的点标记为核心点。因为体素栅格内任意两点在栅格内的最大距离不大于 Eps，所以如果体素栅格内有大于 MinPts 数量的数据点，那么单元格内任意点的 Eps 邻域至少包含 MinPts 个点。

在构建栅格之前，首先需要确定 6 个值，即三维点云数据 X、Y、Z 轴的最大坐标值和最小坐标值。通过这 6 个值和体素栅格的边长就可以确定栅格的行数 n_{rows}、列数 n_{cols} 和层数 n_{lays}，然后计算每个数据点所属的体素栅格。对于体素栅格边界上的点，将其分配在右上方的栅格中。如果有一个数据点正好位于栅格的最顶部或最右边体素栅格的边界上，这时应添加一个额外的行、列、层，用以囊括该点，以防止将其分配到栅格外部。

步骤 2：确定核心点。该步骤将遍历所有的非空体素栅格。要查找栅格内的所有核心点，首先必须检查该单元格内的点数是否大于 MinPts。如果是，则将所有这些点标记为核心点；如果不是，则必须检查该单元格内的每个点，并确定它是否为核心点。要确定点 $p(p\in C)$ 是否是核心点，需要检查该点所在栅格的相邻栅格 $N_{\mathrm{Eps}}(C)$ 内的一些点。$N_{\mathrm{Eps}}(C)$ 定义为某个体素栅格单元 C 及其相邻的栅格单元 C' 的集合，即 $N_{\mathrm{Eps}}(C)=\{C,C'\,|\,\mathrm{dist}(C,C')\leqslant \mathrm{Eps}\}$。

图 3 为体素栅格某层的二维示意图，可以看到：对于三维体素栅格，最多需要检查 124 个栅格单元。计算这些栅格中每个点与 p 之间的距离。当这些单元栅格内的所有点已被检查过，或者当发现 p 的 Eps 邻域内的点数大于等于 MinPts 时，终止查找。

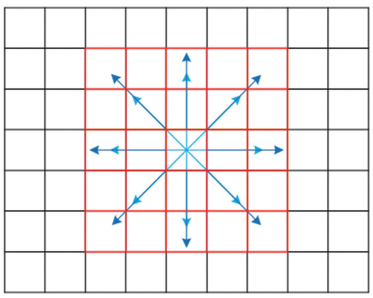

图 3　体素栅格某层的二维示意图

Fig. 3　Two-dimensional schematic of a layer of a voxel grid

步骤 3：合并簇。如果两个不同单元中的两个核心点之间的距离小于等于 Eps，则这两个点属于同一个簇。举个简单的例子，如图 4 所示，相邻的两个栅格单元中各含有 4 个数据点，假设 MinPts=3，则两个栅格单元内的所有点都是核心点。如果 a 与 b 之间的距离小于等于 Eps，则可以得出这 8 个点都属于同一个簇的结论。否则，划分为两个各含有 4 个点的簇。

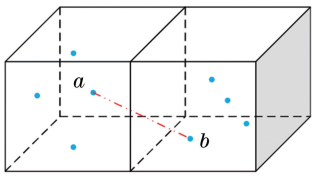

图 4　合并簇的简例

Fig. 4　Simplified example of a merged cluster

步骤 4：确定边界点和离群噪点。这是 VG-DBSCAN 算法的最后一步，这一步通过遍历所有非核心点来完成。如果在 $N_{\mathrm{Eps}}(C)$ 中存在至少一个核心点，并且最靠近 p 的核心点属于簇 clusters_1，则非核心点 p 是簇 clusters_1 的边界点。完成上述 4 个步骤之后，所有不是核心点或边界点的数据点都会被标记为离群噪点。

通过上述 4 个步骤可以快速聚类目标点云，使其与离群噪点分离，滤除被标记为离群噪点的数据点即达到了点云去噪的目的。

4　实验结果与分析

为了验证所提算法的正确性和有效性，采用 Velodyne HDL-64E 型 64 线 LiDAR 采集城市复杂动态环境下的三维点云数据，并将其作为研究对象。实验平台为 Intel(R) Core(TM) i7-6700 CPU@3.40 GHz，16 GB 随机存取存储器(RAM)，Unbuntu 16.04操作系统，Qt Creator5.7 开发环境，开发语言为C++。首先，选取连续 1000 帧经过地面分割处理的点云数据，这些点云中均含有大量的动静态障碍物，如机动车、非机动车、行人等，同时含有大量的离群点。然后，通过所提算法对点云进行滤除离群噪点处理，并与开源点云库 PCL·1.8.1dev 中具有代表性的统计滤波去噪算法和半径滤波去噪算法进行对比，实验结果如图 5～9 和表 1 所示。图 5 是一

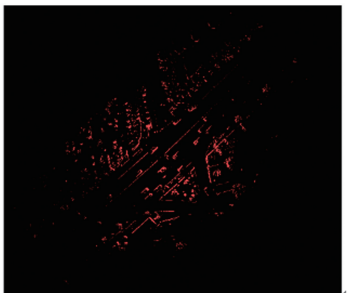

图 5　原始点云

Fig. 5　Original point-cloud

帧经过地面分割处理的原始三维点云数据，可以看出，原始点云数据分布稀疏杂乱且含有大量离群噪点。图 6 是统计滤波去噪算法的结果，该算法的主要参数是 MeanK，用于确定平均距离估计时的最近邻点数。图 7 是半径滤波去噪算法的结果，该算法的主要参数是阈值 MinNeighbors，该阈值是半径最近邻内所包含的最小点数。图 8 是所提算法对点云去噪处理的结果。参数 MinPts 与 MeanK、MinNeighbors 的含义基本类似，都是确定邻域内点云的数量，所以 3 种算法均以这一参数为变量进行了对比实验。结果显示，这 3 个参数设置过大都会对点云数据进行过度处理，即去除了部分小的障碍物，但参数在一定的范围内可以保证 3 种算法在去除离群噪点的同时保留障碍物的局部特征信息，并在一定程度上降低了点云的规模。图 9(b)所示为图 8(c)中的局部，可以看出：所提算法能够很好地聚类主要的目标物，使其与噪声点分离，并滤除这些离群噪点。

表 1 统计了 1000 帧点云处理结果的平均水平，包括去噪前、后点云数量以及处理耗时这 3 个指标的平均值。从表 1 可以看出：3 种算法在去噪前、后的点云数量上基本持平，但是处理时间相差悬殊，统计滤波去噪算法的耗时约为 200 ms，并且耗时随着参数 MeanK 的增大而增大；半径滤波去噪算法的耗时约为 470 ms，且耗时不会随着参数 MinNeighbors 的变化而发生明显变化；所提算法的耗时约为 100 ms，耗时随参数 MinPts 的增大而增大，但增加幅度不大(与统计滤波去噪算法相比)。综上所述，相比于其他两种典型算法，所提算法在点云去噪、简化以及耗时方面均有优势。

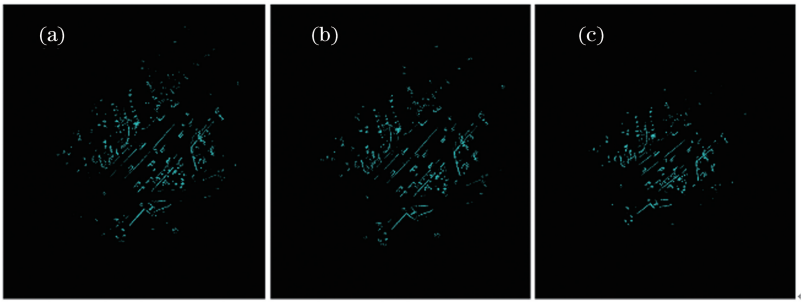

图 6　统计滤波去噪算法的结果。(a) MeanK＝10；(b) MeanK＝30；(c) MeanK＝50

Fig. 6　Results using statistical outlier removal algorithm. (a) MeanK＝10; (b) MeanK＝30; (c) MeanK＝50

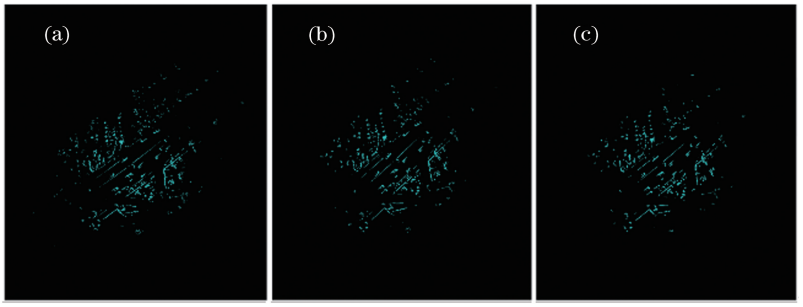

图 7　半径滤波去噪算法的结果。(a) MinNeighbors＝5；(b) MinNeighbors＝10；(c) MinNeighbors＝15

Fig. 7　Results using radius outlier removal algorithm. (a) MinNeighbors＝5; (b) MinNeighbors＝10; (c) MinNeighbors＝15

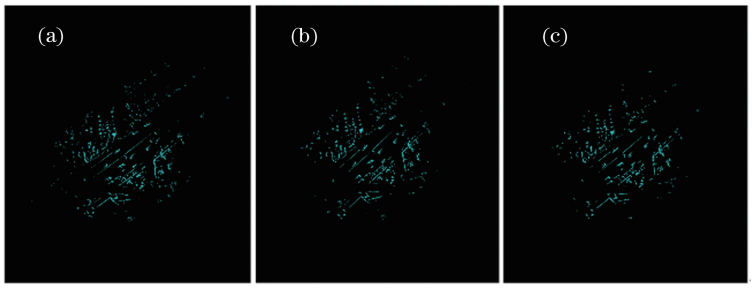

图 8　VG-DBSCAN 算法的去噪结果。(a) Eps＝1，MinPts＝10；(b) Eps＝1，MinPts＝15；(c) Eps＝1，MinPts＝20

Fig. 8　Denoising results using VG-DBSCAN algorithm. (a) Eps＝1, MinPts＝10; (b) Eps＝1, MinPts＝15; (c) Eps＝1, MinPts＝20

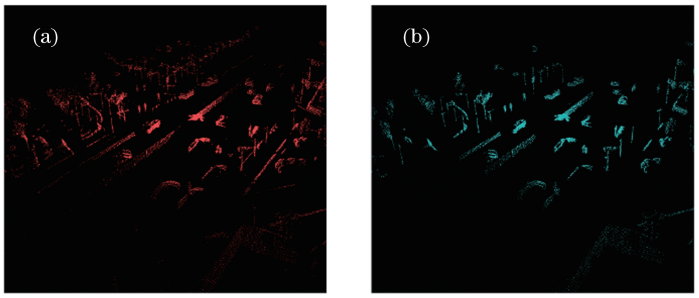

图 9　VG-DBSCAN 算法去噪的局部效果。(a)去噪前;(b)去噪后

Fig. 9　Local denoising results using VG-DBSCAN algorithm. (a) Before denoising; (b) after denoising

表 1　3 种算法的去噪结果

Table 1　Denosing results using three algorithms

Algorithm	Parameter	Point size		Consuming time / ms
		Original	After denoising	
Statistical outlier removal	MeanK=10	42618	38284	180.23
	MeanK=30		37945	227.68
	MeanK=50		36579	288.85
Radius outlier removal	MinNeighbors=5	42618	39744	465.37
	MinNeighbors=10		38543	466.91
	MinNeighbors=15		37349	471.15
VG-DBSCAN	Eps=1, MinPts=10	42618	38248	92.74
	Eps=1, MinPts=15		37153	109.39
	Eps=1, MinPts=20		36142	124.69

为了验证所提算法对点云帧间匹配的改善效果,使用开源库 PCL·1.8.1dev 中的匹配算法——广义迭代最近点(GICP)算法对点云去噪处理前后的帧间匹配进行了对比实验。实验结果如图 10 和表 2 所示。将欧氏适应度[11]作为点云匹配准确度的衡量标准,欧氏适应度为参与匹配的两帧点云中各对应点对之间欧氏距离的平方和。表 2 的数据表明,经过预处理的点云在规模上得到了大幅度简化,基本上滤除了所有的离群噪点,这使得帧间匹配的精确度提高了 2 倍,且匹配耗时仅为之前的 1/3。

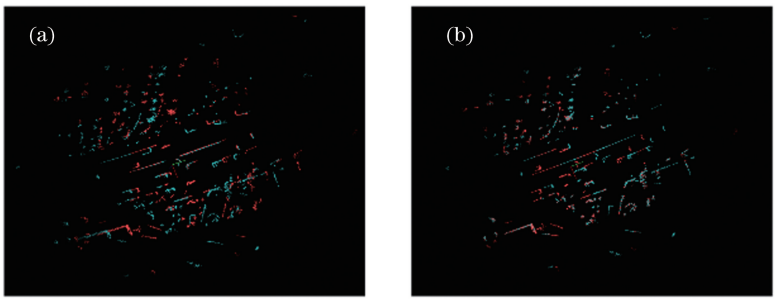

图 10　去噪后点云帧间的匹配结果。(a)匹配前;(b)匹配后

Fig. 10　Point-cloud-matching results after denoising. (a) Before matching; (b) after matching

表 2　去噪处理前后点云的匹配结果

Table 2　Point-cloud-matching results before and after denoising

Denoising operation	Point size		Euclidean fitness score	Consuming time /s
	Current frame	Previous frame		
Without	42618	45104	13.27145	66.752
With	36015	37549	6.12517	21.215

5 结 论

提出了一种应用于城市复杂环境下散乱三维LiDAR点云的去噪算法——VG-DBSCAN算法。该算法采用点云三维体素栅格划分的形式对传统DBSCAN算法进行改进，将点云囊括进每个栅格中，创建一个由栅格单元组成的集合，以此来大幅减小每个对象在数据空间中邻域的搜索范围，实现点云的快速目标聚类，达到分离出离群噪点的目的。通过与点云处理开源库中具有代表性的统计滤波和半径滤波算法进行对比，证明了所提算法能够在保证点云三维几何特征的同时实时处理点云数据，而且可以有效识别并滤除点云中的离群噪点，在点云去噪、简化以及耗时方面均优于现有的典型算法。之后，采用该算法对帧间匹配前两帧点云进行去噪处理，实验结果证明了所提算法可以显著提高匹配的准确度并减少匹配耗时。综上，所提点云去噪算法可以实时为点云帧间匹配提供更优质的点云数据。

参 考 文 献

[1] Uehara K, Saito H, Hara K. Line-based SLAM considering directional distribution of line features in an urban environment[C]. International Conference on Computer Vision Theory and Applications, 2017, 6: 255-264.

[2] Han D B, Xu Y C, Wang R D, *et al*. Calibration of three-dimensional lidar extrinsic parameters based on multiple-point clouds matching [J]. Laser & Optoelectronics Progress, 2018, 55(2): 022803.
韩栋斌, 徐友春, 王任栋, 等. 基于多对点云匹配的三维激光雷达外参数标定[J]. 激光与光电子学进展, 2018, 55(2): 022803.

[3] Chen G B, Gao Z H, He L. Step-by-step automatic calibration algorithm for exterior parameters of 3D lidar mounted on vehicle [J]. Chinese Journal of Lasers, 2017, 44(10): 1010004.
陈贵宾, 高振海, 何磊. 车载三维激光雷达外参数的分步自动标定算法[J]. 中国激光, 2017, 44(10): 1010004.

[4] Kim J U, Kang H B. LiDAR based 3D object detection using CCD information[C]. IEEE Third International Conference on Multimedia Big Data, California, 2017: 1701-1750.

[5] Xiong F G, Huo W, Han X, *et al*. Removal method of mismatching keypoints in 3D point cloud[J]. Acta Optica Sinica, 2018, 38(2): 0210003.
熊风光, 霍旺, 韩燮, 等. 三维点云中关键点误匹配剔除方法[J]. 光学学报, 2018, 38(2): 0210003.

[6] Fleishman S, Drori I, Cohen-Or D. Bilateral mesh denoising [J]. ACM Transactions on Graphics, 2003, 22(3): 950-953.

[7] Nie J H, Hu Y, Ma Z. Outlier detection of scattered point cloud by classification[J]. Journal of Computer-Aided Design & Computer Graphics, 2011, 23(9): 1526-1532.
聂建辉, 胡英, 马孜. 散乱点云离群点的分类识别算法[J]. 计算机辅助设计与图形学学报, 2011, 23(9): 1526-1532.

[8] Li R Z, Yang M, Ran Y, *et al*. Point cloud denoising and simplification algorithm based on method library [J]. Laser & Optoelectronics Progress, 2018, 55(1): 011008.
李仁忠, 杨曼, 冉媛, 等. 基于方法库的点云去噪与精简算法[J]. 激光与光电子学进展, 2018, 55(1): 011008.

[9] Su B Y, Ma J Y, Peng Y S, *et al*. Algorithm for RGBD point cloud denoising and simplification based on *K*-means clustering [J]. Journal of System Simulation, 2016, 28(10): 2329-2334.
苏本跃, 马金宇, 彭玉升, 等. 基于*K*-means聚类的RGBD点云去噪和精简算法[J]. 系统仿真学报, 2016, 28(10): 2329-2334.

[10] Ester M, Kriegel H P, Xu X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise[C]. International Conference on Knowledge Discovery and Data Mining, Oregon, 1996: 226-231.

[11] Rusu R B, Cousins S. 3D is here: point cloud library (PCL)[J]. Proceedings of the IEEE, 2011, 47(10): 1-4.