sensors

MDPI

Article

Robust and Fast Normal Mollification via Consistent Neighborhood Reconstruction for Unorganized Point Clouds

Guangshuai Liu [1,2,*], Xurui Li [1,*], Si Sun [3] and Wenyu Yi [4]

[1] School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China
[2] Sichuan Province Informationization Application Support Software Engineering Technology Research Center, Chengdu 610103, China
[3] Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu 610209, China
[4] Sichuan Research and Design Institute of Agricultural Machinery, Chengdu 610066, China
* Correspondence: motorliu7810@swjtu.edu.cn (G.L.); xuruili@my.swjtu.edu.cn (X.L.)

Abstract: This paper introduces a robust normal estimation method for point cloud data that can handle both smooth and sharp features. Our method is based on the inclusion of neighborhood recognition into the normal mollification process in the neighborhood of the current point: First, the point cloud surfaces are assigned normals via a normal estimator of robust location (NERL), which guarantees the reliability of the smooth region normals, and then a robust feature point recognition method is proposed to identify points around sharp features accurately. Furthermore, Gaussian maps and clustering are adopted for feature points to seek a rough isotropic neighborhood for the first-stage normal mollification. In order to further deal with non-uniform sampling or various complex scenes efficiently, the second-stage normal mollification based on residual is proposed. The proposed method was experimentally validated on synthetic and real-world datasets and compared to state-of-the-art methods.

Keywords: normal estimation; point cloud; feature preserving; normal mollification

Citation: Liu, G.; Li, X.; Sun, S.; Yi, W. Robust and Fast Normal Mollification via Consistent Neighborhood Reconstruction for Unorganized Point Clouds. *Sensors* **2023**, *23*, 3292. https://doi.org/10.3390/s23063292

Academic Editor: Jason Rambach

Received: 17 January 2023
Revised: 15 March 2023
Accepted: 17 March 2023
Published: 20 March 2023

1. Introduction

The normal information is a crucial geometric property of 3D point clouds, which is extensively applied in many fields, such as denoising [1,2], surface reconstruction [3,4], resampling [5] and consolidation [6], and feature detection and extraction [7]. The inaccurate normals are likely to result in the loss of detailed features on the point cloud surface, thereby negatively impacting the subsequent applications. Although many works have been proposed in this research field, normal estimation still confronts numerous challenges. First of all, normal estimation should have high precision, notably for noisy point clouds or non-uniformly sampled surfaces. Next, computational efficiency must be ensured to verify practical significance. Thirdly, adaptive parameters should be designed to apply the estimator to various conditions.

Regression-based normal estimation methods [3,8] are the most widely employed on account of their high efficiency and strong applicability. However, such methods are actually low-pass filters. Consequently, any sharp features of the surface will unavoidably be smoothed. In this case, normal mollification techniques [4,9] are proposed to improve the initial normal field. Nevertheless, these methods require refined parameter tuning and can no longer reliably recover curvature discontinuities under the excessive smoothing of sharp features. Additionally, segmentation-based approaches [10,11] are proposed to generate isotropic sub-neighborhood matching the current point to estimate normal. These methods can generate more reliable normals than other methods but usually require longer run times.

Motivated by the above consideration, we propose a fast, high-quality normal mollification method. In contrast with the conventional normal mollification process, a rough

isotropic sub-neighborhood is sought and added in our method, which effectively avoids the requirement of reliable initial normals and refined parameters. Throughout the process, the normals of all points are initialized by a normal estimator of robust location. Then feature coefficient that measures the extent of points are near to sharp region is utilized to classify the point cloud into feature points and non-feature points. For feature points, an isotropic sub-neighborhood constructed by a Gaussian map and clustering is utilized for the first-stage normal mollification. Meanwhile, further secondary mollification can ensure steady and accurate estimation under various difficult scenarios. The experiments demonstrate that the presented method yields accurate and efficient results in the presence of noise and anisotropic samplings while preserving sharp features. Our main contributions can be summarized as follows:

- A normal estimator of robust location is used to estimate initial normals, which ensures the reliability of the smooth region normal.
- A robust recognition method based on normal differences is proposed to identify points close to sharp features. This method can accurately identify feature points at high noise levels.
- A robust normal mollification process based on neighborhood recognition is proposed, which can efficiently and reliably estimate normals for points around sharp features even in the presence of noise and non-uniform sampling.

2. Related Works

Normal estimation is challenging for point clouds with noise, non-uniformity of sampling, and sharp features. Considerable approaches have been developed for normal estimation in the literature, which are divided into the following six categories:

The first category is based on regression. The classical normal estimation method, proposed by Hoppe et al. [3] (PCA), defines the normal of a point as the eigenvector corresponding to the smallest eigenvalue of the covariance matrix of its neighbors, where the point's local neighborhood is approximated by a plane. Mitra et al. [12] proposed adaptive neighborhood size to improve the robustness of the regression method to noise. However, the method tends to smooth sharp features and thus cannot correctly estimate the normal near the edges. Moreover, by assigning Gaussian weight to the current point's neighbors during the plane fitting, a weighted version PCA is proposed in [8,13], which benefits from weakening the influence of some points, such as noise and outlier points. To better adapt to the shape of the underlying surface, Cazals et al. [14] and Guennebaud [15] utilize higher-order quadric surfaces and algebraic spheres to replace the plane. Mederos et al. [16] extend the plane fitting by introducing a robust statistic approach called M-estimator to reduce the impact of the neighbors belonging to different surface patches, but Newton's method is needed to solve the normal; thus, the calculation is relatively complicated. Wang et al. [17] further consider the normals' likeliness to penalize neighbors having normals much different from the normal of the current point. Recently, Sanchez et al. [18] introduced an iterative weighted PCA (IterWPCA) from a robust M-estimator and effectively dealt with the noise and anisotropy problems. However, the specified curvature and noise scale assigned to the global may cause the algorithm not to converge to the optimal solution.

Another train of thought is based on the Voronoi diagram, which was first proposed by Amenta [19]. In this method, the furthest pole of the Voronoi lattice is used to approximate the normals, but it works only for the noise-free point clouds. Dey et al. [20] extend the idea by seeking the Delaunay sphere and refining the poles, thereby being able to estimate the normal while the presence of noise. Based on the former theories, Alliez et al. [21] present a method that combines the advantages of PCA and Voronoi-based to achieve more stable normal estimation results.

To lessen the impact of noise and outlier points while improving the initial normals, the methods based on normal mollification are studied. Yagou et al. [22,23] propose filters to mollify the normal field locally and introduce three kinds of filters: mean filter, median filter, and alpha-trimming filter to reduce the sensitivity to noise while recovering curvature

discontinuities. Similarly, Jones et al. [9] extend bilateral filtering to estimate normals. Öztireli et al. [4] expound the implicit least square surface from the perspective of local kernel regression, and take the robust local nuclear regression method in robust statistics for reference, proposing a robust normal mollification method. Additionally, a half-quadratic regularization method [24] also restores sharp features while improving noisy normals. Although normal mollification methods can obtain virtually correct normal [25] for the points close to sharp features, as post-processing methods, all of them require dependable initial normal.

The methods mentioned above are limited in the capability to maintain sharp features; hence the fourth category based on the voting method is developed. Li et al. [26] propose a robust local noise estimation method combined with kernel density estimation, which votes all the neighborhood points on a plane set determined by arbitrary triples to select the optimal tangent plane. This approach can accurately estimate the normal of feature points and have strong robustness to noise and outliers. However, it does not take non-uniform sampling into account. To handle this issue, a uniform sampling technique based on the randomized Hough transform is proposed by Boulch and Marlet [27]. Whereas in the case of a large dihedral angle, the difference in normals of plane sampling will be trivial, so these normals will vote for the identical bin, resulting in the blurring of the normals near its edge. Additionally, Zhang et al. [25] introduce a pair consistency voting algorithm (PCV) to gain the optimal tangent plane via point pairs voting between neighborhood points and use density weights to solve non-uniform sampling issues. In order to deal effectively with noise and outliers, Mura et al. [28] proposed a method based on robust statistics that can simultaneously maintain the sharp features of the point cloud. Such methods usually consider that points with large fitting residuals have a negative impact on the normal estimation and should be reduced or dropped during the plane fitting.

The fifth category is based on neighborhood segmentation. Fleishman et al. [10] propose robust moving least squares to segment the neighborhood into multiple piecewise smooth regions, whereas normal estimation of this method is on the premise of reconstruction; thus, it is time-consuming. Zhang et al. [11] design an unsupervised learning process, adopting the low-rank subspace clustering method (LRR) with prior knowledge to divide into multiple isotropic neighborhoods. However, a lot of time is taken to solve the model in each segmentation process. To reduce the computational time, based on LRR technology, Liu et al. [29] utilize the least square method as a guide to segmentation neighborhood, decreasing time and ensuring its high-quality segmentation. Moreover, Yu et al. [30] proposed a neighborhood segmentation-based and neighborhood growth-based method to construct a consistent neighborhood with the current point, but this method could be labile while dealing with sparse sampling models. Different from pure segmentation, Cao et al. [31] obtain the surface patch consistent with the current point through neighborhood shift techniques to estimate accurate normals.

Recently, the sixth category of learning-based normal estimation methods has been proposed [32–35]. Zhou et al. [36] offer a multi-scale neighborhood selection technique and an additional plane feature constraint based on PCPNet [32] to enhance performance. By leveraging both the PointNet and 3DCNN, Hashimoto et al. [37] proposed a joint network that can accurately infer normal vectors from a point cloud. Zhou et al. [38] proposed a normal filter based on multipatch stitching. Thanks to their patch-level architecture, their method can reduce computational costs and improve the robustness of noise removal. Boulch et al. [39] proposed to convert a local point cloud block into a 2D Hough space accumulator by randomly selecting a point triplet and voting for the normals in that plane. Then, the normals are estimated from the accumulator as a continuous estimate of the regression problem. This method does not take full advantage of the 3D information, as it loses information in the transformation phase. Later, a mixture-of-experts (MOE) architecture called Nesti-Net was introduced by Ben-Shabat et al. [40] and relies on a data-driven methodology to determine the ideal scale around each point and boosts sub-network specialization. Usually, such methods are suitable to models with many curvature details

and/or high noises, and many methods are still limited in their ability to handle piecewise smooth surfaces and preserve sharp features.

3. Overview

Given a point cloud $P = \{p_i\}_{i=1}^{n}$ as input, our method takes three steps to estimate the normals. (1) The nearest neighborhood N with S neighbors is obtained for each point p_i, and initial normals are referenced for NERL. Afterward, point cloud P is classified into two types: feature point set P_f or non-feature point set P_n, according to feature coefficients of the points, which are detailed in Section 4. (2) Different neighborhood $\widetilde{N}$ is selected for each feature point. After that, Gaussian map and clustering are utilized to form different clusters $\{\overline{N}_1, \overline{N}_2, ..., \overline{N}_K\}$, where K is the number of clusters of the current point. Each cluster is solved to a plane by the RANSAC algorithm. The residuals between the current point and each fitting plane are calculated, then the cluster that matches the current point is regarded as the optimal cluster that is employed to conduct the normal mollification, as shown in Section 5.1. (3) According to the distribution of fitting residuals, the feature points with larger residuals can be considered abnormal points. Then the second-stage normal mollification is carried out for these points, which is explained in Section 5.2. The overview of our method is concluded in Algorithm 1 and illustrated in Figure 1.

Algorithm 1 The pipeline of our algorithm

Input: Point cloud P;
Output: Normal set $\{n_i\}$;
1: **for** $p_i \in P$ **do**
2: Compute initial normal $\widetilde{n}_i \in p_i$, feature coefficient w_i;
3: **if** $w_i < w_t$ **then**
4: $n_i = \widetilde{n}_i$;
5: **else**
6: Obtain mollification neighborhood $\overline{N}_i \in p_i$ via Algorithm 2;
7: Obtain n_i via the first-stage normal mollification based on Equation (7);
8: **if** fitting residual $r(p_i) > \varepsilon_f$ **then**
9: Obtain mollification neighborhood $N_i \in p_i$ via Section 5.2;
10: Obtain n_i via the second-stage normal mollification based on Equation (8);
11: **end for**

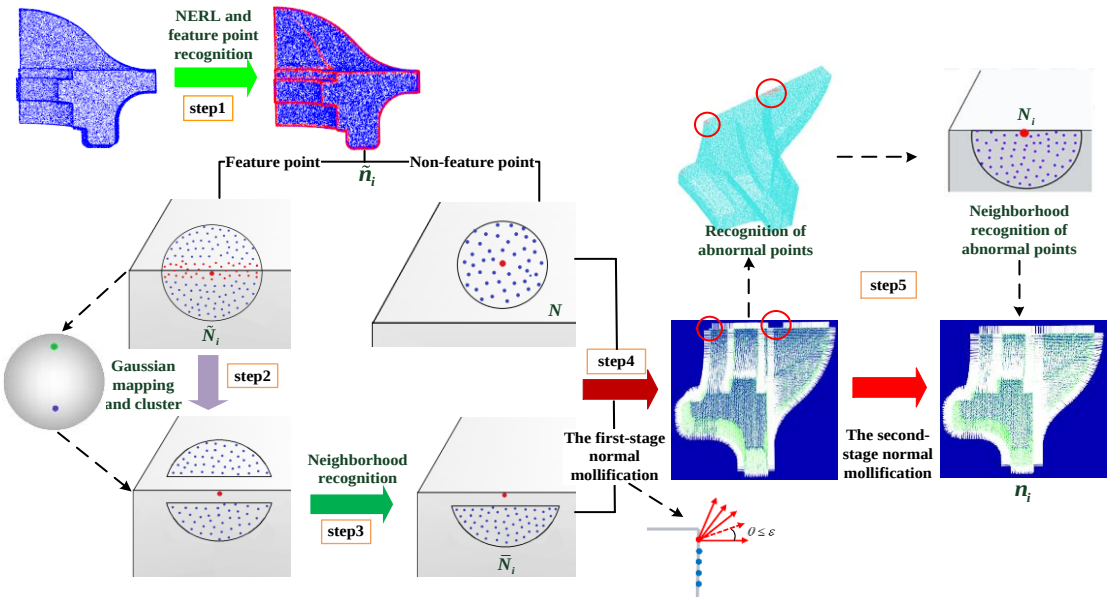

Figure 1. The pipeline of our algorithm (***N*** represents the neighborhood of each stage, and ***n*** denotes the normal of the stage).

4. Normal Initialization and Feature Point Recognition

In order to make the initial normals of smooth areas have a certain accuracy, as well as preparation for the subsequent processing, based on the weighted principal component analysis (WPCA) [13], our NERL takes advantage of M-estimates of location [41] instead of the mean to estimate the normals. For each point p_i and its S neighbors, a covariance matrix C is calculated as

$$\begin{aligned} C &= \frac{1}{\sum w_d(p_j,p_i)} \sum_{j=1}^{S} w_d(p_j,p_i) \cdot (p_j - p_r)(p_j - p_r)^T \\ p_r &= \frac{\sum_{j=1}^{S} w_d(p_j,p_i) \cdot p_j}{\sum_{j=1}^{S} w_d(p_j,p_i)}, \end{aligned} \tag{1}$$

where w_d and p_r are a distance weight and points of robust location, respectively, and $w_d(p_j,p_i) = \exp(-||p_j - p_i||_2/2 \cdot {\sigma_d}^2)$. σ_d is set to the average distance from its ten nearest neighbors of the current point to their one respective neighbor. The estimated normal at p_i is the eigenvector associated with the smallest eigenvalue of the covariance matrix. Figure 2 shows the normal estimation errors of PCA, WPCA, and NERL on the Harpago and Scallop curved surface models.

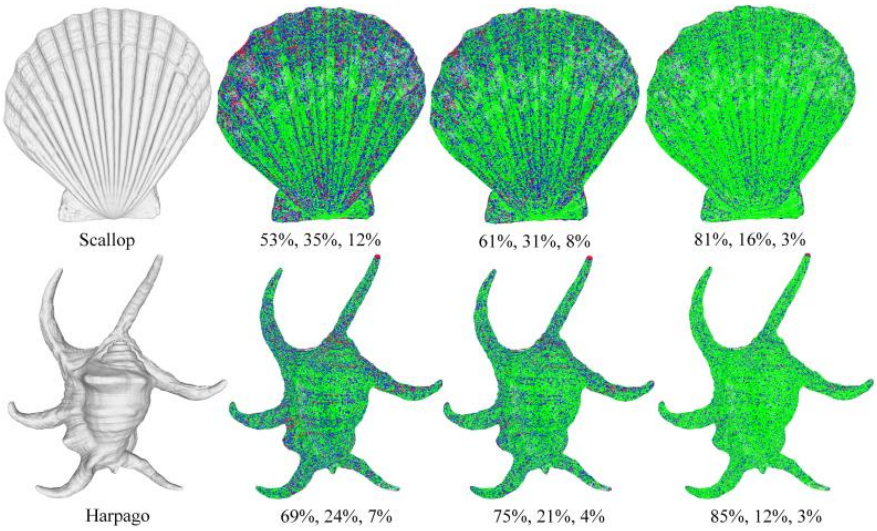

Figure 2. The normal estimation error. Green denotes points with an estimated normal that deviates by less than 5° from the ground-truth normal; blue encodes angular deviations between 5° and 10°; and red marks errors more than 10°. From left to right are the model, PCA, WPCA, and NERL methods, and the percentages of green, blue, and red points are counted under each result.

The normals of the points in the smooth regions are reliably computed through NERL, whereas those of the points near the edges still fail to preserve sharp features. Therefore, it is indispensable to distinguish the feature points further. In order to make feature detection more applicable, we divided it into two cases according to the noise scale to deal with separately. Under the condition of small-scale noise, for each point p_i, the eigenvalues of its covariance matrix computed by PCA and NERL methods can be obtained by singular value decomposition, and the feature coefficients w_{i1}, w_{i2} are calculated by their three eigenvalues, respectively, corresponding to $(\lambda_0, \lambda_1, \lambda_2)$, $(\lambda'_0, \lambda'_1, \lambda'_2)$ with ascending order.

$$w_{i1} = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}, w_{i2} = \frac{\lambda'_0}{\lambda'_0 + \lambda'_1 + \lambda'_2}. \tag{2}$$

Feature points P_f are differentiated via the given threshold w_{t1}, w_{t2}, i.e., $P_f = \{p_i \in P | (w_{i1} > w_{t1}) \cup (w_{i2} > w_{t2})\}$. Although the above method has good efficiency,

it will become invalid under the situation of large-scale noise, as illustrated by Figure 3. Accordingly, a robust method based on normal differences is proposed. For a given point p_i with the neighborhood N_{p_i} of S neighbors, a covariance matrix U is constructed by the normal difference of point pairs in the neighborhood:

$$U = \frac{1}{S} \sum_{n_j, n_k \in N_{p_i}} (n_j - n_k)(n_j - n_k)^T, \tag{3}$$

where n_j and n_k are the normals of the neighborhood points. The size of the three eigenvalues of the covariance matrix U reflects the dispersion degree of the normals in the neighborhood. For smooth regions, the three eigenvalues tend to be zero, and as the enlargement of dispersion degree, the size of the eigenvalues increases accordingly. Thus, for a point p_i, its feature coefficient w_{i3} is defined as the sum of three eigenvalues. We still specify a threshold w_{t3}, and feature points are denoted as $P_f = \{p_i \in P | (w_{i3} > w_{t3}\}$. Figure 3 indicates that our method can still perform accurate feature detection even if under large-scale noise. Note that in practical implementation, there is no strict standard for the selection of both, which depends on the prior cognition of the processing object.

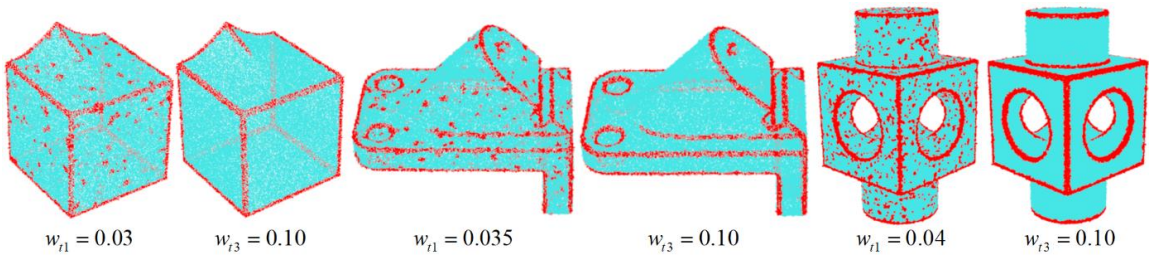

Figure 3. Feature detection results by common method [11] and our method under large-scale noise. Left: smooth-feature model with 0.25% Gaussian noise. Middle: anchor model with 0.30% Gaussian noise. Right: block model with 0.30% Gaussian noise.

5. Robust Normal Estimation

In this section, we explain how to reliably estimate the normals of the distinguished feature points via the normal mollification methodology, which divides into two successive steps: a preliminary normal estimation based on neighborhood segmentation and a further normal estimation based on residual under various difficult scenarios.

5.1. Normal Mollification Based on Neighborhood Segmentation

Gaussian map and clustering: The Gaussian map of the point cloud is used to map the normals of the surface in Euclidean space to the unit sphere. Based on the fact that the normals of points change continuously in the smooth regions but alter dramatically when located at the sharp surface, the normals of smooth areas will form a single cluster, while those near the sharp feature will form multiple clusters on the Gaussian sphere and each cluster corresponds to different surfaces, as shown in Figure 4. Since the fitting-based approaches generally assume that the surface is smooth everywhere and the estimated normals are continuous at sharp edges, which makes the threshold employed for subsequent clusters unreliable, we temporarily remove the feature points from the neighborhood in the first-stage normal mollification to form a single cluster for each patch on the Gaussian sphere. Therefore, for each feature point $p_i \in P_f$, neighborhood $\widetilde{N}$ of $\widetilde{S}$ neighbors without feature points will be selected.

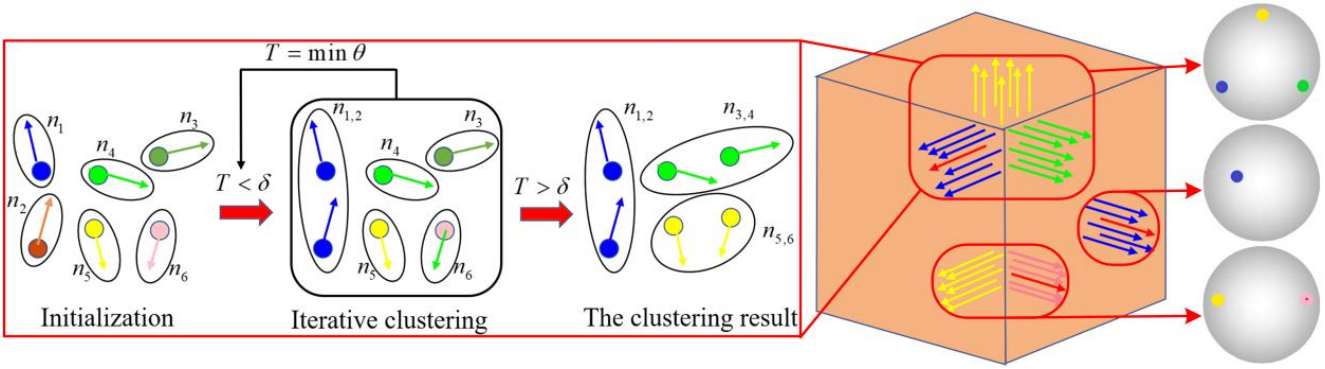

Figure 4. Gaussian map and clustering of cube.

The dataset of the neighborhood of the feature point $\widetilde{N} = \left\{p_i, i = 1, 2..., \widetilde{S}\right\}$ is mapped to the Gaussian sphere, and the mapped dataset can be expressed as $G(\widetilde{N}) = \left\{n_i, i = 1, 2..., \widetilde{S}\right\}$, where n_i is the corresponding normal. Since the amount of sub-neighborhood is not foreknown, the parameter-free clustering algorithm is adopted. Thus, here, we select the parameterless hierarchical agglomerative clustering (HAC), where the bottom-up clustering procedure is also appropriate for this process. In this algorithm, each data point is regarded as a separate cluster beforehand. Then, according to the similarity between clusters, they are gradually merged into increasing clusters until the similarity between any two clusters exceeds a specified threshold.

Calculating the similarity between two clusters, $\widetilde{N}_1$ and $\widetilde{N}_2$, is also termed the linkage criterion, which includes three categories: single linkage, complete linkage, and average linkage. The common average linkage shows the best performance in our application:

$$D_{ave}(\widetilde{N}_1, \widetilde{N}_2) = \frac{1}{\left|\widetilde{N}_1\right| \cdot \left|\widetilde{N}_2\right|} \sum_{i \in \widetilde{N}_1} \sum_{j \in \widetilde{N}_2} d(i, j), \tag{4}$$

where $\left|\widetilde{N}_1\right|$, $\left|\widetilde{N}_2\right|$ represent the number of points in clusters $\widetilde{N}_1, \widetilde{N}_2$, respectively; i and j denote i-th and j-th points in the clusters $\widetilde{N}_1, \widetilde{N}_2$; $d(i, j)$ is expressed as the similarity between two points; and

$$d(i, j) = \arccos(n_i \cdot n_j), \tag{5}$$

where $n_i \cdot n_j$ is the inner product of two unit normals. On the Gaussian sphere, they will be merged when the similarity between two clusters is less than a given threshold δ.

An appropriate threshold δ is key to obtaining credible clustering results. A small threshold can result in the isotropic neighborhood being divided into multiple clusters. Inversely, a relatively large threshold makes the anisotropic neighborhood group into an identical cluster. However, on account of the feature points being removed in advance, the normals of different patches have obvious recognizability; thus, the appropriate threshold is in a relatively large tolerance. After considerable tests, the threshold of 7° performs good outcomes in this algorithm.

Neighborhood recognition: The dataset in the neighborhood of feature points $\widetilde{N} = \left\{p_i, i = 1, 2..., \widetilde{S}\right\}$ is grouped into several isotropic clusters $\left\{\overline{N}_1, \overline{N}_2, ..., \overline{N}_K\right\}$ by using a Gaussian map and clustering, where K is the number of clusters. To eliminate the influence of noise and outliers, the RANSAC algorithm is utilized for each cluster. Then the residuals between the current point and each fitting plane are obtained, and the cluster corresponding to the minimum residual is selected as the mollification neighborhood of the current point. The cluster and neighborhood recognition algorithms are shown in Algorithm 2.

Algorithm 2 Neighborhood recognition of feature points

Input: $p_i, p_i \in P_F$;
Output: p_i's mollification neighborhood $\overline{N}_i$;
1: **for** $i = 0$ to P_F.size() **do**
2: Seek $\widetilde{S}$ nearest neighborhood $\widetilde{N}$;
3: View every point $\left\{p_1, p_2, ...p_{\widetilde{S}}\right\}$ as a single cluster $\left\{\widetilde{N}_1, \widetilde{N}_2, ..., \widetilde{N}_{\widetilde{S}}\right\}$;
//see Section 5.1 and Figure 4
4: **for** $j = 1$ to $\widetilde{S}$ **do**
5: Compute similarity: $D_{ave}(\widetilde{N}_p, \widetilde{N}_q), \quad p \neq q$ based on Equation (4);
6: $T = \min\left\{D_{ave}(\widetilde{N}_p, \widetilde{N}_q)\right\}$;
7: **if** $(T < \delta)$ **then**
8: Find two cluster $\widetilde{N}_m, \quad \widetilde{N}_n$ with minimum similarity;
9: Merge $(\widetilde{N}_m, \widetilde{N}_n)$;
10: **else**
11: Obtain the ultimate cluster result: $\{\overline{N}_1, \overline{N}_2, ..., \overline{N}_K\}$;
12: **break**;
13: **end For**
14: $\{\varepsilon_1, \varepsilon_2, ..., \varepsilon_S\}$ =computeResidual $(p_i, \{\overline{N}_1, \overline{N}_2, ..., \overline{N}_S\})$; //see Section 5.2
15: $\overline{N}_i = \min\{\varepsilon_1, \varepsilon_2, ..., \varepsilon_S\}$corresponding to $\overline{N}_c, c \in \{1, 2, ..., S\}$;
16: **end for**
17: **return** $\overline{N}_i$ with $\overline{S}_i$ neighbors;

Mollification: The isotropic sub-neighborhood matching the current point is employed to mollify its normal. Similar to previous studies [5], our iteration formula of the first-stage normal mollification is given by Equation (6).

$$n_i^{k+1} = \frac{\sum_{j=1}^{\overline{S}_i} w_n(n_i^k, n_j) n_j}{\sum_{j=1}^{\overline{S}_i} w_n(n_i^k, n_j)}, \tag{6}$$

where n_j is the normal of a point in the mollification neighborhood $\overline{N}_i$ of the current point p_i, n_i^k is normal of p_i after the k-th mollification, and $w_n(n_i^k, n_j)$ is a normal deviation kernel function as shown in Equation (7).

As the deviation between the normal of the k-th iteration and the normal of the $(k-1)$-th iteration is less than the threshold ε, the iteration will be terminated, i.e., $1 - n_i^{k-1} \cdot n_i^k \leq \varepsilon$, and the threshold $\varepsilon = 10^{-4}$ is set.

$$w_n(n_i^k, n_j) = \exp\left(\frac{-\left\|n_i^k - n_j\right\|^2}{\sigma_n^2}\right), \tag{7}$$

where $||\cdot||$ is the $l_2 -$ norm, and σ_n is the normal deviation bandwidth. To make the normal mollification more accurate, the selection of this parameter should reduce the influence of outlier normals as much as possible in the normal mollification process. In the practical utilization, σ_n is set to the maximum normal deviation, i.e., $\sigma_n = \max(\left\|n_i - n_j\right\|), \quad j \in [1, \overline{S}_i]$.

5.2. Normal Mollification Based on Residual

Detection and neighborhood recognition of abnormal points: Due to the existence of noise, non-uniform sampling, or various complications, some non-feature points are over-identified as feature points, such as sharp corners, narrow-bands, and low-sampling density regions, so the normals in some points may be incorrectly estimated.

To cope with this kind of issue, as well as compensate for the impact of excluded feature points in the first stage, the second-stage normal mollification is devised. First of all,

the residual set $\left\{r(P_f)\right\}$ between each feature point and the consistent optimal plane can be calculated, and abnormal points P_a are defined as $P_a = \left\{ p_i \in P_f \middle| r(p_i) > \varepsilon_f \right\}, \varepsilon_f = \overline{\mu} + 3\sigma$, where $\overline{\mu}$ and σ are the mean and standard deviation of the residual set, respectively. Note that it is impossible for the recognition to be 100% precise. However, excessive identification only increases the running time of the method and does not degrade the results.

Next, similar to the first stage, we need to seek the mollification neighborhood of the abnormal points for normal readjustment. Therefore, a neighborhood with a larger size $S^* = 6\widetilde{S}$ is selected. Since the normals of most feature points are correctly estimated in the first stage, they would be involved when constructing the second-stage mollification neighborhood. Given an abnormal point p_i and its neighbor p_{ij}, we define the residual $r_{ij} = \left| n_{ij} \cdot (p_{ij} - p_i) \right|$. The mollification neighborhood N_i is constituted by selecting neighbors if r_{ij} is less than the specified threshold σ_τ, which is set to 0.25 degrees, corresponding to the amplitude in this article.

Mollification: After the abnormal points' mollification neighborhood is sought, the second-stage normal mollification is designed to estimate the normals of these points more finely. The formula of the second-stage normal mollification is given by Equation (8).

$$n_i = \frac{\sum\limits_{p_j \in N_i} w_s(r_{ij}) n_{ij}}{\sum\limits_{p_j \in N_i} w_s(r_{ij})}, \tag{8}$$

where $w_s(r_{ij})$ is a residual kernel function defined as $w_s(r_{ij}) = \exp(-(r_{ij}/\sigma_s)^2)$, and σ_s is set as $\sigma_\tau/3$.

6. Implementation Results

To evaluate the performance of the proposed method, a variety of point cloud models of sharp features with a complicated neighborhood, non-uniform sampling, and noises are tested. We contrast our method with some state-of-the-art approaches: PCA [3], BF [4], PCPNet [32], DeepFit [33], MTRNet [34], RNE [26], HF [27] (only HF_cubes is involved), IterWPCA [18], and PCV [25]. The learning-based methods used for comparison are trained via the PCPNet shape dataset [30]. For the sake of fair comparisons, the same neighborhood size is applied to each algorithm, and all other parameters required by other methods, if any, are set to the default or the values of the best results obtained through trial and error. In addition, the parameters of our algorithm are summarized below:

- $S, \widetilde{S}, S^*$ are the number of neighbors for calculating the initial normal, the first-stage, and the second-stage mollification, respectively ($\widetilde{S} = 2S, S^* = 6\widetilde{S}$).
- w_{t1}, w_{t2} are the thresholds used to distinguish feature points under low-level noise.
- w_{t3} is the threshold used to distinguish feature points under high-level noise.
- δ is the threshold for clustering ($\delta = 7^\circ$).
- σ_n, σ_s are the normal deviation and the residual bandwidth for the first-stage and the second-stage mollification, respectively.

In this paper, three different scores are used to evaluate the performance of the algorithms quantitatively: (1) the runtime; (2) the root mean square with a threshold (RMS_τ); and (3) the number of bad points (NBPs), where (2) and (3) are defined as

$$\mathrm{RMS}_\tau = \sqrt{\frac{1}{|P|} \sum_{p \in P} f(n_p, \hat{\widetilde{n}}_p)^2}, \tag{9}$$

where

$$f(n_p, \hat{\widetilde{n}}_p) = \begin{cases} n_p \hat{\widetilde{n}}_p, if \quad n_p \hat{\widetilde{n}}_p < \tau \\ \pi/2, \quad otherwise \end{cases}, \tag{10}$$

n_p is the ground-truth normal of p, and $\widetilde{n}_p$ is the estimated normal of p. $\widehat{n_p \widetilde{n}_p}$ is the angle between n_p and $\widetilde{n}_p$. $\tau = 10^\circ$ is set, which means those points whose errors are more than 10° are regarded as bad points.

All the noise used in the experiments is Gaussian noise, with different standard deviations as a percentage of the bounding box diagonal. Our method is implemented using Microsoft Visual Studio 2015 with C++ and Point Cloud Library (PCL) [42]. In the specific test process of our algorithm, the parameters $\widetilde{S} = 2S$, $w_{t1} = 0.02$, $w_{t2} = 0.012$, and $w_{t3} = 0.2$ (note that the neighborhood size below refers to $\widetilde{S}$) are the default unless otherwise specified. All the experiments have been performed on the same computer with Inter(R) Core(TM) i3-7100 3.90 GHZ and 16 GB RAM without parallel computing.

6.1. Quantitative Evaluation

6.1.1. Accuracy

To evaluate the effectiveness of our method under various complex conditions, three basic categories are used for performance testing, as shown in Figure 5. Here, the first category is used to evaluate performance on sharp features, the second is a test of robustness to non-uniform samplings, and the third is utilized to assess the ability to preserve small structures. For each model, the neighborhood size is defined as 40 points. The second recognition strategy only is used in Detials in our method. Additionally, for PCPNet and MTRNet, we adopt the multi-scale versions provided by authors with the default parameters. DeepFit does not have a multi-scale version, so the 256 neighborhood points and 3-order jet recommended by the author are adopted. The results for each model are presented in Figure 6 (the errors of Pipe and Icosahedron are the average of the test results of gradient and striped anisotropic samplings).

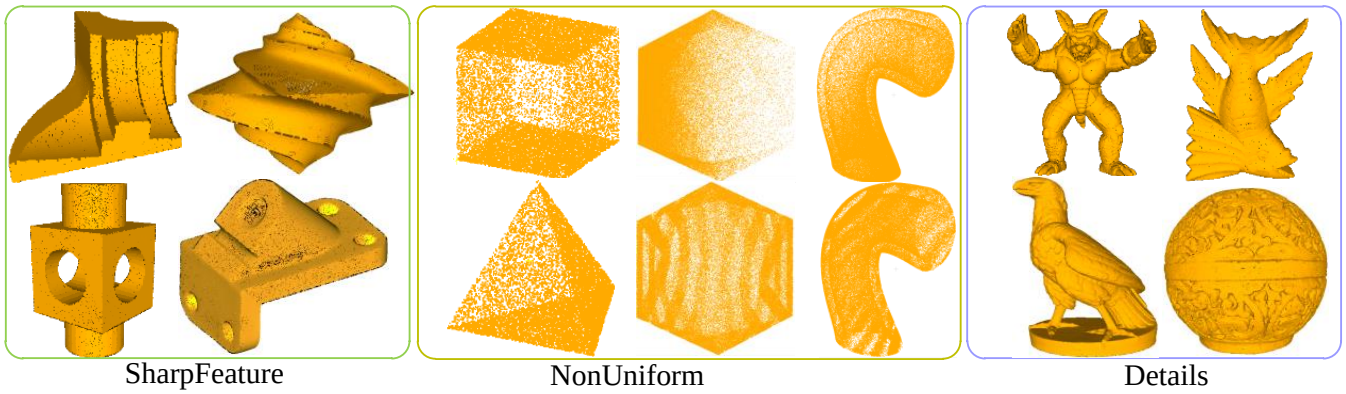

Figure 5. The point cloud models used to test the experiment.

Overall, PCA generates overly smooth normals near sharp edges, which can be markedly seen from the NBPs of the first two categories of models in Figure 7, but it has a slight improvement in the processing of more curved models. PCPNet obtains the worst result due to its poor accuracy on smooth regions. Furthermore, DeepFit, as a deep-fitting method, underperforms in the face of sharp features. In contrast, MTRNet obtains better results, especially for detecting details. Although BF and IterWPCA maintain sharp features to a certain extent, there are still considerable bad points, as presented in Figure 7. HF and RNE have almost similar results, and RNE is inferior to HF in the Details module, whereas it has some advantages in anisotropic sampling. PCV achieves good outcomes in handling the above cases, but in comparison, our method takes on lower and lower computational costs (see Section 6.3).

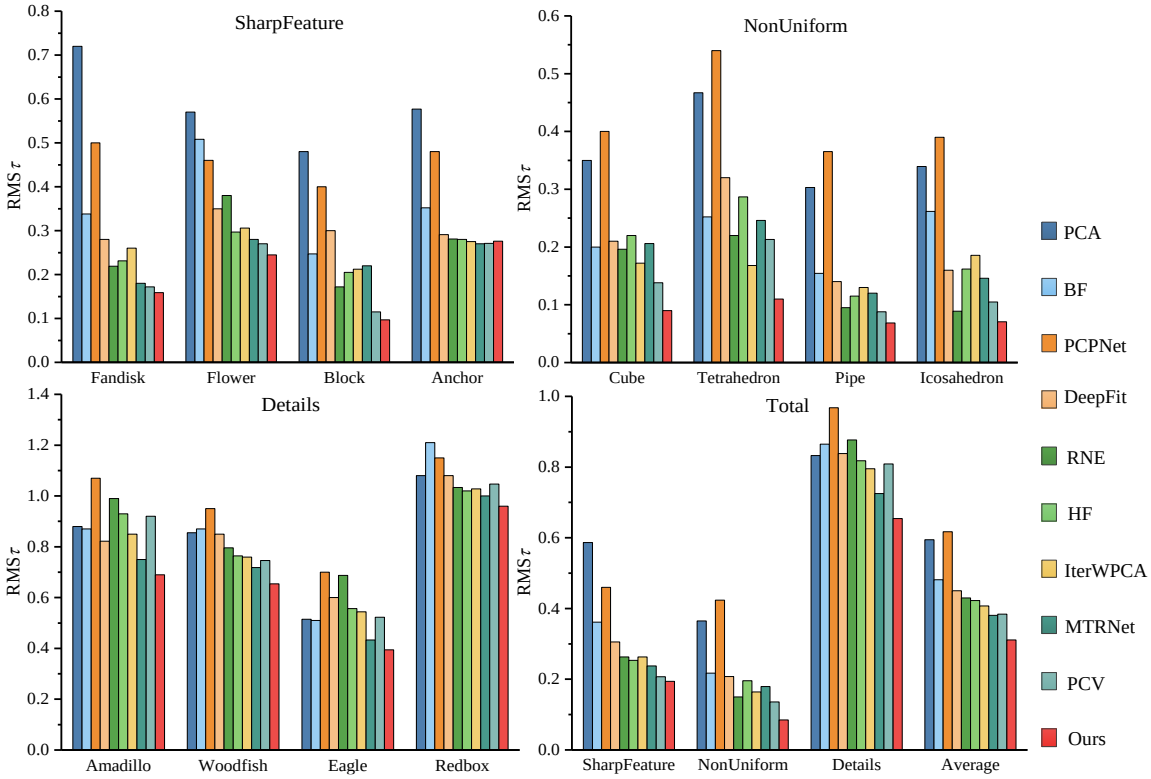

Figure 6. The RMS_τ estimated by various methods, where the first three respectively represent the angular error of sharp feature models, non-uniform sampling models, and detail models; the comparison of the comprehensive performance of each algorithm in each category is demonstrated at the end, in which the average errors of each category and all models are included.

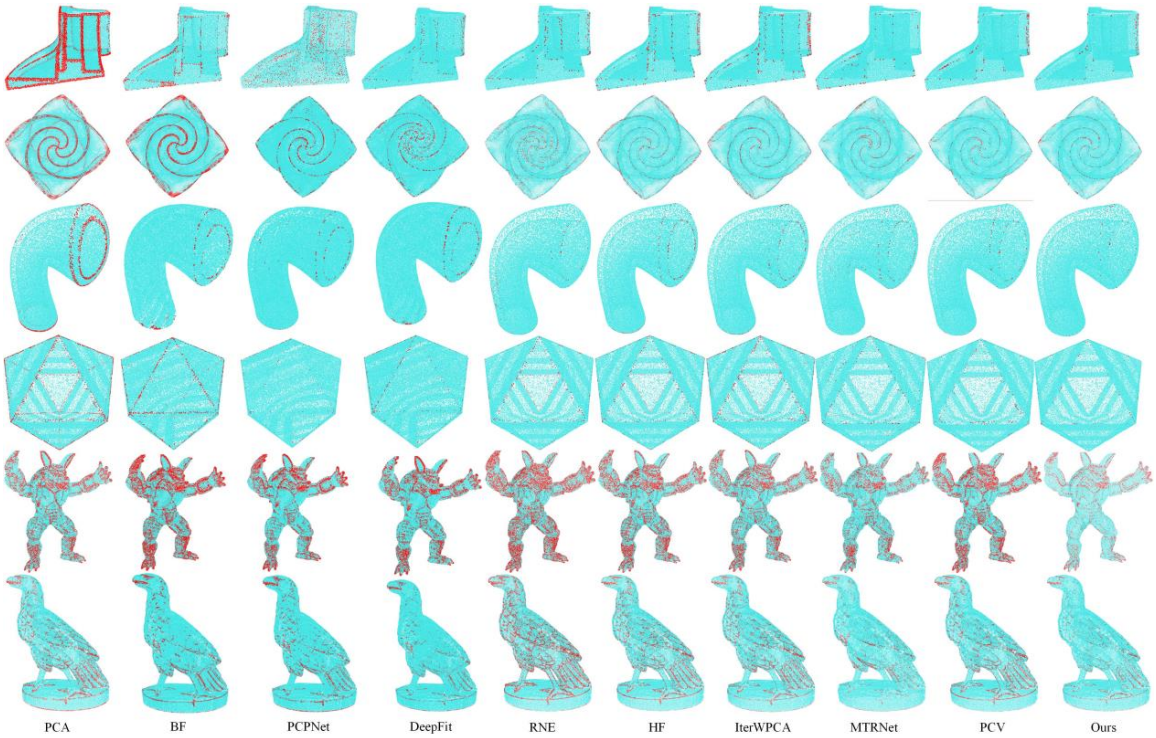

Figure 7. Visualization of NBPs on partial models of SharpFeature, NonUniform, and Details.

6.1.2. Robustness to Noise

In this experiment, two models of Cube and Lagera with variational Gaussian noise levels are tested, where the Cube model focuses on relatively small noise levels varying from 0% to 1.5% with an interval of 0.25% and the second recognition strategy is used when the noise level reaches 0.5%; the Lagera noise data from the PCPNet dataset centers on relatively large noise levels, which contains noise varying from 0% to 10% with an interval of 2%, so the second recognition strategy is used, and w_{t3} is set as 0.6 when the noise level exceeds 6%. The neighborhood size is defined by 60 points for each model. The statistical results and the visualization of the angular error are presented in Figures 8 and 9, respectively. Compared with other methods, due to noise affecting the stability of voting results, HF has bad behavior under different noises of the two models. The accuracy of the normal estimated by MTRNet in the Lagera model containing curved surfaces is well guaranteed with the increase in noise level, whereas on account of incorrect estimation of normals at sharp features and noisy normal, as shown in Figure 9, it has a high RMS_τ in the Cube model. RNE has great contrast between models with different features. In comparison, it is bad at large noises and curved models. The errors of IterWPCA and PCV have moderate results but a great amplitude rise with the increase in noise level. The proposed method achieves the lowest RMS_τ, which indicates that our method is more advantageous in robustness to noise.

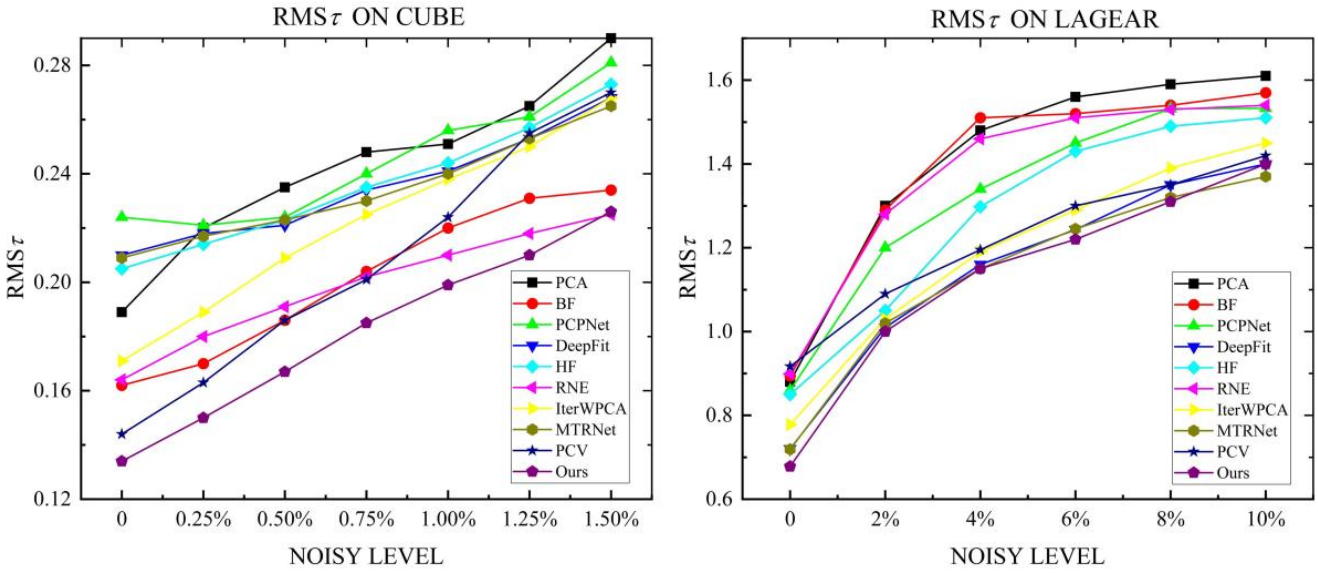

Figure 8. Comparison of the robustness of different algorithms to noise on Cube models with small noise levels and Lagera models with large noise levels.

6.2. Qualitative Evaluation

We further demonstrate the performance of our method in point cloud filtering applications. In the experiment phase, we utilize an effective point updating algorithm [1,43], a point cloud smoothing algorithm Bilateral method [44], and EGT [45], in which the normal information used in this algorithm is matched by the estimated normal output via each method. To obtain better results in the experiment, we alternately call the normal estimation and position update for several iterations. Because the excessive implementation of this algorithm could smooth out sharp edges, the number of iterations is empirically set to 4. The Joint model with 48K points and 1.0% noise is used in our experiment, where except for the first iteration, the first recognition strategy is used in the remaining iterations, and the results are shown in Figure 10. We can see that our method has higher fidelity in preserving features than other methods.

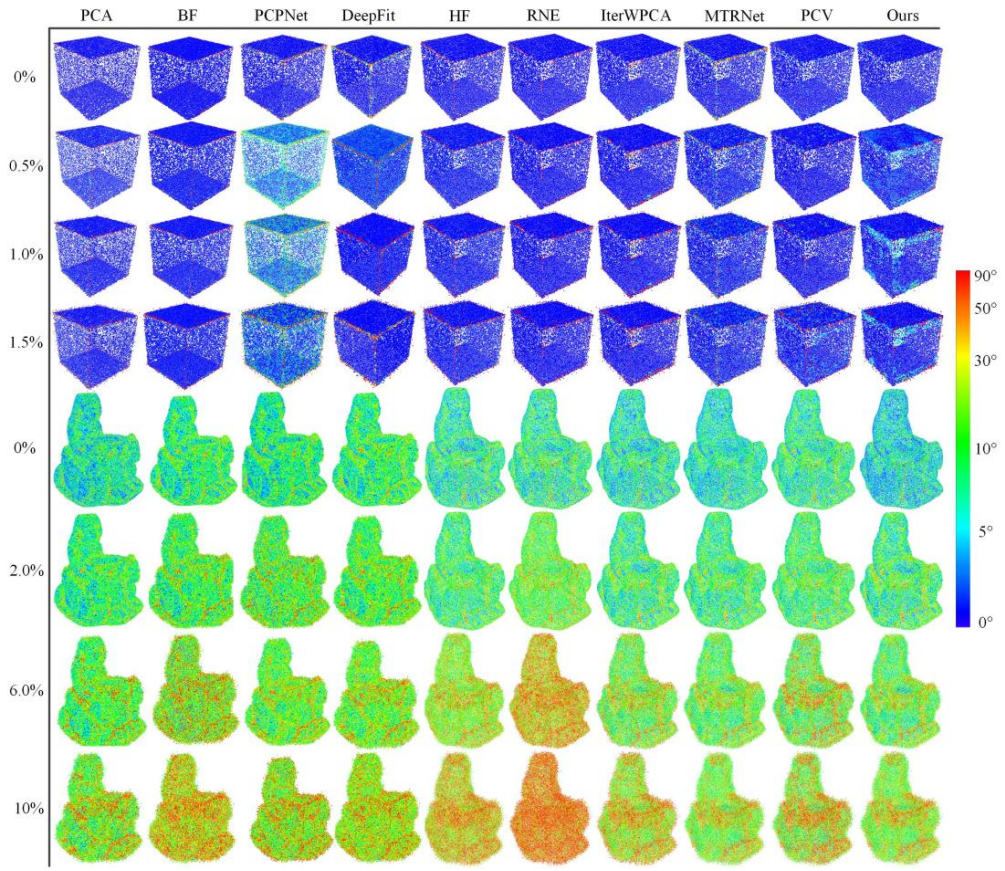

Figure 9. Visualization of angular error evaluated in Cube and Lagera models in the experiment of robustness to noise.

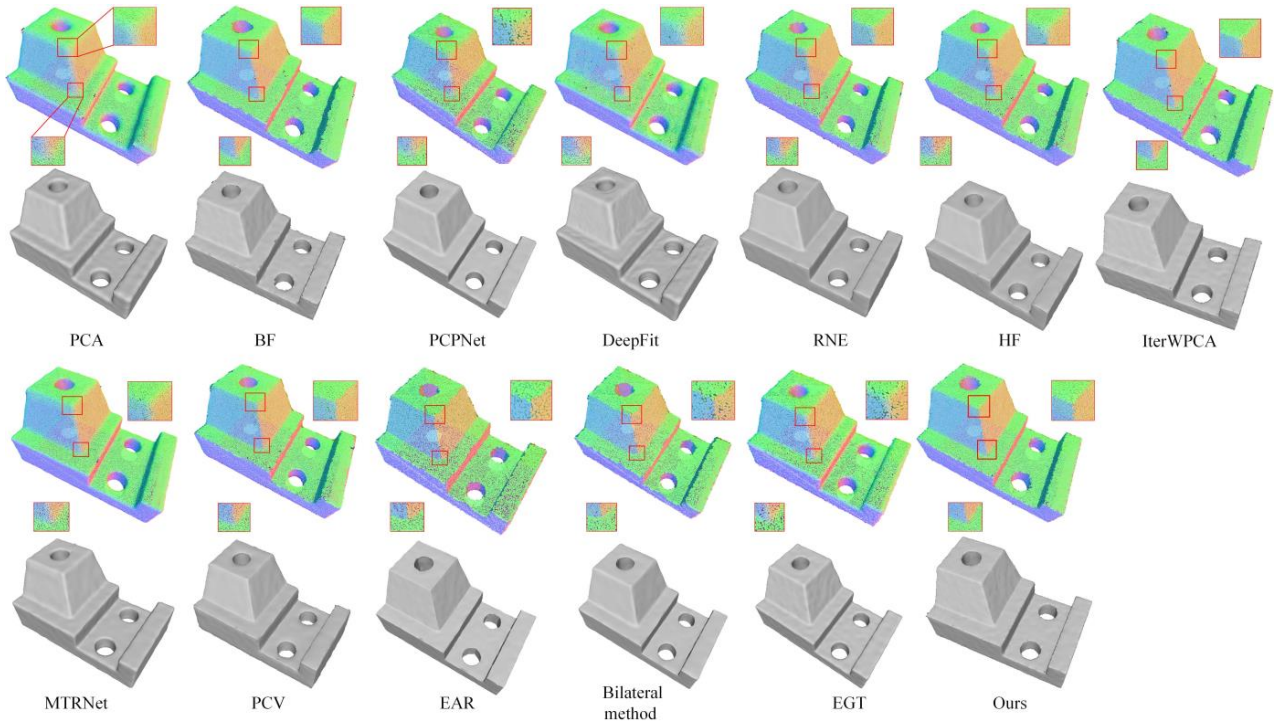

Figure 10. Comparison on point cloud filtering. First row: filtering results with upsampling. Second row: corresponding to surface reconstruction results.

Real scan data often has many defects, such as noise, outliers, and non-uniform sampling. In order to illustrate the effectiveness of our method on real scan data, we tested our method on the Pyramid and Office models and compared them with PCA, as shown in Figure 11, where the first recognition strategy is adopted in our method. As we can see, the PCA method results in the loss of the sharp edges and corners, but the sharp features are favorably preserved by our method.

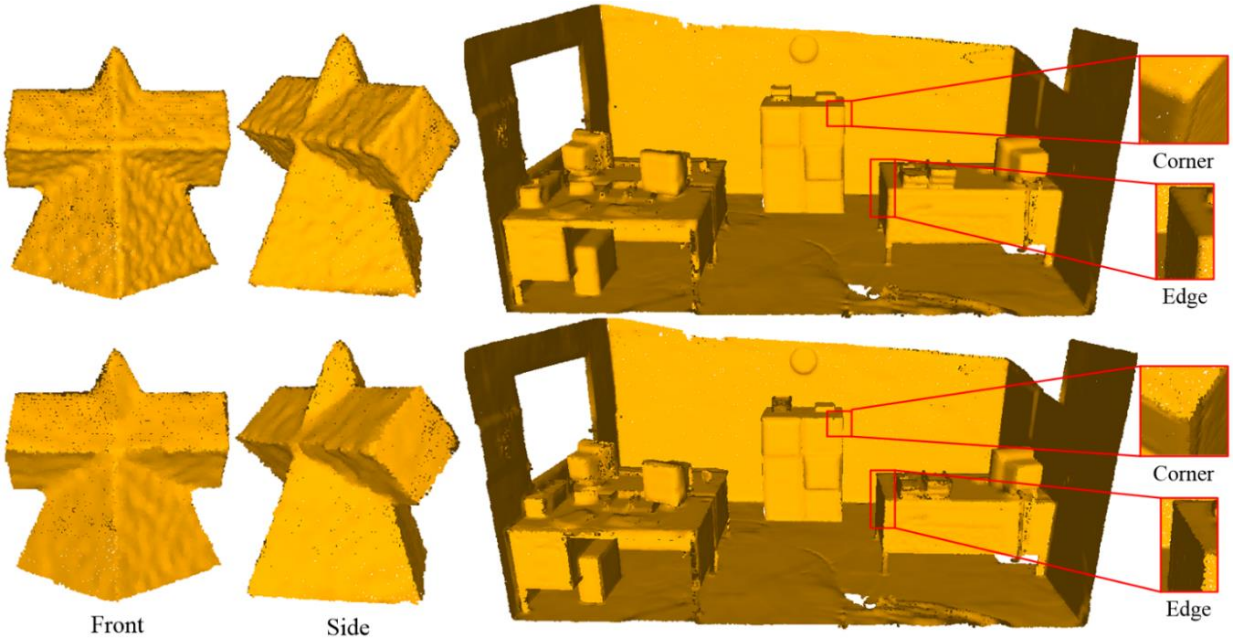

Figure 11. The normal estimation results on real scan data. The result of PCA is shown in the top row, and the result of the proposed method is shown in the bottom row.

6.3. Computational Time

We present the time consumption of all algorithms applied to the models in Figure 5 in Table 1. It can be seen that PCA and BF are the fastest two methods, whereas BF has a larger improvement in error results. Due to the voting strategy of point pairs, PCV is the slowest method, but it has a better effect. Moreover, learning-based approaches take relatively more time. Our method is slightly faster than RNE and HF, and the accuracy has been significantly improved. Moreover, it is worth noting that the proposed method can be fully parallelized easily, which will remarkably reduce the computational time taken by our method.

Table 1. Mean computational time (in seconds) of different algorithms.

	PCA	BF	PCPNet	DeepFit	MTRNet	RNE	HF	IterW-PCA	PCV	Ours
SharpFeature (48,068)	0.07	0.56	61	59.7	42.5	23.2	15.7	6.25	184.7	2.98
NonUniform (72,000)	0.63	0.98	90.8	78.3	64.5	42.55	17	14.1	296.7	4.34
Details (189,021)	1.43	3.82	249.5	208.2	161.6	84.5	76	22.3	523.7	32.5
Average	0.71	1.79	133.8	115.4	89.5	50.1	36.2	21.3	335	13.3

7. Conclusions

In this article, we present a robust normal estimation method for point clouds with sharp features, where via a normal estimator of robust location and a robust feature point recognition, reliable initial normals of smooth regions and accurate recognition of points close to sharp regions can be obtained. Further, the two-stage normal mollification based on neighborhood recognition ensures our algorithm can deal with various difficult conditions. We demonstrate the validity of our work by testing on both synthetic and scanned point

clouds. Compared with other algorithms, the proposed method has higher quality, lower computational cost, and robustness to noise and non-uniform sampling. Moreover, it is simple and easy to implement.

However, more faithful normals can be generated with delicate parameters turning. In the future, we would like to choose these parameters adaptively according to various noises and sampling densities.

Author Contributions: Conceptualization, G.L. and X.L.; methodology, G.L.; software, X.L.; validation, S.S. and W.Y.; writing—original draft preparation, G.L.; writing—review and editing, X.L.; visualization, S.S.; funding acquisition, W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (Grant No. 52175031), Sichuan Science and Technology Program (Grant No. 2021YFN0021), Sichuan Province Information Application Support Software Engineering Technology Research Center Open Project (Grant No. 2021RJGC-Z01), and Sichuan Provincial Science and Technology Innovation (Miao Zi Project) (Grant No. 2022103).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Our sincere thanks also go to the Aim@Shape project, the Stanford Computer Graphic Laboratory, and Three D Scans for providing the models in the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, Z.; Xiao, X.; Zhong, S.; Wang, W.; Li, Y.; Zhang, L. A feature-preserving framework for point cloud denoising. *Comput. Aided Des.* **2020**, *127*, 102857. [CrossRef]
2. Lu, D.; Lu, X.; Sun, Y.; Wang, J. Deep feature-preserving normal estimation for point cloud filtering. *Comput. Aided Des.* **2020**, *125*, 102860. [CrossRef]
3. Hoppe, H.; DeRose, T.; Duchampt, T.; McDonald, J.; Stuetzle, W. Surface reconstruction from unorganized points. In Proceedings of the 19th Annual ACM Conference on Computer Graphics and Interactive Techniques, Chicago, IL, USA, 26–31 July 1992; pp. 71–78.
4. Öztireli, A.C.; Guennebaud, G.; Gross, M. Feature preserving point set surfaces based on non-linear kernel regression. *Comput. Graph Forum* **2009**, *28*, 493–501. [CrossRef]
5. Huang, H.; Wu, S.; Gong, M.; Cohen-Or, D.; Ascher, U.; Zhang, H. Edge-aware point set resampling. *ACM Trans. Graph.* **2013**, *32*, 1–12. [CrossRef]
6. Huang, H.; Li, D.; Zhang, H.; Ascher, U.; Cohen-Or, D. Consolidation of unorganized point clouds for surface reconstruction. *ACM Trans. Graph.* **2009**, *28*, 1–7. [CrossRef]
7. Demarsin, K.; Vanderstraeten, D.; Volodine, T.; Roose, D. Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Comput. Aided Des.* **2007**, *39*, 276–283. [CrossRef]
8. Pauly, M.; Keiser, R.; Kobbelt, L.P.; Gross, M. Shape modeling with point-sampled geometry. *ACM Trans. Graph.* **2003**, *22*, 641–650. [CrossRef]
9. Jones, T.R.; Durand, F.; Zwicker, M. Normal improvement for point rendering. *IEEE Comput. Graph. Appl.* **2004**, *24*, 53–56. [CrossRef]
10. Fleishman, S.; Cohen-Or, D.; Silva, C.T. Robust moving least-squares fitting with sharp features. *ACM Trans. Graph.* **2005**, *24*, 544–552. [CrossRef]
11. Zhang, J.; Cao, J.; Liu, X.; Wang, J.; Liu, J.; Shi, X. Point cloud normal estimation via low-rank subspace clustering. *Comput. Graph.* **2013**, *37*, 697–706. [CrossRef]
12. Mitra, N.J.; Nguyen, A. Estimating surface normals in noisy point cloud data. In Proceedings of the Nineteenth Annual Symposium on Computational Geometry, San Diego, CA, USA, 8–10 June 2003; pp. 322–328.
13. Gross, M.H.; Pfister, H. *Point-Based Graphics*; Morgan Kaufmann Publishers: Burlington, UK, 2007.
14. Cazals, F.; Pouget, M. Estimating differential quantities using polynomial fitting of osculating jets. *Comput. Aided Geom. Des.* **2005**, *22*, 121–146. [CrossRef]
15. Guennebaud, G.; Gross, M. Algebraic point set surfaces. *ACM Trans. Graph.* **2007**, *26*, 23. [CrossRef]
16. Mederos, B.; Velho, L.; Figuiredo, L.H. Robust smoothing of noisy point clouds. In Proceedings of the SIAM Conference on Geometric Design and Computing, Philadelphia, PA, USA, 2013; pp. 405–416.

17. Wang, Y.; Feng, H.Y.; Félix-étienne, D. An adaptive normal estimation method for scanned point clouds with sharp features. *Comput. Aided Des.* **2013**, *45*, 1333–1348. [CrossRef]
18. Sanchez, J.; Denis, F.; Coeurjolly, D.; Dupont, F.; Trassoudaine, L.; Checchin, P. Robust normal vector estimation in 3D point clouds through iterative principal component analysis. *ISPRS J. Photogramm. Remote Sens.* **2020**, *163*, 18–35. [CrossRef]
19. Amenta, N.; Bern, M. Surface reconstruction by Voronoi filtering. In Proceedings of the 14th Annual Symposium on Computational Geometry, Minneapolis, MN, USA, 7–10 June 1998; pp. 39–48.
20. Dey, T.K.; Goswami, S. Provable surface reconstruction from noisy samples. In Proceedings of the 20th Annual Symposium on Computational Geometry, Brooklyn, NY, USA, 4–9 June 2004; pp. 330–339.
21. Alliez, P.; Cohen-Steiner, D.; Tong, Y.; Desbrun, M. Voronoi-based variational reconstruction of unoriented point sets. In Proceedings of the 5th Eurographics Symposium on Geometry Processing, Barcelona, Spain, 4–6 July 2007; pp. 39–48.
22. Yagou, H.; Ohtake, Y.; Belyaev, A. Mesh smoothing via mean and median filtering applied to face normal. In Proceedings of Geometric Modeling and Processing, Washington, DC, USA, 10–12 July 2002; pp. 124–131.
23. Yagou, H.; Ohtake, Y.; Belyaev, A. Mesh denoising via iterative alpha-trimming and nonlinear diffusion of normals with automatic thresholding. In Proceedings of Computer Graphics International Conference, Tokyo, Japan, 9–11 July 2003; pp. 28–33.
24. Calderon, F.; Ruiz, U.; Rivera, M. Surface-normal estimation with neighborhood reorganization for 3D reconstruction. In Proceedings of the Progress in Pattern Recognition, Image Analysis and Applications, Valparaiso, Chile, 13–16 November 2007; pp. 321–330.
25. Zhang, J.; Cao, J.; Liu, X.; Chen, H.; Li, B.; Liu, L. Multi-normal estimation via pair consistency voting. *IEEE Trans. Visual Comput. Graph.* **2018**, *25*, 1077–2626. [CrossRef]
26. Li, B.; Schnabel, R.; Klein, R.; Cheng, Z.; Dang, G. Robust normal estimation for point clouds with sharp features. *Comput. Graph.* **2010**, *34*, 94–106. [CrossRef]
27. Boulch, A.; Marlet, R. Fast and robust normal estimation for point clouds with sharp features. *Comput. Graph. Forum* **2012**, *31*, 1765–1774. [CrossRef]
28. Mura, C.; Wyss, G.; Pajarola, R. Robust normal estimation in unstructured 3D point clouds by selective normal space exploration. *Vis. Comput.* **2018**, *34*, 961–971. [CrossRef]
29. Liu, X.; Zhang, J.; Cao, J. Quality point cloud normal estimation by guided least squares representation. *Comput. Graph.* **2015**, *51*, 106–116. [CrossRef]
30. Yu, Z.; Wang, T.; Guo, T.; Li, H.; Dong, J. Robust point cloud normal estimation via neighborhood reconstruction. *Adv. Mech. Eng.* **2019**, *11*, 1–19. [CrossRef]
31. Cao, J.; Chen, H.; Zhang, J.; Li, Y.; Liu, X.; Zou, C. Normal estimation via shifted neighborhood for point cloud. *J. Comput. Appl. Math.* **2017**, *329*, 57–67. [CrossRef]
32. Guerrero, P.; Kleiman, Y.; Ovsjanikov, M.; Mitra, N.J. PCPNET: Learning local shape properties from raw point clouds. *Comput. Graph. Forum* **2018**, *37*, 75–85. [CrossRef]
33. Ben-Shabat, Y.; Gould, S. Deepfit: 3D surface fitting via neural network weighted least squares. In Proceedings of the 16th European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 20–34.
34. Cao, J.; Zhu, H.; Bai, Y.; Zhou, J.; Pan, J.; Su, Z. Latent tangent space representation for normal estimation. *IEEE Trans. Ind. Electron.* **2022**, *69*, 921–929. [CrossRef]
35. Zhu, R.; Liu, Y.; Dong, Z.; Wang, Y.; Jiang, T.; Wang, W.; Yang, B. AdaFit: Rethinking Learning-based Normal Estimation on Point Clouds. In Proceedings of the IEEE International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 6118–6127.
36. Zhou, J.; Huang, H.; Liu, B.; Liu, X. Normal estimation for 3D point clouds via local plane constraint and multi-scale selection. *Comput.-Aided Des.* **2020**, *129*, 102916. [CrossRef]
37. Hashimoto, T.; Saito, M. Normal Estimation for Accurate 3D Mesh Reconstruction with Point Cloud Model Incorporating Spatial Structure. In Proceedings of the CVPR Workshops, Long Beach, CA, USA, 16–20 June 2019.
38. Zhou, J.; Jin, W.; Wang, M.; Liu, X.; Li, Z.; Liu, Z. Fast and accurate normal estimation for point clouds via patch stitching. *Comput.-Aided Des.* **2022**, *142*, 103121. [CrossRef]
39. Boulch, A.; Marlet, R. Deep learning for robust normal estimation in unstructured point clouds. *Comput. Graph. Forum* **2016**, 281–290. [CrossRef]
40. Ben-Shabat, Y.; Lindenbaum, M.; Fischer, A. Nesti-net: Normal estimation for unstructured 3d point clouds using convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 10112–10120.
41. Huber, P.J.; Ronchetti, E.M. *Robust Statistics*; Wiley: Hoboken, NJ, USA, 2009.
42. Rusu, R.B.; Cousins, S. 3D is here: Point Cloud Library (PCL). In Proceedings of the International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2011; pp. 1–4.
43. Lu, X.; Schaefer, S.; Luo, J.; Ma, L.; He, Y. Low rank matrix approximation for 3D geometry filtering. *IEEE Trans. Vis. Comput. Graph.* **2020**, *28*, 1835–1847. [CrossRef]

44. Fleishman, S.; Drori, I.; Cohen-Or, D. Bilateral mesh denoising. In Proceedings of the ACM SIGGRAPH 2003 Papers, San Diego, CA, USA, 27–31 July 2003; pp. 950–953.
45. Agathos, A.; Azariadis, P.; Kyratzi, S. Elliptic Gabriel Taubin smoothing of point clouds. *Comput. Graph.* **2022**, *106*, 20–32. [CrossRef]