



Article

Single-Stage Adaptive Multi-Scale Point Cloud Noise Filtering Algorithm Based on Feature Information

Zhen Zheng ^{1,2} , Bingting Zha ^{1,2,*}, Yu Zhou ^{1,2}, Jinbo Huang ^{1,2}, Youshi Xuchen ^{1,2} and He Zhang ^{1,2}

¹ ZNDY of Ministerial Key Laboratory, Nanjing University of Science and Technology, Nanjing 210094, China; zhengzhen@njust.edu.cn (Z.Z.); zy2020@njust.edu.cn (Y.Z.); huangjinbo@njust.edu.cn (J.H.); xcyouth@njust.edu.cn (Y.X.); hezhangz@njust.edu.cn (H.Z.)

² School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

* Correspondence: zhabingting@njust.edu.cn; Tel.: +86-25-8431-5455

Abstract: This paper proposes a single-stage adaptive multi-scale noise filtering algorithm for point clouds, based on feature information, which aims to mitigate the fact that the current laser point cloud noise filtering algorithm has difficulty quickly completing the single-stage adaptive filtering of multi-scale noise. The feature information from each point of the point cloud is obtained based on the efficient k-dimensional (k-d) tree data structure and amended normal vector estimation methods, and the adaptive threshold is used to divide the point cloud into large-scale noise, a feature-rich region, and a flat region to reduce the computational time. The large-scale noise is removed directly, the feature-rich and flat regions are filtered via improved bilateral filtering algorithm and weighted average filtering algorithm based on grey relational analysis, respectively. Simulation results show that the proposed algorithm performs better than the state-of-art comparison algorithms. It was, thus, verified that the algorithm proposed in this paper can quickly and adaptively (i) filter out large-scale noise, (ii) smooth small-scale noise, and (iii) effectively maintain the geometric features of the point cloud. The developed algorithm provides research thought for filtering pre-processing methods applicable in 3D measurements, remote sensing, and target recognition based on point clouds.

Keywords: point cloud denoising; kd-tree; grey relational analysis; principal component analysis; adaptive threshold; bilateral filtering algorithm



Citation: Zheng, Z.; Zha, B.; Zhou, Y.; Huang, J.; Xuchen, Y.; Zhang, H. Single-Stage Adaptive Multi-Scale Point Cloud Noise Filtering Algorithm Based on Feature Information. *Remote Sens.* **2022**, *14*, 367. <https://doi.org/10.3390/rs14020367>

Academic Editor: Sander Oude Elberink

Received: 3 December 2021

Accepted: 11 January 2022

Published: 13 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the rapid development of three-dimensional (3D) laser scanning technology, it has become easier to acquire accurate point clouds of ground targets, and point cloud processing technology has emerged as a key research direction in the field of vision processing [1–4]. However, in the process of acquiring point clouds of the scene where the target is located, the light detection and ranging (LiDAR) is often affected by the platform, as well as the accuracy of the transceiver sensor, interference caused by the detection environment, reflection characteristics of the target, and complexity of the target scene. As a result, the point cloud data obtained will inevitably be noisy; thus, it is necessary to perform noise filtering pre-processing steps [5–8].

According to its characteristics in the 3D point cloud, point cloud noise can be categorized as large-scale noise and small-scale noise. Large-scale noise is far from normal feature points, whereas small-scale noise is mixed with normal feature points [9]. The multi-scale noise filtering methods are divided into two-stage and single-stage methods. The two-stage methods adopt serial processing mode, and the processing flow is as follows: first, the point cloud without large-scale noise is obtained via the large-scale noise filtering method; then, the clean point cloud is obtained via the small-scale filtering method. The same point cloud is traversed multiple times (≥ 2 times), so the processing time is longer. The single-stage methods adopt a parallel processing mode, according to the information

about each point in the point cloud, multi-scale noise filtering can be achieved through a single traverse, so time efficiency is relatively high.

For large-scale noise filtering algorithms, the most commonly used methods are the PCL (point cloud library), such as statistical outlier removal, and radius filtering methods, etc. However, there are problems such as poor filtering effect and difficulty in determining parameters (such as the range of nearest neighbors k , standard deviation std , the radius of the neighborhood R , etc.). In order to improve the filtering effect, some scholars have made a lot of efforts to design some other large-scale filtering methods to improve the filtering effect. Su et al. proposed a feature-preserving point cloud denoising algorithm, based on the (i) Euclidean distance from the point to the cluster center of the k -means clustering algorithm and (ii) change in curvature of neighboring points. The algorithm has a better ability to maintain the point cloud features. However, when the large-scale noise content is too high, it will appear to cluster the noise into one class, and it is sensitive to the initial point and k (the number of neighborhood points) [10]. Zhao et al. obtained a collection of raster cells through voxel raster division to improve the k -mean based method, and they used a spatial clustering algorithm (density-based spatial clustering of applications with noise, DBSCAN) to determine the boundaries, as well as outlier noise to achieve large-scene scattered point cloud denoising. The algorithm is also sensitive to the parameters of radius and neighboring points number; when these parameters are not properly selected, the clustering time will increase, and the classification effect will be poor. [11]. Li et al. used the octree index to organize the point cloud and remove the initial noise, and they combined it with Bayesian estimation theory and statistical methods to remove noise precisely, the algorithm error rate is 0.09% (the PCL algorithm error rate is 0.48%), but how to choose the octree resolution is a problem [12]. Fan et al. explored an efficient image-based filtering approach to remove the data points located outside the indoor space of interest by mathematical morphology operations, hole filling, and connectively analysis; the approach is computationally efficient (<3 ms/10,000 points), but the algorithm needs to optimize each parameter manually, which is not conducive to automated point cloud processing. [13]. We summarize the advantages and disadvantages of various large-scale filtering methods are shown in Table 1. As can be seen from the table, these large-scale filtering algorithms are hardly balanced, in terms of adaptivity, speed, and filtering effect. Therefore, for large-scale automatic filtering tasks, we should ensure the automatic completion of the task, while increasing the speed and filtering effect as much as possible.

Table 1. Features of various large-scale noise filtering algorithms.

Method	Advantage	Disadvantage
PCL methods	fast, simple	poor filtering effect, manual parameter required
cluster-based	good filtering effect	slow, manual parameter required
index-based	fast, simple	manual parameter required
image-based	fast, good filtering effect	manual parameter required

For small-scale noise filtering algorithms, Fleishman et al. applied the bilateral filtering algorithm to point cloud processing for the first time [14]. Based on the Fleishman study, Zheng et al. proposed an effective, simple, and rapid anisotropic grid denoising algorithm, based on using local neighborhoods to filter grid vertices in the normal direction; however, since the bilateral filtering factor cannot be adjusted with the region features, this approach led to an over-photosmoothing of the flat region [15]. Sun et al. implemented normal weighted average filtering by designing a quadratic-based weighting function that automatically performed iterations in the filtered direction, this approach is as fast as the bilateral method, but it does not filter well at the edges or corner points [16]. Wang et al. used a point set relation method to determine the

adaptive weighting parameters, thus extending the LSM (least squares method) to smooth the point cloud, and the results show that the smoothing effect of the algorithm is good; however, the quantity and quality of the training dataset affect the performance of training-based methods [17]. Duan et al. first applied a neural network to a 3D point cloud denoising architecture, known as neural projection denoising (NPD), and better denoising effects can be obtained for a given point cloud model. However, the size of the database and complexity of the online environment limit the application of the method in the field of online point cloud filtering [18]. Zeng et al. extended the low-dimensional flow model used for image blocks to surface blocks of point clouds, which eventually achieves denoising, while preserving visually salient structural features, such as edges. However, this algorithm was highly complex and exhibited poor timeliness (running time is about 20 times longer than robust implicit moving least squares method) [19]. Wang et al. proposed a denoising method based on a normal vector distance classification, which divides the point cloud into two parts, i.e., smooth and sharp regions, and filters and denoises the smooth and sharp regions, using the weighted local optimal projection (WLOP) and bilateral filtering algorithms, respectively. However, a disadvantage of this means is that they usually require setting complex rules to ensure satisfactory removals [20]. In conclusion, for small-scale noise filtering, the most difficult part is to improve the timeliness and maintain the high-frequency information of the point cloud.

Following a survey of relevant literature, no study has been undertaken dedicatedly to achieve single-stage, multi-scale noise filtering. Conventional multi-scale noise filtering methods generally involve two-stage noise filtering, with the first stage using a large-scale filtering method to remove large-scale noise and the second stage using a small-scale filtering method to photosmooth the point cloud. Li et al. eliminated the large-scale noise via statistical outlier removal method (SOR) and radius filtering and filtered the small-scale noise by rapid bilateral filtering [21]. Liu et al. proposed an efficient denoising algorithm based on image and normal vector threshold judgement. First, large-scale noise points are eliminated using a global threshold judgement-based image; then, the Kuwahara filter algorithm is used for data smoothing, and denoising is achieved based on normal vector threshold judgements [22]. Li et al. proposed a library-based point cloud denoising method for multi-scale noise, using statistical and radius filtering algorithms to remove large-scale noise. Meanwhile, they used an improved bilateral filtering algorithm to smooth small-scale noise, but the filtering effect was still relatively poor [9]. Zhao et al. proposed a hierarchical (from coarse to fine) point cloud denoising algorithm. Circular filtering, using the anisotropic diffusion equation, was used to achieve large-scale denoising of the point cloud; then, small-scale denoising of the point cloud was achieved based on the curvature of the coarse-filtered point cloud. However, this approach is not effective when dealing with edges or corner points [23]. These above multi-scale filtering algorithms use a two-stage approach, and the two stages are independent of each other, which reduces the processing efficiency; so, it is necessary to perform both large- and small-scale filtering, based on the same point cloud features.

To quickly achieve single-stage, multi-scale noise adaptive filtering while maintaining the features of the point cloud, this paper proposes a fast, adaptive, multi-scale noise filtering algorithm for point clouds, based on feature information. The remaining structure of this paper is as follows. Section 2 describes the specific approach proposed. Section 3 shows and analyzes the experiment results. Section 4 provides a discussion and analysis of the results. Section 5 provides a brief summary and conclusions, as well as the future research directions related to the proposed method.

2. Materials and Methods

2.1. Study of Point Cloud Noise Feature

Point cloud noise is categorized as anomalous noise or burr noise. Anomalous noise, also known as large-scale noise, has high-frequency characteristics and is expressed as

“spikes”; this includes solitary points in the form of single points, as well as clusters of outlier points. Burr noise is also known as small-scale noise, whereas small-scale noise is mixed with normal feature points.

To evaluate the effects of various scales of noise on the point cloud, a sphere of radius $R_1 = 1$ was simulated, with O as the center of sphere I; the 24 points on the surface of sphere I are normal points in the point cloud. The distribution of the distances from the 12 points to the center of the sphere O follows a Gaussian distribution, with mean $\bar{L}_S = 1$ and standard deviation $\sigma = 0.2$. Specifically, the distances from the 12 points to O are {0.86, 0.948, 0.95, 0.971, 1.063, 1.105, 1.118, 1.161, 1.204, 1.228, 1.271, 1.306}, all of which constitute small-scale noise (Gaussian noise). The 12 points, distributed around the surface of sphere II (radius $R_2 = 2$), represent large-scale noise points (anomalous noise) in the point cloud, and their distances from O are {2, 2, 2, 2, 2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.5, 2.5, 2.5, 2.5}.

All of the points in the k -neighborhood ($k = 2$) were removed in the order of their original distance from the center (from small to large). For example, the distance to the first point is 0.86, with the average distance $\bar{L} = 0.86$, the distance of the second point is 0.948, with the corresponding average distance $\bar{L} = 0.894$, etc.; this allows a calculation of the average distance of the 48 points in the k -neighborhood. The trend of the average distance and the original distance in the k -neighborhood as a function of the number of points is shown in Figure 1. For large-scale isolated points (anomalous noise), the average distance increased rapidly, and the distance from large-scale noise ($R_2 = 2$) to the center O was relatively farther than the distance of normal points. Additionally, the average distance of the k -nearest neighbors containing large-scale noise was greater than the average distance of k -nearest neighbors not containing large-scale noise. These results indicated that anomalous noise can be filtered out directly via proper spatial distance. Therefore, we use distance-based feature information to separate the large-scale noise in this paper. Small-scale noise points are generally distributed in the vicinity of 1 and have less of an influence on the average distance; therefore, it is difficult to separate small-scale noise from the point cloud cluster by spatial distance. Instead, it is generally smoothed, according to the distribution law of Gaussian noise, to reduce the impact of small-scale noise on the follow-up. Therefore, we smooth small-scale noise based on vector information and noise distribution model in this paper.

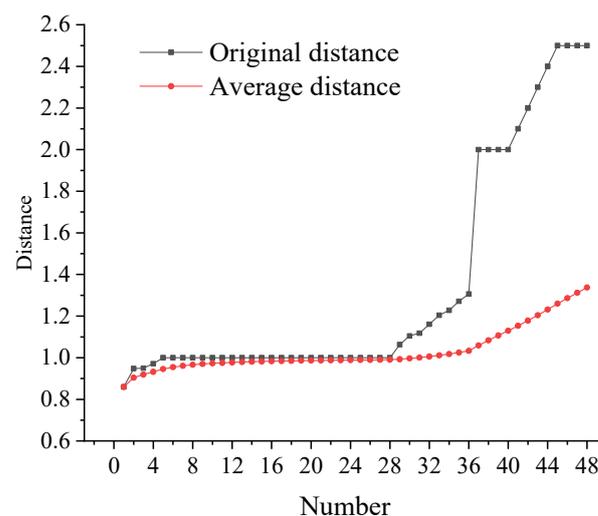


Figure 1. Effect of noise at each scale on the average distance.

2.2. Study Data and Evaluation Methods

2.2.1. Study Data

In order to test the performance of the algorithm proposed in the paper, six point cloud models from public datasets were selected as the research data, as shown in Figure 2. The number of points for the Engine model was 3028, the number of points for the Blade

model was 50,653, the number of points for the Building model was 93,039, the number of points for the Airplane model was 8176, the number of points for the Fish model was 3164, the number of points for the Cube model was 3177, the number of points for the People model was 5573, and the number of points for the Car model was 29,452. Large-scale noise, with a ratio of 0.02, and small-scale noise, of 20 dB, were added to the point cloud, where the small-scale noise represents Gaussian white noise. Figure 2a shows the original point cloud, Figure 2b shows the point cloud after adding large-scale noise, Figure 2c shows the point cloud after adding small-scale noise, and Figure 2d shows the point cloud after adding multi-scale noise. The experiments verifying the proposed algorithm were carried out based on these point cloud models.

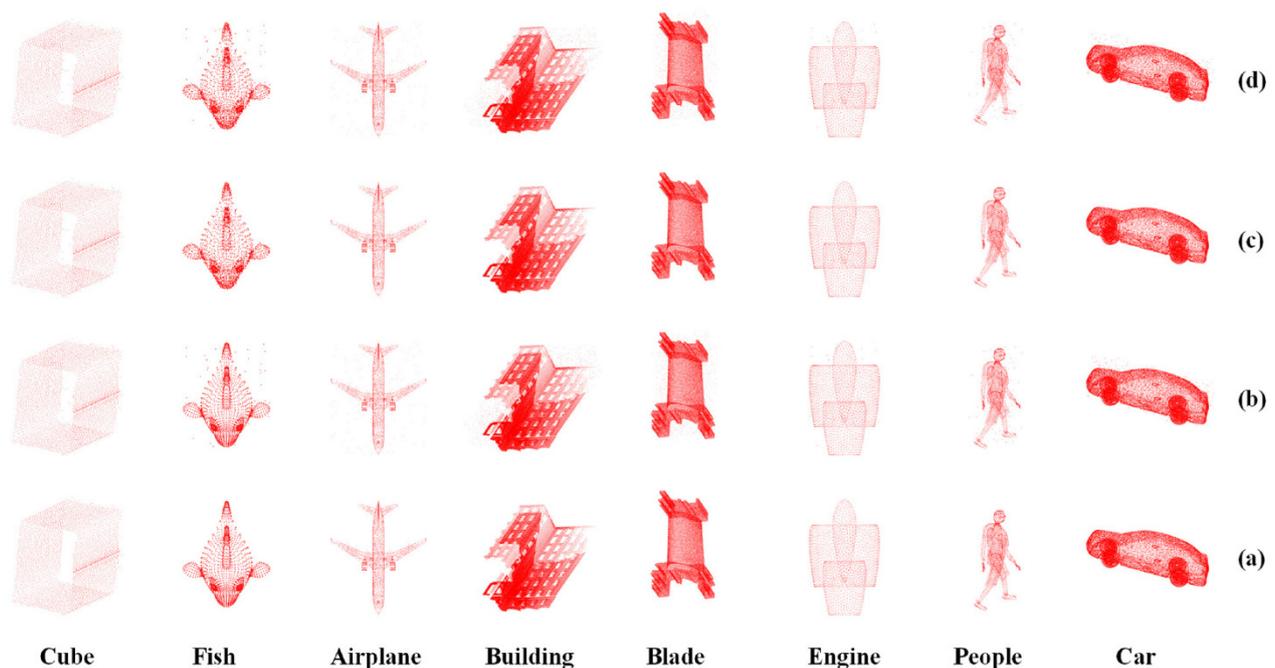


Figure 2. The point cloud models used in this work. (a) are the original point cloud models; (b) are the point cloud models after adding large-scale noise; (c) are the point cloud models after adding small-scale noise; (d) are the point cloud models after adding multi-scale noise.

2.2.2. Performance Evaluation

Since most current multi-scale filtering algorithms involve two-stage filtering methods (i.e., large-scale noise is filtered first, followed by small-scale noise), this paper compares the developed method with state-of-art large- and small-scale noise filtering algorithms to verify the large-scale noise denoising and small-scale noise photosmoothing abilities of the algorithm proposed in this paper.

A 3D comparison method was used to evaluate the noise filtering accuracy in this paper. Point clouds, filtered by various methods, were compared with the original noise-free point clouds in three dimensions, and the deviation distribution was obtained; the deviation values were counted to evaluate the advantage of the filtering algorithm in terms of filtering accuracy. However, isolated points are removed during 3D reconstruction, which affects the assessment of large-scale noise filtering effects; therefore, it is necessary to increase the objective evaluation index of large-scale noise filtering efficiency. The accuracy index (Acc) was used in this work to assess the large-scale noise filtering effect; the accuracy index represents the ratio of the total number Tu of all correctly classified points (correctly detected large-scale noise and normal points) to the total number of all elements in the classification task, as presented in Equation (1), where Fa denotes the total number of miss classified points and error classified points. The miss classified points were generally large-scale noise points that were incorrectly

classified as normal points, and the error classified points were normal points that were incorrectly classified as large-scale noise points.

$$Acc = Tu / (Tu + Fa) \quad (1)$$

Let us define the attribute information P_i of the k -nearest neighbor centroid as presented in Equation (2). Then P_i indicates the dispersion contribution of the point to the neighborhood, such that the smaller the value of p_i , the richer the detail of the point cloud.

$$p_i = \frac{D_i}{PSD_i + \sum_{j=1}^k PSD_j} \quad (i = 1, 2, \dots, n; j = 1, 2, \dots, k) \quad (2)$$

Entropy in information theory describes the uncertainty in the appearance of certain data, such that a higher entropy value indicates less information. Grey relation entropy theory is generated by combining grey relation analysis with entropy theory. Let a discrete sequence $\mathbf{X} = \{P_i | i = 1, 2, \dots, n\}$, containing n elements and $\forall i, p_i \geq 0$, be defined, such that $\sum p_i = 1$, $H_{\otimes}(\mathbf{X})$ is the grey entropy of the sequence \mathbf{X} , as presented in Equation (3) [24,25]. The total grey entropy of the point cloud indicates the amount of the detailed feature information contained in the point cloud. The smaller the value of $H_{\otimes}(\mathbf{X})$, the more chaotic the point cloud, more information it contains, and more it contributes to the geometric features of the region; this also means information contained is richer and more detailed. Additionally, when all values in the sequence are equal, the maximum grey entropy value of the sequence \mathbf{X} is $\ln n$, so the less high frequency information, the higher the grey entropy value.

$$H_{\otimes}(\mathbf{X}) \triangleq \begin{cases} -\sum p_i \ln p_i & p_i \neq 0 \\ 0 & p_i = 0 \end{cases} \quad (3)$$

The average grey entropy (AGE) of the point cloud represents the average geometric feature contribution of each point in the point cloud, and this parameter ph is defined as presented in Equation (4) [25]:

$$ph = \frac{H_{\otimes}(\mathbf{X})}{n} \quad (4)$$

2.3. Methodology

Because conventional point cloud noise filtering algorithms have difficulty achieving single-stage adaptive filtering of multi-scale noise, an adaptive single-stage, multi-scale noise filtering algorithm for point clouds, based on feature information, is proposed herein. Scattered point clouds do not have a clear topology. Therefore, an efficient kd-tree topological data structure with multidimensional data indexing was adopted in this study. The distanced from each point within the k -nearest neighborhood to the center point and the normal vector of each point in the k -neighborhood based on the kd-tree were obtained. The normal vector was corrected via Gaussian filtering and a kd-tree traversal method, and the average normal vector angle θ in the k -neighborhood was obtained. The adaptive threshold-based point cloud segmentation algorithm divided the point clouds into large-scale noise (LSN), feature-rich regions (FRR), and flat regions (FR) and processed them by removing noise, as well as applying the improved bilateral filtering algorithm and weighted mean algorithm based on grey relation analysis (WMGRA), respectively. This method effectively enabled rapid, single-stage, multi-scale noise filtering of noisy point clouds. The flow chart of the proposed algorithm is shown in Figure 3.

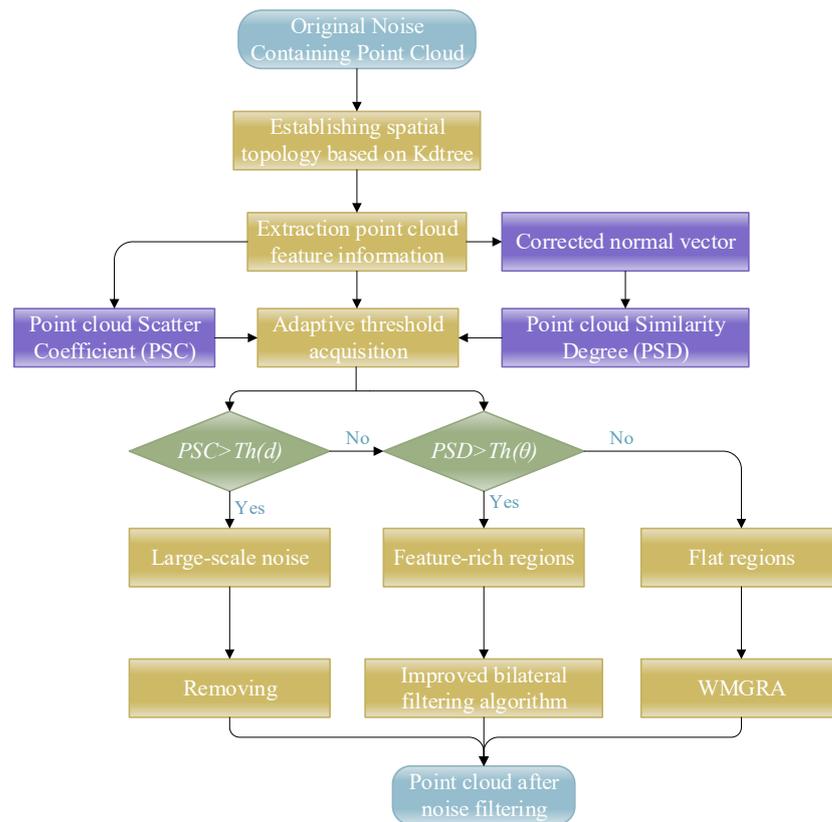


Figure 3. Flow chart of fast adaptive multi-scale noise filtering algorithm.

All points (n) in the point cloud comprise of the point cloud scatter coefficient sequence $\mathbf{D} = \{PSC_i | i = 1, 2, \dots, n\}$ and point cloud similarity degree sequence $\mathbf{\Theta} = \{PSD_i | i = 1, 2, \dots, n\}$.

By determining the appropriate separation threshold th_d of \mathbf{D} and separation threshold th_θ of $\mathbf{\Theta}$, through the adaptive threshold acquisition method (Section 2.3.3), the discriminant function of each region segmentation could be obtained using Equation (5).

$$p_i = \begin{cases} LSN & KSC > th_d \\ FRR & KSC < th_d, KSD < th_\theta \\ FR & KSC < th_d, KSD > th_\theta \end{cases} \quad (5)$$

For large-scale noise, the noise-removal process was performed directly. Small-scale noise was mainly Gaussian noise, which was effectively processed via WMGRA denoising, which embodies the advantages of low algorithm complexity and short computational durations. However, it destroys the point cloud details in feature-rich regions, resulting in the blurring of edge and corner points of the cloud. Therefore, to remove the noise, while maintaining the high-frequency features of the target, the proposed method employed WMGRA (Section 2.3.4) for small-scale noise smoothing in flat regions and an improved bilateral filtering algorithm (Section 2.3.5) for small-scale noise smoothing in feature-rich regions.

2.3.1. Normal Vector Estimation and Correction

The point clouds (shown in Figure 2) are a typical scattered point cloud without a clear topology, so we used a dichotomous tree (kd-tree) data structure to construct the basic topology of these point clouds in this paper, and the normal vectors were estimated using principal component analysis (PCA) in the k-neighborhood [26–28].

Assuming that the set of points in the k -neighborhood $\mathbf{X} = \{q_i \mid i = 1, 2, \dots, k\}$ and k is the number of point clouds in the neighborhood, the covariance matrix of \mathbf{X} is defined as presented in Equation (6):

$$\mathbf{C} = \begin{bmatrix} q_1 - \bar{\mathbf{P}} \\ \dots \\ q_k - \bar{\mathbf{P}} \end{bmatrix}^T \begin{bmatrix} q_1 - \bar{\mathbf{P}} \\ \dots \\ q_k - \bar{\mathbf{P}} \end{bmatrix}, \bar{\mathbf{P}} = \frac{1}{k} \sum_{i=1}^k q_i \quad (6)$$

where \mathbf{C} represents the extent to which points in the k -neighborhood deviate from the center of mass $\bar{\mathbf{P}}$. In this case, \mathbf{C} has three non-negative real eigenvalues, and the normal vector $\vec{\mathbf{n}}_p$ of the feature point is the eigenvector corresponding to the smallest eigenvalue. Although $\vec{\mathbf{n}}_p$ represents the direction of least change for all points in the neighborhood and has some noise immunity, the normal vector in the neighborhood will have discontinuous directions when the point cloud is cluttered; therefore, the estimated normal vector must be corrected.

All points were traversed according to the kd-tree topology, and the normal vector orientations of all points were corrected to be consistent. According to the vector dot product, when the normal vector $\vec{\mathbf{n}}_p$ of the central feature point and the normal vectors of other points in the k -neighborhood are negative, i.e., indicating two vectors have opposite directions, the direction of the seed point is favored, as presented in Equation (7).

$$\vec{\mathbf{n}}_p = \begin{cases} \vec{\mathbf{n}}_p & \vec{\mathbf{n}}_p \cdot \vec{\mathbf{n}}_{q_i} \geq 0 \\ -\vec{\mathbf{n}}_p & \vec{\mathbf{n}}_p \cdot \vec{\mathbf{n}}_{q_i} < 0 \end{cases} \quad (7)$$

To ensure that the normal vector in the k -neighborhood varies continuously, the weighted mean filter was used in this study, and the normal vector correction model is given in Equation (8):

$$\vec{\mathbf{n}}_p = \frac{\sum_{q_i \in \mathbf{X}} \vec{\mathbf{n}}_{q_i} \exp\left(-\frac{\|\vec{\mathbf{p}} - \vec{\mathbf{q}}_i\|^2}{2\sigma_n^2}\right)}{\sum_{q_i \in \mathbf{X}} \exp\left(-\frac{\|\vec{\mathbf{p}} - \vec{\mathbf{q}}_i\|^2}{2\sigma_n^2}\right)} \quad (8)$$

where $\vec{\mathbf{n}}_{q_i}$ is the normal vector of points in the k -neighborhood, $\vec{\mathbf{p}} - \vec{\mathbf{q}}_i$ represents the Euclidean distance between points $\vec{\mathbf{p}}$ and $\vec{\mathbf{q}}_i$, and σ_n represents the influence factor of the Euclidean distance to $\vec{\mathbf{p}}$ in the k -neighbourhood on $\vec{\mathbf{n}}_{q_i}$. The radius of the k -neighbourhood is taken in this paper.

2.3.2. Point Cloud Scatter Coefficient and Similarity Degree

The mean distance of the k -nearest neighbors containing large-scale noise is greater than the mean distance of k -nearest neighbors not containing large-scale noise; thus, we define the point cloud scatter coefficient (*PSC*) within the k -nearest neighbors based on this feature. Specifically, *PSC* represents the point cloud dispersion within the k -nearest neighbors. The larger the *PSC*, the more discrete and sparser the point cloud distribution, and the higher the probability that the k -nearest neighbors contain large-scale noise. The *PSC* is defined as presented in Equation (9).

$$PSC = \frac{1}{k} \sum_{q_i \in \mathbf{X}} \|\vec{\mathbf{p}} - \vec{\mathbf{q}}_i\| \quad (9)$$

The point cloud similarity degree (*PSD*) describes the directional consistency of the normal vectors of the points and centroids in the k -nearest neighborhood. The larger the

PSD , the closer the normal vector directions of the points, and the PSD is determined as presented in Equation (10).

$$PSD = \frac{1}{k} \sum_{q_i \in X} \cos \langle \vec{n}_p, \vec{n}_{q_i} \rangle \quad (10)$$

2.3.3. Adaptive Threshold Acquisition Algorithm

The normal vector directions of the points in the flat region were generally consistent, whereas the normal vector directions of the points in the feature-rich region (including high-frequency feature information and small-scale noise) were more diverse. According to this parameter, the target point cloud can be partitioned into a flat region and a feature-rich region by choosing a suitable threshold. For the processor to automatically perform the denoising filtration, the filtering algorithm must automatically determine the optimal threshold (th), according to the point clouds in an adaptive manner.

The sequence of point cloud eigenvalues is $\mathbf{T} = \{T_i | i = 1, 2, \dots, n\}$. Let $T_t = th$ and t be the index of the corresponding serial number of T_t . Then, t can divide the sequence \mathbf{T} into two parts, i.e., the background and target. When $1 < i \leq t$, the elements T_i are background; when $t < i \leq n$, the elements are target. Calculating the proportion of each element as $Q_i = T_i / \sum_{i=1}^n T_i$, the total proportion of background elements is $Q_1 = \sum_{i=1}^k Q_i$, and the total proportion of target elements is $Q_2 = \sum_{i=k+1}^n Q_i$.

The mean of the background elements is $b_1 = (\sum_{i=1}^k T_i Q_i) / Q_1$, the mean of the target elements is $b_2 = (\sum_{i=k+1}^n T_i Q_i) / Q_2$, and the mean of the entire sequence is $b_G = \sum_{i=1}^n T_i Q_i$ [29]. The separation evaluation criteria objective function is defined as presented in Equation (11).

$$J = Q_1(b_1 - b_G)^2 + Q_2(b_2 - b_G)^2 \quad (11)$$

If we let $b_1 < b_G < b_2$, the larger the $\Delta b = b_2 - b_1$, the further the target element is from the interval where the background element is located, i.e., the better the value of the objective function of the evaluation criterion, the more effective the separation. All separation evaluation criterion objective function values form the set $\{J_i | i = 1, 2, \dots, t, \dots, n\}$, such that there must be a maximum criterion objective function value. In other words, there is an optimal separation point k^* , such that $J(k^*) = \max_{0 \leq k \leq l} J(k)$, and the best separation is achieved when a threshold $th = T_{k^*}$ is selected.

2.3.4. Weighted Mean Algorithm Based on Grey Relation Analysis

In 1982, Professor Deng founded the discipline of grey system theory for analyzing uncertain systems with the characteristics of “small data (sample size is small)” and “poor information (less useful information in the data)”. After more than 30 years of vigorous development, grey system theory has been applied to forecasting, planning, decision-making, gemology, clustering, control, etc. [30–33]. The distribution that small-scale noise obeys in the neighborhood is unknown, and the relationship between points in the neighborhood is also unknown, so the local information of points in the neighborhood to be processed is an uncertain system. Therefore, grey system theory is well applicable to point cloud noise filtering tasks. To this end, we make the first attempt to apply grey system theory to point cloud noise processing in this paper. Grey relation analysis (GRA) is an important branch of the grey theory system. It can be used to measure the similarity and proximity between two grey systems. Small-scale noise points are far from the median of the neighborhood in the flat regions, so the similarity and proximity between systems C_A (consists of small-scale noise points) and C_B (Consists of the mean of the k -neighborhood) is very low. According to this feature, we propose a small-scale noise filtering algorithm for flat regions, based on WMGRA, in this paper.

In the k -neighborhood, the angle $\theta_i = \cos \langle \vec{n}_p, \vec{n}_{q_i} \rangle$ of the normal vector is related to the grey comparison sequence $\mathbf{X}_1 = \{x_i = \theta_i | i = 1, 2, \dots, k\}$ used for grey relation analysis.

The mean $m = \text{mean}\{\mathbf{X}_1\}$ of the k -neighborhood is an element of GRA, and the grey reference sequence is $\mathbf{X}_0 = \{x_0(i) = m | i = 1, 2, \dots, k\}$. Small-scale noise obeys a Gaussian distribution, so the weighted mean value algorithm based on GRA (WMGRA) can smooth the small-scale noise in flat regions.

The grey reference sequence is subjected to a dimensionless unitary transformation of the initial values with the comparison sequence (Equation (12)).

$$\begin{cases} y_0(j) = \frac{x_0(j)}{\min\{\mathbf{X}_0, \mathbf{X}_1\}} \\ y_1(j) = \frac{x_1(j)}{\min\{\mathbf{X}_0, \mathbf{X}_1\}} \end{cases} \quad (12)$$

The proximity of each element in the grey comparison sequence to the corresponding element in the reference sequence is computed using Equation (13).

$$r(j) = \frac{1}{1 + |y_0(j) - y_1(j)|} \quad (j = 1, 2, \dots, k) \quad (13)$$

The small-scale noise filtering factor β in the flat region is defined as presented in Equation (14).

$$\beta = \frac{\sum_{q_i \in \mathbf{X}} (1 - r_i) \langle \vec{p} - \vec{q}_i, \vec{n}_{q_i} \rangle}{\sum_{q_i \in \mathbf{X}} (1 - r_i)} \quad (14)$$

In this paper, the degree of proximity $r(j)$ serves as the weight assignment coefficient, and the points after filtering the flat region are obtained using Equation (15); the time complexity of the WMGRA is $O(n)$:

$$\vec{P}_i = \vec{P} + \beta \vec{n}, \quad (i = 1, 2, \dots, n; j = 1, 2, \dots, k) \quad (15)$$

where \vec{P}_i is the filtered point, \vec{P} is the original point, \vec{n}_p is the normal vector at \vec{P} , and β is the small-scale noise filtering factor defined by Equation (14). The small-scale noise filtering task is completed by moving the processing point \vec{P} a certain distance along the direction of the normal vector \vec{n}_p .

2.3.5. Improved Bilateral Filtering Algorithm

The point cloud bilateral filtering algorithm is based on development of image bilateral filtering, which accounts for the spatial Euclidean distance between pixel points in the k -neighborhood and the differences in the normal vectors of each point. This process can be described as a point moving a certain distance in its normal direction. Therefore, the bilateral filtering algorithm is generally used as a smoothing process for small-scale noise, and it cannot effectively remove anomalous noise points. The bilateral filtering algorithm based on point cloud data is presented in Equation (16) [14,34]:

$$\vec{p}_i = \vec{p} + \alpha \vec{n}_p \quad (16)$$

where \vec{p}_i is the filtered point, \vec{p} is the original point, \vec{n}_p is the normal vector at \vec{p} , and α is the bilateral filtering factor defined by Equation (17):

$$\alpha = \frac{\sum_{\vec{q} \in R_k} w_s(\|\vec{p} - \vec{q}\|) w_r(\langle \vec{p} - \vec{q}, \vec{n}_p \rangle) \langle \vec{p} - \vec{q}, \vec{n}_p \rangle}{\sum_{\vec{q} \in R_k} w_s(\|\vec{p} - \vec{q}\|) w_r(\langle \vec{p} - \vec{q}, \vec{n}_p \rangle)} \quad (17)$$

where R_k is a set of the nearest neighbors of the central point \vec{p} , \vec{q} is a point within R_k , \vec{n}_p is the normal at the point \vec{p} , \vec{n}_q is the normal at the point \vec{q} , $\|\vec{p} - \vec{q}\|$ is the Euclidean distance between points \vec{p} and \vec{q} , $\langle \vec{p} - \vec{q}, \vec{n}_p \rangle$ denotes the length of the projection of \vec{q} in the direction of \vec{p} normal to \vec{n}_p , $w_s(x) = \exp(-x^2/2\sigma_s^2)$ is the Gaussian kernel function, and σ_s is the Euclidean distance from \vec{q} to \vec{p} in the k -neighborhood for \vec{p} . The larger the σ_s , the better the effect on the smoothness of the point cloud and larger the average distance from all points in the k -neighborhood to \vec{p} . The expression $w_r(x) = \exp(-x^2/2\sigma_r^2)$ is another form of the Gaussian kernel function, where σ_r is the effect of the projection distance of each point in the k -neighborhood on the normal vector of \vec{q} on \vec{p} ; the larger the effect, the better the retention of the feature. The value of σ_r is generally the standard deviation of the set of projected distances of the points on the \vec{p} normal vector [15].

Because the WMGRA ignores the distance information between points, the high-frequency feature points will be over-smoothed when using this method to process the feature-rich region. Typically, a bilateral filtering algorithm is used to smooth the feature-rich region of the point cloud. However, the traditional bilateral filtering factor uses the projection distance of neighboring points on their tangent planes as the weighting factor, which has some limitations in the sharp region, where the feature changes are more drastic.

For the feature-rich region of the point cloud shown in Figure 4, it is necessary to maintain the high-frequency feature of point P. The projection lengths of \vec{n}_1 and \vec{n}_2 on \vec{n} are the same, i.e., $d_1 = d_2$, if the traditional bilateral filtering factor is used. Accordingly, the impacts of \vec{n}_1 and \vec{n}_2 on \vec{n} are the same, but the weight of \vec{n}_2 should be larger than that of \vec{n}_1 to maintain the feature of P.

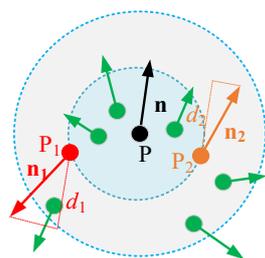


Figure 4. Diagram of the retention factor.

The cosine of the angle between the normal vectors was used as the feature retention factor of the bilateral filtering factor; thus, the weight was minimized when the angle is 90° . When the normal vector of each point in the k -neighborhood is relatively consistent with the direction of the normal vector of the point to be sought, there is a smaller angle between the two vectors, and the corresponding assigned weight should be larger. The definition of the improved bilateral filtering factor α is presented in Equation (18), where $\text{CosAng}_{pq} = 1 - \cos\langle \vec{n}_p, \vec{n}_{q_i} \rangle$, and $\text{Eu}_{pq} = \|\vec{p} - \vec{q}\|$ denotes the Euclidean distance between \vec{p} and \vec{q} .

$$\alpha = \frac{\sum_{q_i \in X} w_s(\text{Eu}_{pq}) w_r(\text{CosAng}_{pq})(1 - r_i) \langle \vec{p} - \vec{q}_i, \vec{n}_{q_i} \rangle}{\sum_{q_i \in X} w_s(\text{Eu}_{pq}) w_r(\text{CosAng}_{pq})(1 - r_i)} \tag{18}$$

The filtered points of the feature-rich region were obtained using Equation (19), and the time complexity of the improved bilateral filtering algorithm is $O(n^2)$.

$$\vec{P}_i = \vec{P} + \alpha \vec{n} \tag{19}$$

3. Results

3.1. Normal Vector Correction Algorithm Validation

To verify the effectiveness of the normal vector filtering algorithm proposed in this paper, the point cloud normal vector was extracted from the Cube point cloud. The effect of the normal vector extraction is presented in Figure 5, where Figure 5a shows the point cloud normal vector obtained by the moving least square (MLS) method, and Figure 5b shows the normal vector obtained by the normal vector filtering algorithm, proposed in this work.

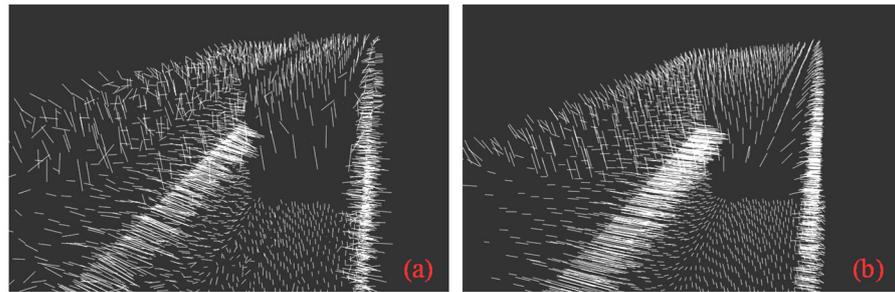


Figure 5. Normal vector of a cube point cloud. (a) is the normal vector obtained by MLS method; (b) is the normal vector obtained by proposed method.

Comparing these two vectors reveals that the point cloud normal vectors obtained by the MLS method are messier, whereas the flat areas of the point cloud normal vectors obtained by the proposed algorithm are neater and more uniform, with smoother changes at the edges. These results verify the effectiveness of the normal vector filtering algorithm proposed in this paper.

3.2. Region Segmentation Algorithm Validation

To verify the effectiveness of the adaptive point cloud region segmentation algorithm, experiments were performed using the Cube point cloud and the Engine point cloud, and the region segmentation is presented in Figure 6, where the green points represent flat regions, the red points indicate feature-rich regions, and the blue points denote large-scale noise. It is clear from these figures that the segmented, feature-rich regions are mainly concentrated at the edges or corners of the point cloud, which is consistent with the actual situation and verifies the effectiveness of the proposed adaptive point cloud region segmentation algorithm.

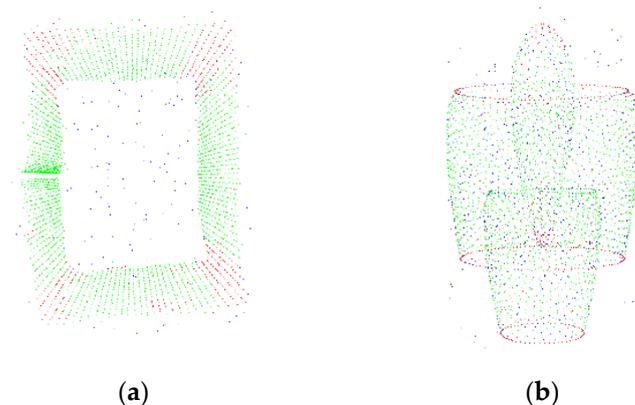


Figure 6. Area segmentation effect in the (a) Cube Engine point cloud; (b) Engine point cloud.

3.3. Large-Scale Noise Filtering Experiments

To verify the large-scale noise filtering performance of the proposed algorithm, we compared it to other state-of-the-art algorithms, including the statistical outlier removal

filtering method (SOR), the radius outlier removal filtering method (ROR), the denoising algorithm (based on the octree index, proposed by Li et al. [9]), and the denoising method (based on the DBSCAN algorithm, proposed by Zhao et al. [11]).

Comparing these various methods confirmed that certain parameters can significantly influence the filtering effect. For example, the parameters k_0 and std in the SOR method indicate that, when the points in the k_0 -nearest neighborhood exceed the variance of std times (generally, $std = 1, 2, 3$), the points to be sought are considered as large-scale noise. Similarly, the parameters R and k_1 in the ROR denote the radius of the neighborhood and number of points in the neighborhood, respectively; if the number of points in the neighborhood with radius R is less than k_1 , the processing point is considered as large-scale noise ($k_1 = 3$ in this paper). Moreover, different filtering results were obtained for different octree layers k_2 and different DBSCAN parameters (search neighborhood radius = Eps , minimum neighborhood points = $MinPts$).

For the purpose of comparing these methods, some parameters were set to more appropriate values in this paper; for example, $std = 2$, $k_1 = 3$. Figure 7 shows the large-scale noise filtering effect achieved by various methods on Airplane and Building point cloud models, where the parameter used for SOR was $k_0 = 50$, the parameter used for ROR was $R = 1.5$, the parameter used for the octree method was $k_2 = 5$, and the parameter used for DBSCAN was $Eps = 2.5$. Figure 7 reveals that the ROR and octree method have numerous error classified points and miss classified points, whereas the DBSCAN method and proposed algorithm have essentially no false detection cases and only a few missed detection cases.

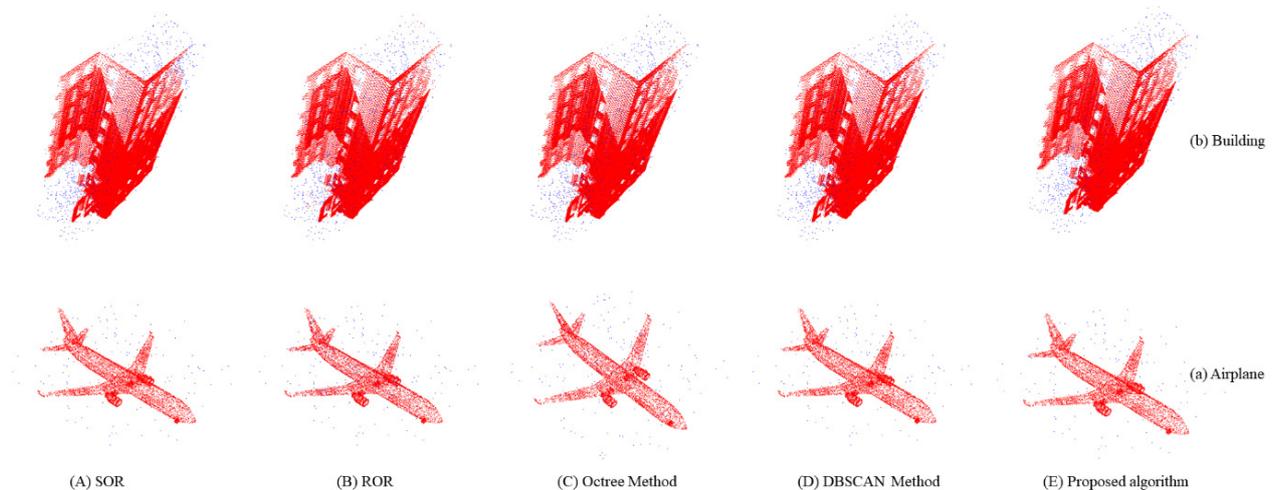


Figure 7. Large-scale noise filtering effects. (a) are Airplane point cloud; (b) are Building point cloud.

To objectively describe the effectiveness of various methods for dealing with large-scale noise, the number of miss-classified and error classified points, as well as the accuracy index (Acc) were obtained, when evaluating Airplane and Building point clouds, as shown in Figure 8. Because the points in the neighborhood are more concentrated in the Airplane point cloud, many large-scale noise values and original points are mixed together and difficult to separate; therefore, the number of miss classified points is much larger than the number of error classified points, and the Acc of our algorithm is the smallest. In the case of the Building point cloud, more Building edge points are mixed with the noise, so the error classified points are more likely to occur; thus, the number of error classified points is larger than the number of missed detections, the accuracy index of the proposed algorithm is the smallest and only slightly larger than that of DBSCAN ($Eps = 0.5$). Considering the drawback that the DBSCAN algorithm is time-sensitive and the parameters must be selected manually, the proposed algorithm has the strongest overall large-scale noise denoising ability. These results and comparisons further verify the denoising ability of the algorithm proposed in this paper for large-scale noise.

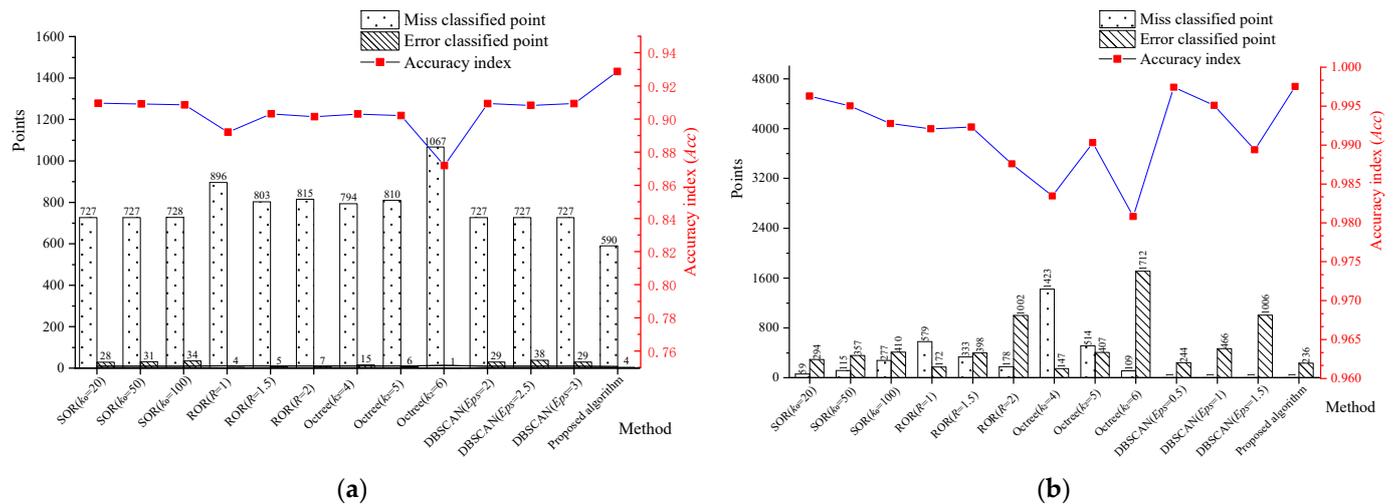


Figure 8. Large-scale noise filtering results. (a) are Airplane point cloud; (b) are Building point cloud.

3.4. Small-Scale Noise Filtering Experiments

To verify the small-scale noise filtering performance of the proposed algorithm, we compared it to other state-of-the-art algorithms, including the robust implicit moving least square method (RIMLS) proposed by Öztireli et al. [35], the bilateral filtering algorithm (BFA) developed by Zheng et al. [15], and the smoothing algorithm based on WLOP and bilateral filtering (WLOPBF) proposed by Wang et al. [20]. The small-scale noise filtering effects (i.e., smoothing effect) of the Fish and Blade point clouds were obtained using the algorithm proposed in this paper, and the results are compared with those obtained with the three aforementioned algorithms in Figure 9. When dealing with edge regions, BFA and WLOPBF adopt a traditional bilateral filtering algorithm, so the errors at the edges are larger. In contrast, when dealing with flat regions, the errors of WLOPBF, RIMLS, and our algorithm are relatively smaller.

With the original point cloud as the reference point cloud and the point cloud after applying the small-scale noise filtering algorithm as the test comparison point cloud, the 3D deviation distribution chromatogram of the processed point cloud was obtained (Figure 10). The deviation of the algorithm proposed herein is generally concentrated around zero, indicating that the smoothing effect of the developed method is superior to that of the three previous strategies.

To objectively describe the effect of various methods, when dealing with small-scale noise, the average deviation (AD), point number beyond the critical deviation value (PBCV, when the deviation range is $[-1, 1]$), and AGE of various methods were determined, when dealing with Fish and Blade point clouds (Figure 11). The AGE of the algorithm proposed in this paper is the smallest, which indicates that the high frequency information features are better maintained. Additionally, the AD and PBCV of the proposed algorithm are the smallest, which further verifies the effectiveness of the small-scale filtering of the algorithm developed in this work.

In general, the time required for small-scale noise smoothing is more than that for large-scale noise removal. Therefore, this section compares the running time of the proposed algorithm with other small-scale filtering algorithms to test the timeliness of the algorithm proposed in this paper, and the processing time of various methods were obtained, when dealing with Fish, Blade and People point clouds, to compare the timeliness of each algorithm; the running time is shown in Figure 12. For the Fish model, the proposed algorithm in this paper is 38.64% faster than the BFA and 71.23% faster than RIMLS; for the Blade model, the proposed algorithm in this paper is 39.39% faster than the BFA and 68.7% faster than RIMLS; for the People model, the proposed algorithm in this paper is 37.78% faster than the BFA and 65.72% faster than RIMLS; for the Car model, the

proposed algorithm in this paper is 43.57% faster than the BFA and 68.5% faster than RIMLS. Therefore, it can be concluded from Figure 12 that the proposed algorithm in this paper can quickly achieve multi-scale noise filtering of point clouds.

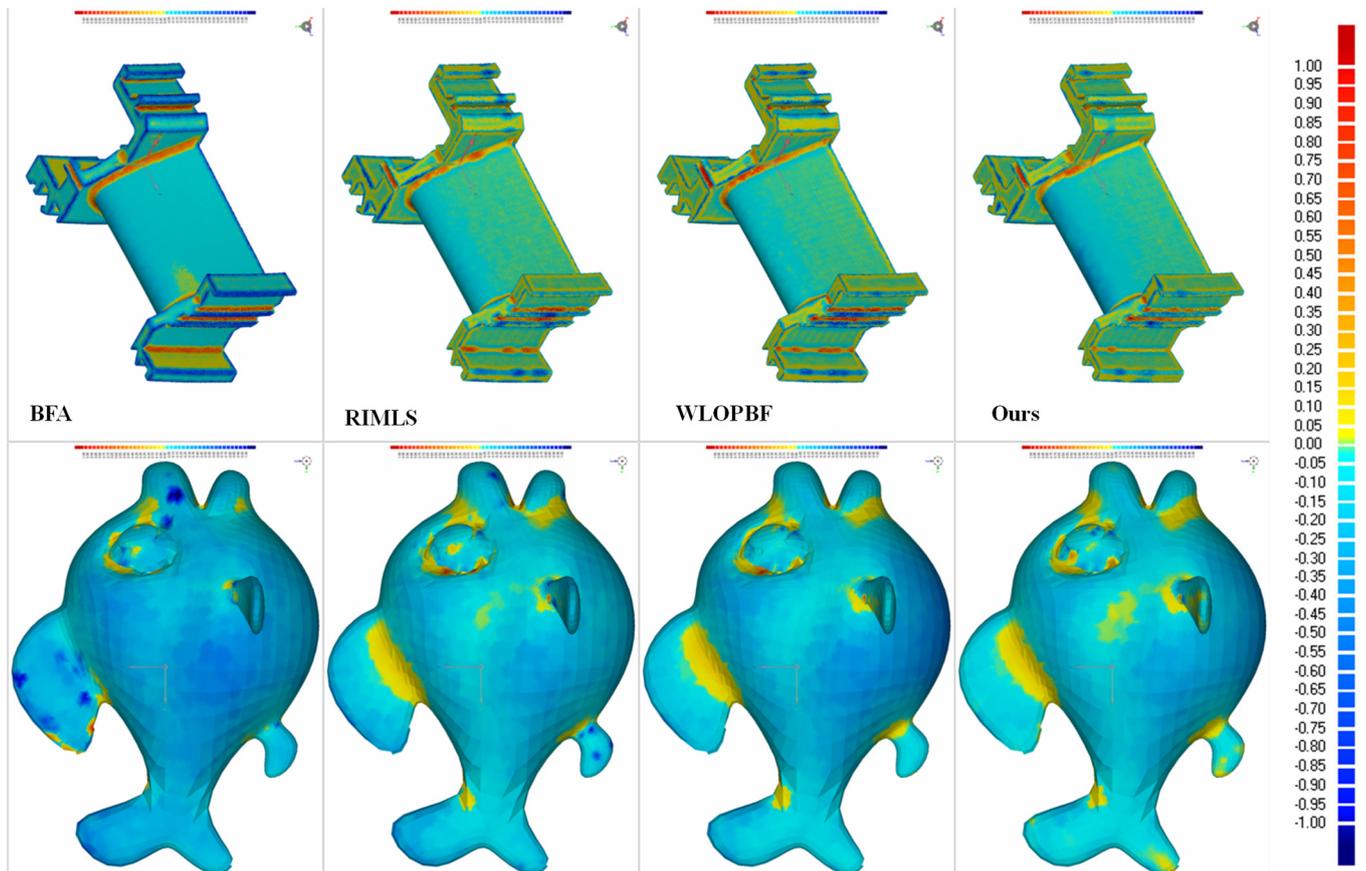


Figure 9. Small-scale noise filtering effects. (Bottom) are Fish point cloud; (Top) are Blade point cloud.

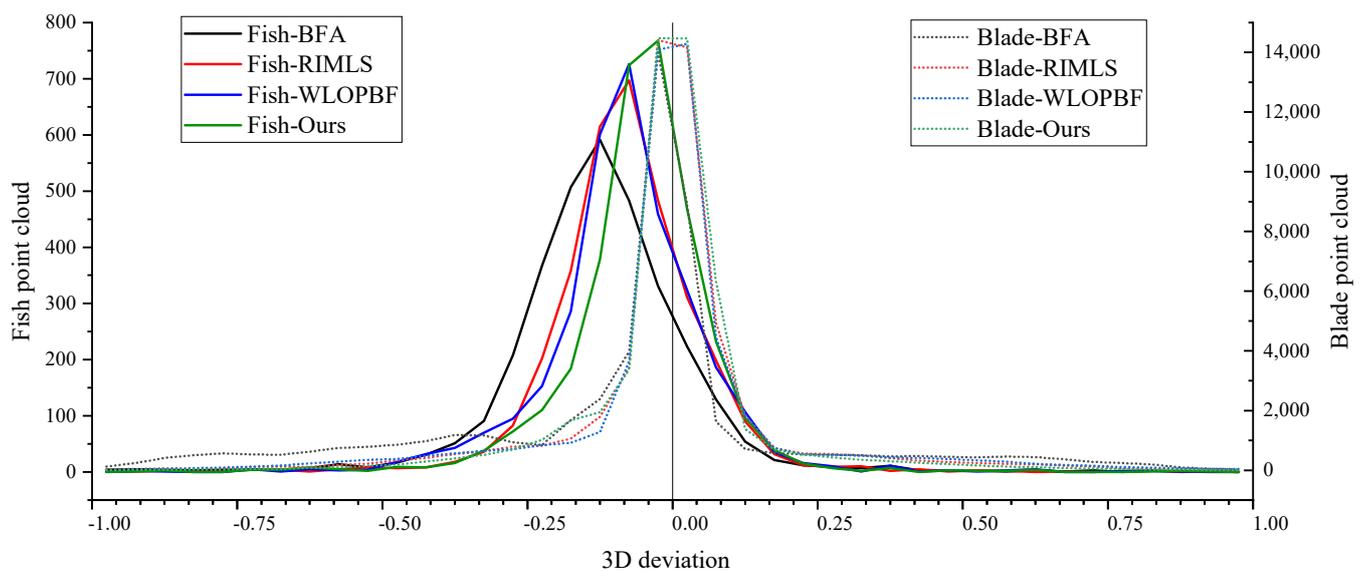


Figure 10. 3D deviation distribution chromatogram.

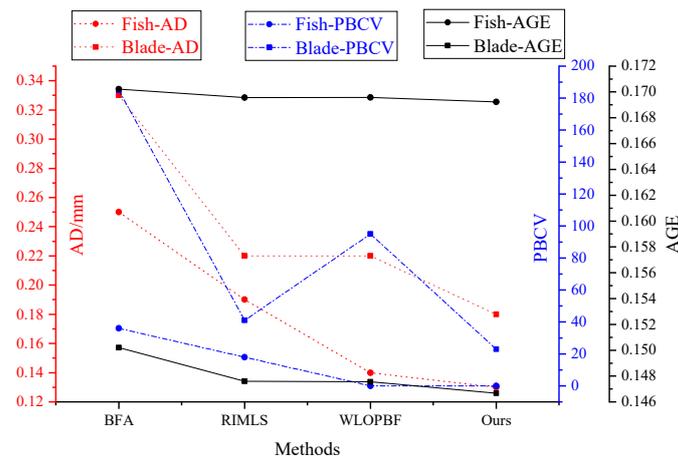


Figure 11. Small-scale noise filtering results.

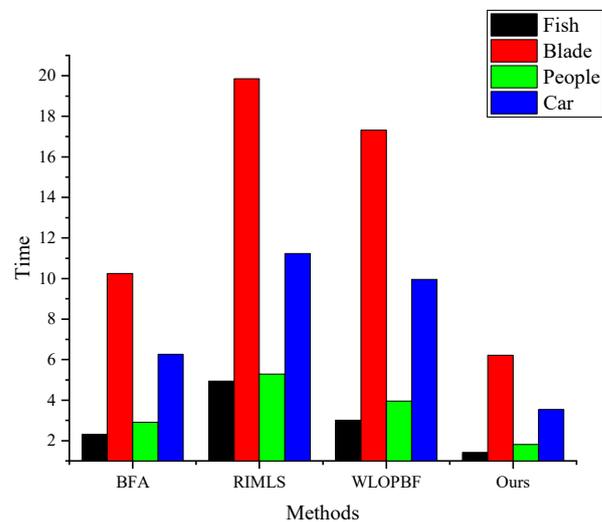


Figure 12. Running time of small-scale noise filtering methods.

4. Discussion

The accuracy of normal directly affects the filtering effect of noise. Therefore, one of the important future directions of noise filtering algorithm is the normal vector filtering algorithm. Sun et al. proposed a normal vector filtering method based on weighted averaging of neighboring normal [16]. However, it uses a constant angle threshold, which makes it difficult to achieve the best filtering effect. To this end, Liang et al. proposed an adaptive angle threshold acquisition method, which achieved a better normal vector filtering and point cloud smoothing effect, but the complexity of the algorithm is high, and the parameters in the threshold acquisition model are difficult to determine [36]. In order to deal with these problems, we propose a vector estimation and correction algorithm, based on directional discrimination and Gaussian filtering; the results show that the point cloud normal vector estimated by this method is more regular and smoother, compared to the traditional MLS method.

None of these conventional approaches to date achieves single-stage, multi-noise filtering, these methods typically use a serial (two-stage) processing model to achieve the tasks [9,21,22]. In general, large-scale noise is usually sensitive to the distance in the neighborhood, while small-scale noise is usually related to the angle between the normal vectors in the neighborhood. To this end, we obtain the feature information of each point by traversing all points through an efficient kd-tree and achieve multi-scale noise filtering by partitioned parallel processing, based on feature information (point

cloud scatter coefficient, PSC; point cloud similarity degree, PSD). Our results suggest that the algorithm proposed in this paper can single-stage quickly and adaptively (i) filter out large-scale noise, (ii) smooth small-scale noise, and (iii) effectively maintain the geometric features of the point cloud.

To objectively evaluate the comprehensive performance of the algorithms proposed in this paper, first, we compared the developed method with state-of-art large-scale filtering algorithms to verify the large-scale noise denoising abilities of the proposed method; then, we compared the developed method with state-of-art, small-scale noise filtering algorithms to verify the small-scale noise photosmoothing abilities of the proposed algorithm. Moreover, we propose several filtering effect evaluation indexes. For example, for large-scale noise filtering evaluation, there are miss classified points index, error classified points index, and large-scale noise recognition accuracy rate index (Acc); for small-scale noise filtering evaluation, there are total grey entropy and average grey entropy (AGE). These indexes offer new ways to objectively evaluate the performance of the point cloud filtering algorithms.

There are two general ways to reduce the time for small-scale noise filtering, one of which is to design low-complexity algorithms [16,37], and the other is to partition the point cloud [20]. In this paper, we use both of the above methods to reduce running time. First, a point cloud partitioning process is used to divide the point cloud into large-scale noise, a feature-rich region, and a flat region; these three regions can be handled separately later to reduce the waste of computing power. Second, a WMGRA-based, small-scale noise filtering algorithm for the flat region is designed in this paper, and the complexity of the algorithm is only $O(n)$, so smooth tasks can be done quickly. Our results suggest that the running time of the proposed algorithm is minimal, compared to other state-of-the-art algorithms. Furthermore, we apply grey model theory to the field of point cloud filtering for the first time; these studies, thus, offer a new strategy to point cloud processing.

Although the proposed algorithm achieves single-stage, adaptive, multi-scale point cloud noise filtering, there are also limitations. First, the results show that the processing time of the algorithm is still too long to be applied to high-speed online processing platforms. For example, the complexity of the adaptive threshold acquisition method designed in this work is high, which needs to be improved in the future. Second, the normal vector, PSD, and PSC are related to the neighborhood size. Generally, the larger the neighborhood is, the more realistic and accurate the description of the point cloud is; therefore, the better the filtering effect is, but at the cost of increased computing time. The selection of neighborhood size in this paper is obtained empirically, and the effect of point cloud neighborhood on the filtering results is not investigated; so, how to select the neighborhood size appropriately is a future research direction. In addition, the set of feature information obtained in this paper is inconsistent, such as PSD is k -nearest neighbors and PSC is k -neighborhood; in addition, the adaptive threshold acquisition algorithm is based on the global. So, the set of all feature information needs to be unified in future study, but since different point cloud models (or different noise distribution, different noise content, etc.) correspond to different optimal choices, and the threshold selection method based on local information is sensitive to noise. Therefore, it is a very challenging task to unify all sets.

5. Conclusions

This paper proposes a fast, single-stage, adaptive, multi-scale noise filtering algorithm for point clouds. Based on the feature information (e.g., point cloud scatter coefficient and point cloud similarity degree), obtained by a kd-tree and the amended normal vector, the noise-containing point cloud is divided into three parts: large-scale noise, feature-rich regions, and flat regions. The large-scale noise is removed directly, and the feature-rich and flat regions are filtered using an improved bilateral filtering algorithm and WMGRA, respectively. The results confirm that the proposed algorithm can quickly and adaptively filter out large-scale noise and smooth, small-scale noise, while also effectively maintaining the geometric features of the point cloud.

In follow-up work, we will focus on the three aspects: optimizing the efficiency of the algorithm to improve filtering performance, reducing the complexity of the algorithm for adapting to high-speed online processing situations, and building more robust features information to improve point cloud region segmentation abilities.

Author Contributions: Conceptualization, Z.Z. and B.Z.; data curation, Z.Z. and Y.Z.; funding acquisition, B.Z., H.Z. and Z.Z.; methodology, Z.Z.; resources, Z.Z. and H.Z.; algorithm, Z.Z. and J.H.; supervision, B.Z. and H.Z.; validation, Y.Z., J.H. and Y.X.; visualization, Z.Z. and Y.X.; writing—original draft, Z.Z.; writing—review and editing, Z.Z., B.Z., Y.Z., J.H., Y.X. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: The research in this article was funded by the National Natural Science Foundation of China, grant number 51709147; the project funded by China Postdoctoral Science Foundation, grant number 2021M701713; the Central University Special Funding for Basic Scientific Research, grant No. 30918012201.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Zhang, W.; Qi, J.; Wan, P.; Wang, H.; Xie, D.; Wang, X.; Yan, G. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sens.* **2016**, *8*, 501. [[CrossRef](#)]
2. Chen, H.; Wei, M.; Sun, Y.; Xie, X.; Wang, J. Multi-Patch Collaborative Point Cloud Denoising via Low-Rank Recovery with Graph Constraint. *IEEE Trans. Vis. Comput. Graph.* **2020**, *26*, 3255–3270. [[CrossRef](#)] [[PubMed](#)]
3. Riveiro, B.; González-Jorge, H.; Martínez-Sánchez, J.; Díaz-Vilario, D.; Arias, P. Automatic detection of zebra crossings from mobile LiDAR data. *Opt. Laser Technol.* **2015**, *70*, 63–70. [[CrossRef](#)]
4. Leal, E.; Sanchez-Torres, G.; Branch, J.W. Sparse Regularization-Based Approach for Point Cloud Denoising and Sharp Features Enhancement. *Sensors* **2020**, *20*, 3206. [[CrossRef](#)]
5. Hu, C.; Pan, Z.; Li, P. A 3D Point Cloud Filtering Method for Leaves Based on Manifold Distance and Normal Estimation. *Remote Sens.* **2019**, *11*, 198. [[CrossRef](#)]
6. Der, S.; Redman, B.; Chellappa, R. Simulation of error in optical radar range measurements. *Appl. Opt.* **1997**, *36*, 6869–6874. [[CrossRef](#)]
7. Irfan, M.A.; Magli, E. Joint Geometry and Color Point Cloud Denoising Based on Graph Wavelets. *IEEE Access* **2021**, *9*, 21149–21166. [[CrossRef](#)]
8. Zhou, S.; Liu, X.; Wang, C.; Yang, B. Non-iterative denoising algorithm based on a dual threshold for a 3D point cloud. *Opt. Lasers Eng.* **2020**, *126*, 105921. [[CrossRef](#)]
9. Li, R.; Yang, M.; Ran, Y.; Zhang, H.; Jing, J.; Li, P. Point Cloud Denoising and Simplification Algorithm Based on Method Library. *Laser Optoelectron. Prog.* **2018**, *55*, 251–257.
10. Su, B.; Ma, J.; Peng, Y.; Sheng, M. Algorithm for RGBD Point Cloud Denoising and Simplification Based on K-mean Clustering. *J. Syst. Simul.* **2016**, *28*, 2329–2334.
11. Zhao, K.; Xu, Y.; Li, Y.; Wang, R. Large-Scale Scattered Point-Cloud Denoising Based on VG-DBSCAN Algorithm. *Acta Opt. Sin.* **2018**, *38*, 370–375.
12. Li, X.; Zhang, F.; Wang, Y.; Chen, H. A Research on MBES data classification denoising algorithm based on Octree index. In *IOP Conference Series: Earth and Environmental Science*; IOP Publishing: Bristol, UK, 2019; Volume 237, p. 022011.
13. Fan, L.; Cai, Y. An Efficient Filtering Approach for Removing Outdoor Point Cloud Data of Manhattan-World Buildings. *Remote Sens.* **2021**, *13*, 3796. [[CrossRef](#)]
14. Fleishman, S.; Drori, I.; Cohenor, D. Bilateral Mesh Denoising. *ACM Trans. Graph.* **2003**, *22*, 950–953. [[CrossRef](#)]
15. Zheng, Y.; Fu, H.; Au, O.; Tai, C. Bilateral Normal Filtering for Mesh Denoising. *IEEE Trans. Vis. Comput. Graph.* **2011**, *17*, 1521–1530. [[CrossRef](#)] [[PubMed](#)]
16. Sun, X.; Rosin, P.L.; Martin, R.; Langbein, F. Fast and Effective Feature-Preserving Mesh Denoising. *IEEE Trans. Vis. Comput. Graph.* **2007**, *13*, 925–938. [[CrossRef](#)]
17. Wang, P.; Liu, Y.; Tong, X. Mesh denoising via cascaded normal regression. *ACM Trans. Graph.* **2016**, *35*, 1–12. [[CrossRef](#)]
18. Duan, C.; Chen, S.; Kovacevic, J. 3D Point Cloud Denoising via Deep Neural Network Based Local Surface Estimation. In Proceedings of the 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 8553–8557.
19. Zeng, J.; Cheng, Y.; Ng, M.; Pang, J.; Yang, C. 3D Point Cloud Denoising Using Graph Laplacian Regularization of a Low Dimensional Manifold Model. *IEEE Trans. Image Process* **2020**, *29*, 3474–3489. [[CrossRef](#)] [[PubMed](#)]

20. Wang, X.; Wu, L.; Chen, H. Denoising of scattered point cloud data based on normal vector distance classification. *J. Jilin Univ., Eng. Technol. Ed.* **2020**, *50*, 278–288.
21. Li, P.; Wu, H.; Jing, J.; Li, R. Noise classification denoising algorithm for point cloud model. *Comput. Eng. Appl.* **2016**, *52*, 188–192.
22. Liu, T.; Du, H. An efficient 3D point cloud data denoising algorithm for ship block visual measurement. *MATEC Web of Conferences.* **2017**, *139*, 41.
23. Zhao, F.; Zhou, M.; Zhao, F.; Zhou, M. Hierarchical point cloud denoising algorithm. *Opt. Precis. Eng.* **2020**, *28*, 1618–1625. [[CrossRef](#)]
24. Zhang, Q.; Guo, X.; Deng, J. Grey relation Entropy Method of Grey Relation Analysis. *Syst. Eng. Theory Pract.* **1996**, *8*, 8–12.
25. Zheng, Z.; Zha, B.; Xuchen, Y.; Yuan, H.; Gao, Y.; Zhang, H. Adaptive Edge Detection Algorithm Based on Grey Entropy Theory and Textural Features. *IEEE Access* **2019**, *7*, 92943–92954. [[CrossRef](#)]
26. Wu, L.; Shi, H.; Chen, H. Denoising of three-dimensional point data based on classification of feature information. *Opt. Precis. Eng.* **2016**, *24*, 1465–1473.
27. Chen, X.; Zhang, G.; Hua, X. Point Cloud Simplification Based on the Information Entropy of Normal Vector Angle. *Chin. J. Lasers.* **2015**, *42*, 336–344.
28. Zhang, N.; Sun, J.; Jiang, P.; Liu, D.; Wang, P. Pose estimation algorithms for lidar scene based on point normal vector. *Infrared Laser Eng.* **2020**, *49*, 171–178.
29. Otsu, N. A threshold selection method from gray-level histogram. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [[CrossRef](#)]
30. Zheng, Z.; Zha, B.; Yuan, H.; Xuchen, Y.; Gao, Y.; Zhang, H. Adaptive Edge Detection Algorithm Based on Improved Grey Prediction Model. *IEEE Access* **2020**, *8*, 102165–102176. [[CrossRef](#)]
31. Deng, J. Grey information space. *J. Grey Syst.* **1985**, *1*, 103–117.
32. Zheng, Z.; Zha, B.; Zhang, H. Target Recognition Method of Laser Line Scanning Imaging Fuze Based on DHGF Algorithm. *Chin. J. Lasers.* **2018**, *45*, 147–154.
33. Liu, S.; Yang, Y.; Xie, N.; Forrest, J. New progress of Grey System Theory in the new millennium. *Grey Syst. Theory Appl.* **2016**, *6*, 2–31. [[CrossRef](#)]
34. Chaudhury, K. Acceleration of the shiftable $O(1)$ algorithm for bilateral filtering and nonlocal means. *IEEE Trans. Image Process.* **2013**, *22*, 1291–1300. [[CrossRef](#)]
35. 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](#)]
36. Liang, X.; Liang, J.; Guo, C.; Cao, J. Study on Scatter Point Cloud Denoising Technology Based on Self-adaptive Optimal Neighborhood. *China Mech. Eng.* **2010**, *21*, 639–643.
37. Yadav, S.K.; Reitebuch, U.; Polthier, K. Robust and High Fidelity Mesh Denoising. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 2304–2310. [[CrossRef](#)]