Received August 13, 2019, accepted August 30, 2019, date of publication September 5, 2019,
date of current version September 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2939684

A Novel Simplification Method for 3D Geometric Point Cloud Based on the Importance of Point

CHUNYANG JI[1,2], YING LI[2,3], JIAHAO FAN[2,3], AND SHUMEI LAN[2,3]

[1]Software Institute, Jilin University, Changchun 130012, China
[2]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China
[3]College of Computer Science and Technology, Jilin University, Changchun 130012, China

Corresponding author: Jiahao Fan (jihanfan@hotmail.com)

ABSTRACT 3D point cloud simplification is an important pretreatment in surface reconstruction for sparing computer resources and improving reconstruction speed. However, existing methods often sacrifice the simplification precision to improve the simplification speed, or sacrifice the speed to improve precision. A proper balance between the simplification speed and the simplification accuracy is still a challenge. In this paper, we propose a new simplification method based on the importance of point. Named as detail feature points simplified algorithm (DFPSA), this algorithm has distinct processes to achieve improvements in three aspects. First, a rule of k neighborhood search is set to ensure the points found are the closest to the sample point. In this way, the accuracy of calculated normal vector of the point cloud is significantly improved, and the search speed is largely increased. Second, a formula that considers multiple characteristics for measuring the importance of point is proposed. Thereupon, the main detail features of the point cloud are preserved. Finally, an octree structure is employed to simplify the remaining points, through which holes in reconstructing point cloud are obviously reduced. The DFPSA is applied to four different data sets, and the corresponding results are compared with those of other five algorithms. The experimental results demonstrate that the DFPSA brings better simplification effects than existing counterparts, and the DFPSA not only can simplify point cloud but also has good effect in simplifying subject's narrow contours.

INDEX TERMS 3D geometric point cloud simplification, detail feature point, k neighborhood search rule, the importance of point.

I. INTRODUCTION

With the development of sensor technology [1], point cloud, a data form collected by sensors, is gaining increasing attention. Point cloud data have been used in many fields, such as monitor structural behavior [2], road curb detection [3], building information modelling [4], agricultural application [5]. However, sensors often collect large amounts of point information without clear association among points, this situation causes heavy workload in point cloud processing. Hence, point cloud simplification is one of the most important pretreatment steps in point cloud processing.

A lot of researches have been done about point cloud simplification, and existing methods for point cloud simplification are divided into two kinds: mesh-based simplification and direct simplification.

The mesh-based simplification methods build irregular meshes based on point cloud, and then remove redundant meshes by a given rule to achieve the goal of simplification. Luebke [6] used mesh compression to simplify point cloud. Hur *et al.* [7] used Delaunay triangulation to remove extra point data. Medial meshes is used by Sun *et al.* [8]. The mesh-based simplification can effectively preserve the structure of point cloud, but the overhead of building meshes on computer is substantive, especially when point cloud data are huge, in which case the calculation cost could be particularly high.

Direct simplification, on the other hand, aims to simplify point cloud according to the characteristics of points. Clustering is a direct simplification algorithm. Shi *et al.* [9] proposed an adaptive simplification of point cloud using k-means clustering. Chang *et al.* [10] used k-means clustering based on boundary reservation to reduce point cloud. Additionally, Song and Feng [11] used a global clustering to simplify point cloud. Of the direct simplification methods, spatial index is an important method to process point cloud. Shi *et al.* [12] used kd-tree to obtain spurious trails and updated point cloud map. Goswami *et al.* [13] used kd-tree to process points.

The associate editor coordinating the review of this manuscript and approving it for publication was Huanqiang Zeng.

Spatial index is also used to directly simplify point cloud. Shao and Xi [14] and Song *et al*. [15] used octree to simplify point cloud. Li *et al*. [16] put forward a uniform simplification method based on space bounding box. There are also other direct simplification algorithms. Xuan *et al*. [17] constructed normal angle local entropy to assess point, and simplified point cloud in a gradual way. Zang *et al*. [18] presented a multi-level method for retaining geometric features of different sizes. Han *et al*. [19] proposed a simplified algorithm based on edge-preserving. Yuan*et al*. [20] used a conformal geometric algebra approach to reduce point. Wei *et al*. [21] simplified point cloud based on curvature co-occurrence histograms. Abzal *et al*. [22] developed a simplification mask for multi-shot optical scanners to reduce point during the data acquisition phase. Although the direct simplification method can reduce the calculation overhead, retaining features of what becomes a significant challenge.

Generally, the point cloud data collected by sensor only contain the three-dimensional coordinates of point without topological information. In order to determine the geometric information of the given point, it is necessary to collect neighborhood information. Usually, k neighborhood is used to calculate neighborhood relationship [23]. The k points with the shortest distance from the sample point in the spatial Cartesian coordinate system are treated as the k neighborhood of the sample point. In this process, the commonly used methods to find neighborhood are space bounding box [24] and kd-tree [25]. According to the neighborhood information of point cloud, the normal vector of each point can be calculated, and then the geometric information of each point is obtained. To calculate the normal vector, PCA (Principal Component Analysis) and MLS (Moving Least Square) are often used [26]. Of two methods, PCA is efficient, but it is not easy to calculate the normal vector precisely when there are large errors in neighborhood information. In contrast, MLS is accurate, whereas it sacrifices efficiency.

Overall, the aforementioned methods of point cloud simplification see room for improvement in the following aspects. First, most ways to find k neighborhood use fixed direction, which sometimes result in generating inaccurate neighborhood of the sample and may produce large deviations of the calculation of normal vector. Whereas normal vector is an important indicator for evaluating point cloud, an inaccurate calculation of normal vector directly leads to an inaccurate evaluation of the importance of the point. Second, the methods of point evaluation are often simple. For example, only the normal vector of point is considered, whereas other information such as curvature at point is not considered. This may lead to an inaccurate evaluation of the importance of point and may subsequently compromise the simplification effect. Third, the same simplification method is often used for entire point cloud without considering the difference between feature areas and non-feature areas, which results in either an elongated simplification time or poor results.

In order to solve the above problems, a new algorithm, named as detail feature points simplified algorithm (DFPSA), is proposed and trialed. First, a space bounding box is constructed to find the neighborhood of the sample quickly. In order to ensure the identified k neighborhood is the nearest k points apart from the sample, a method for finding k neighborhood based on distance and density is proposed. Second, four characteristic operators are calculated to describe the importance of point: the difference between the normal vector of the sample and every normal vector of the sample's neighbor (the normal vector difference), the projection distance from the neighbor point to the tangent plane of the sample (the projection distance), the spatial distance between the sample and every neighborhood (the spatial distance), the difference between the curvature at the sample and the curvature at every sample's neighborhood (the curvature difference). The larger the normal vector difference is, the greater the degree of mutation of the sample is. Also, the larger the projection distance is, the greater the curvature changed. Likewise, the greater the spatial distance is, the farther apart of two points are. Moreover, the greater the curvature difference is, the sharper the sample point is. Based on the four formulas above, the importance of the sample is high. In other words, the sample is a detail feature point. Therefore, these four characteristic operators are added up to determine the importance of the sample. Third, the DFPSA uses octree to simplify non-feature points. In each octree leaf node, the DFPSA saves one most representative point and deletes the others. Given the straightforward structure of octree is simple, simplifying the non-feature points is easy. In summary, the DFPSA not only guarantees the simplification speed, but also saves the detail feature points. Thus, the DFPSA ensures the accuracy of point cloud simplification.

In the following sections, the related work is introduced in section II. Key steps of the proposed algorithm, which include the k neighborhood search rule based on distance and density, the evaluation of feature point, and the simplification of non-feature points are elaborated in section III. The whole procedure of the DFPSA is introduced in section IV. The experimental results about this paper and the contrasts with other related methods are provided in section V. The conclusion is in section VI.

II. RELATED WORK

In this section, we briefly introduce the previous studies on which the DFPSA depends.

A. CONSTRUCTING SPACE BOUNDING BOX

The purpose of using space bounding box is to find the neighborhood of points more easily. Huang *et al*. [27] used space bounding box to divide point cloud space, and then constructed a space bounding sphere to find k neighborhood of each point. Xiong *et al.* [24] proposed a fast search algorithm for k-neighborhood. When calculating the edge length of a cube, it not only considered the size of data set, but also introduced the number of neighborhood. This algorithm has strong generality and adaptability. Li *et al*. [16] created a k neighborhood three-dimensional voxel grid, and then combined space bounding box to find k neighborhood.

The steps to construct a space bounding box [24] are as follows. First, based on the maximum and minimum of point cloud coordinate information, the biggest bounding box, which surrounds all the points is constructed. Second, the large bounding box is divided into several small cubes evenly and the side length of each cube is calculated. Small cubes may include many points or even empty. The side length of cube is L.

$$L = \sqrt[3]{\omega} \times \sqrt[3]{(k/n) \times (x_{\max} - x_{\min}) \times (y_{\max} - y_{\min}) \times (z_{\max} - z_{\min})} \tag{1}$$

k is the number of neighborhood points which are looked for. n is the total number of the point cloud. $x_{\max}$, $x_{\min}$, $y_{\max}$, $y_{\min}$, $z_{\max}$, $z_{\min}$ are maximum and minimum coordinate values of the point cloud respectively. ω is a constant.

B. BUILDING THE RELATIONSHIP

Based on space bounding box, steps are followed in building the relationship between the cube and the point cloud. First, the resolution of the minimum cuboid space in the x, y, and z directions are respectively calculated with the following formulas [24]:

$$N_x = ceil((x_{\max} - x_{\min})/L) \tag{2}$$

$$N_y = ceil((y_{\max} - y_{\min})/L) \tag{3}$$

$$N_z = ceil((z_{\max} - z_{\min})/L) \tag{4}$$

N_x, N_y and N_z are resolution. Therefore, the number of small cubes N_{cube} is shown in (5).

$$N_{cube} = N_x \times N_y \times N_z \tag{5}$$

Then, drawing on the vertex information of each cube and coordinate information of each point, the relationship mapping between the point and the index of cube that contains this point is established.

C. CALCULATING THE NORMAL VECTOR

Normal vector is important geometric information of point cloud. Many studies simplify point cloud based on normal vector. Xuan *et al.* [17] used PCA to calculate the normal vector of each point, and then calculated normal angle local entropy to evaluate the importance of point. They deleted the least important point based on the importance and updated the normal vector to simplify the point cloud recursively. Han *et al.* [19] used the topological relationship between points and normal vectors to extract edge points and simplify point cloud. There are many ways to calculate normal vector. OuYang and Feng [28] proposed a calculating normal vector method based on fitted directional tangent vector. Li *et al.* [29] presented an algorithm for directly calculating normal vector of point cloud with sharp features. In this paper, we use PCA to calculate the normal vector.

First, k neighborhood about the sample p is gotten, the neighborhood of p and the sample p constitute the point set p_{area}. And then the geometric center $\overline{p}$ of the p_{area} and the covariance matrix C are calculated. p_p is a point in p_{area}.

$$\overline{p} = (1/k) \times \sum_{p_p \in p_{area}} p_{area} \tag{6}$$

$$C = \sum_{p_p \in p_{area}} (p_p - \overline{p})(p_p - \overline{p})^T / k \tag{7}$$

Through the covariance matrix C, the eigenvalues $\lambda_j (j = 0, 1, 2)$ of C and the corresponding eigenvectors $v_j (j = 0, 1, 2)$ are calculated. Among them, the smallest eigenvectors v_0 is the normal vector n_p of the sample p.

In addition, Pauly *et al.* [26] proved that surface change $\sigma_n(p)$ is equal to curvature cu_p. Hence, the curvature cu_p is calculated as shown in (8).

$$cu_p = \sigma_n(p) = \lambda_0/(\lambda_0 + \lambda_1 + \lambda_2) \tag{8}$$

D. BUILDING OCTREE

Octree is a way to simplify point cloud. Shao and Xi [14] used octree to divide point cloud space and retained one point in each leaf node. Shao and Xi [14] suggested that the simplification method based on octree considers space as a whole and has a good effect. Since point cloud is partitioned in space before simplification, the algorithm is simple and efficient. That is to say the octree can simplify point cloud clearly and quickly. Song *et al.* [15] proposed a robust simplified algorithm based on octree. Yu *et al.* [30] presented a simplification method based on point saliency. They determined an octree subdivision criterion and then constructed an octree to simplify point cloud.

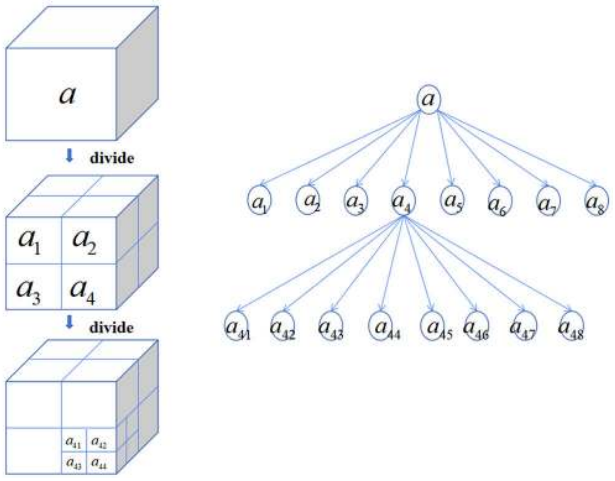

FIGURE 1. Schematic diagram of an octree structure.

In this paper, we use octree to simplify non-feature points. A simple schematic diagram of an octree structure [31] is shown in Fig. 1. On the left side in Fig. 1, cube a is the smallest outer cube that surrounds the point cloud. And according to the partition conditions, a is divided into eight-molecule cube a_1 - a_8, then the sub-cube a_4 is continue divided to eight cubes a_{41} - a_{48}. On the right side in Fig. 1, the tree structure

of the octree is shown. Circle a is the root node of the octree and it is the representation of cube a in the tree structure. Similarly, the remaining nodes are also representations of the corresponding cubes in the tree structure.

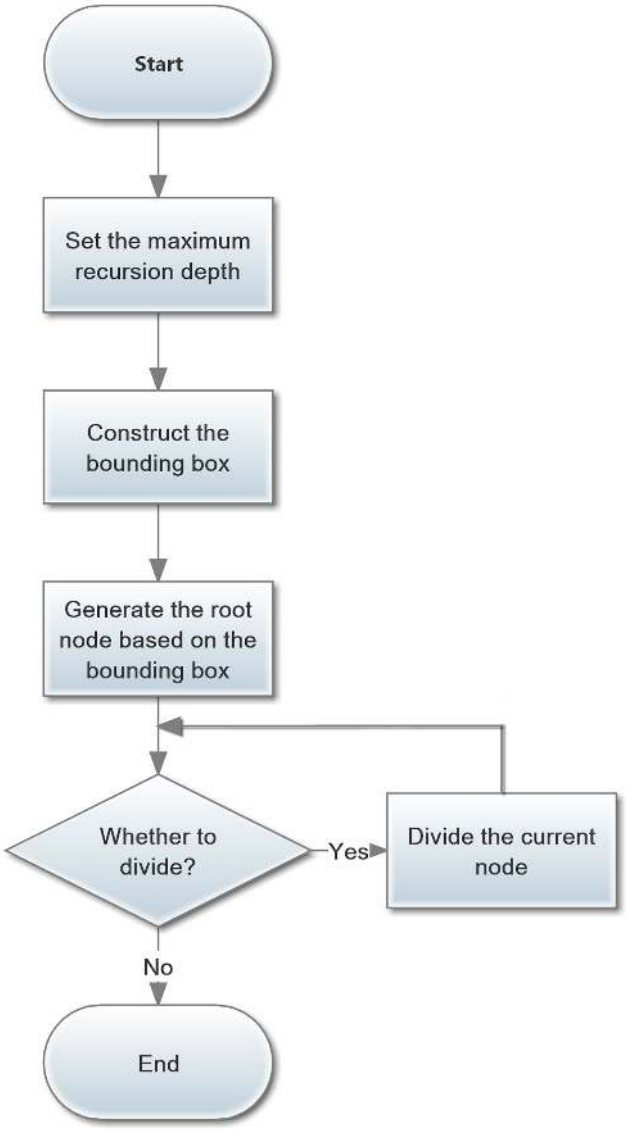

FIGURE 2. Flowchart of constructing an octree.

The steps of constructing an octree are shown in Fig. 2. First, the maximum recursion depth, or the maximum number of layers for the octree is set. Second, the maximum and minimum spatial coordinate values of the overall point cloud are found, and the smallest outer cube that surrounds the point cloud is built, this outer cube is the space bounding box. Third, based on the space bounding box, the root node is generated. Fourth, if the maximum recursion depth is not reached, the current cube is evenly subdivided into eight. Fifth, if the number of points in the child cube is as same as the number of points in the father cube and the number is not zero, the child cube is stop cut up. Sixth, the fourth step is repeated until the maximum recursion depth is reached.

III. METHODOLOGY

The DFPSA features three key steps: searching for k neighborhood, evaluating importance of each point, and simplifying non-feature points.

A. SEARCHING FOR K NEIGHBORHOOD

The traditional k neighborhood search methods usually follow the fixed search direction, and as result the points found are often not the closest ones to the sample. Fig. 3 shows an example about searching for k neighbor using a fixed direction. p is located in $cube_5$, the fixed direction method searches cubes consecutively from left to right, and then expands the search direction from front to back. That is, the search order is $cube_1$ - $cube_2$ - $cube_3$ - $cube_4$ - $cube_5$ - $cube_6$ - $cube_7$ - $cube_8$ - $cube_9$. In this process, as long as the k neighborhood points of the sample p are found, the subsequent cubes are no longer searched. In Fig. 3, it can be seen that most of the neighborhood points of sample p are probably located in $cube_5$, $cube_6$, $cube_8$, and $cube_9$, whereas searching method using a fixed direction may not reach these cubes.

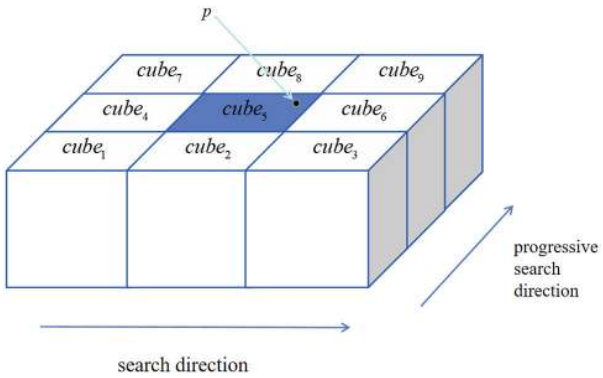

FIGURE 3. Searching for k neighborhood using a fixed direction.

In order to have a thorough search of the neighborhood, a new search method is employed. First, a space bounding box is constructed, and this space bounding box is divided according to the principle of the space bounding box. Second, based on the coordinate information of each cube, an index of cubes is obtained. Third, the relationship between the space bounding box and each point is built. Fourth, the cube which includes the sample p is obtained, and the cube index is c_p.

Since a space bounding box is made up of cubes of the same size, in addition to the cubes at boundary position, each cube c_{pi} has 26 adjacent cubes. Of these 26 cubes, 6 cubes are adjacent to the six faces of c_{pi} respectively, and the remaining 20 cubes only share edges or points with c_{pi}.

In this context, there are two situations: c_p is not a boundary cube and c_p is a boundary cube. Therefore, we discuss the implementation of fifth step from the following two situations.

Case one, c_p is not a boundary cube. First, c_p is searched. If we do not obtain k neighborhood points in c_p, we calculate a series of distances from the sample p to the six faces of the cube c_p, and sort these distances from small to large at

the same time. Then according to the distances sort order, searches the 6 cubes in turns which are adjacent to the six faces of c_p. When these 6 cubes have been searched, but the k neighborhood points of the sample p are not received, we expand our search area. Hence, the point cloud densities of the remaining 20 cubes which only share edges or points with c_p are calculated. At the same time, we sort these densities from large to small and continue searching these cubes based on density sorting. When the 20 cubes have been searched without finding the k neighborhood of the sample p, we continue search cubes adjacent to 26 cubes by density from large to small.

Case two, c_p is a boundary cube. The k neighborhood search rule about boundary cube is the same as the k neighborhood search rule which is searches non-border cube. But if the searched cube is absent, we skip this cube and continue searching next cube.

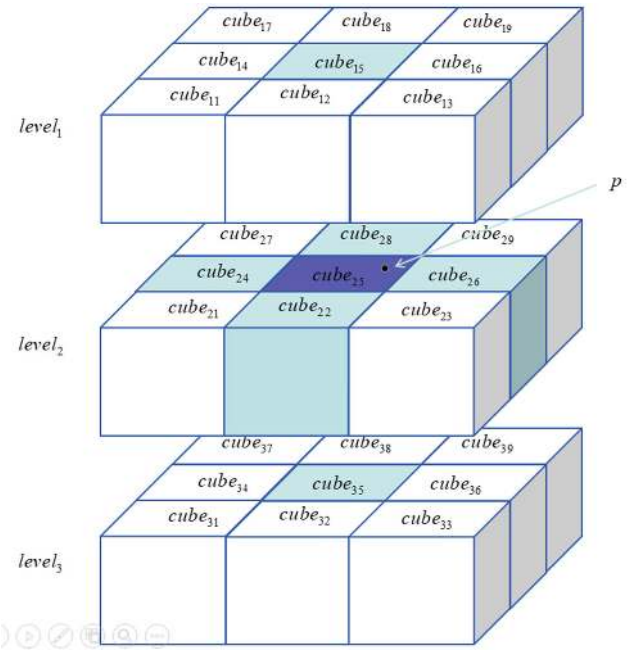

FIGURE 4. Searching for k neighborhood using our method.

Fig. 4 shows an example of using our searching for k neighborhood method. In order to describe our method clearly, we extract a small part of the space bounding box, and embody it hierarchically as Fig. 4 shows. The sample point p is located in $cube_{25}$. The light blue cubes are directly adjacent to the six faces of $cube_{25}$, while the white cubes only share points or edges with $cube_{25}$. We need to find the k neighborhood of the sample p. First, $cube_{25}$ is searched. If the k neighborhood points of p are found in $cube_{25}$, the searching process is stopped. Otherwise, a series of distances from the sample p to the six faces of the $cube_{25}$ are calculated, and these distances are sorted from small to large at the same time. We use the indexes of six cubes directly adjacent to $cube_{25}$ to represent the distances from the sample p to the six faces of $cube_{25}$. Assume that the order of distances from small to large is: $cube_{15}$ - $cube_{28}$ - $cube_{26}$ - $cube_{22}$ - $cube_{24}$ - $cube_{35}$, and the corresponding cubes are searched according to this order. In the process of searching, if the k neighborhood points of p are searched, the searching process is stopped. If the k neighborhood points of p are not found after searching these six cubes, the densities of points in the cubes which only share points or edges with $cube_{25}$ are calculated and sorted from large to small, this is the search order. We assume that the order of point densities of the cubes only share points or edges with $cube_{25}$ from large to small is as follows: $cube_{29}$ - $cube_{19}$ - $cube_{18}$ - $cube_{16}$ - $cube_{17}$ - $cube_{13}$ - $cube_{27}$ - $cube_{23}$ - $cube_{36}$ - $cube_{38}$ - $cube_{39}$ - $cube_{33}$ - $cube_{37}$ - $cube_{21}$ - $cube_{32}$ - $cube_{31}$ - $cube_{34}$ - $cube_{11}$ - $cube_{14}$ - $cube_{12}$, so the search order is the density order. In the process of searching, if the k neighborhood points of p are searched, the searching process is stopped. If the k neighborhood points of p are not found after searching these 20 cubes, the search scope is extended, and we continue to search cubes adjacent to $cube_{11}$ - $cube_{39}$ (except $cube_{25}$) in order of point densities from large to small. Stated thus, one possible search order of Fig. 4 is: $cube_{25}$ - $cube_{15}$ - $cube_{28}$ - $cube_{26}$ - $cube_{22}$ - $cube_{24}$ - $cube_{35}$ - $cube_{29}$ - $cube_{19}$ - $cube_{18}$ - $cube_{16}$ - $cube_{17}$ - $cube_{13}$ - $cube_{27}$ - $cube_{23}$ - $cube_{36}$ - $cube_{38}$ - $cube_{39}$ - $cube_{33}$ - $cube_{37}$ - $cube_{21}$ - $cube_{32}$ - $cube_{31}$ - $cube_{34}$ - $cube_{11}$ - $cube_{14}$ - $cube_{12}$.

There is a special case where when a single point is located exactly at the border between two (or more) sub-cubes, the point belongs to the first sub-cube searched. That is to say, when searching a cube, we not only search for the points which are surrounded by it, but also search for the points at its boundary. When a cube has been searched, the points located in or at it are temporarily deleted from the point cloud data. Thus, a single point is located at the border between two (or more) sub-cubes can be thought of as is located only at the first sub-cube searched.

The above method makes full use of the structure of the space bounding box. After sub-cubes are used to divide the point cloud, the information of each point is mapped to the information of the sub-cube. Depending on the coordinates of the sample point, the coordinates of the cube, and the length of the cube, we can quickly get the index of the sample cube where the sample point is located, and then find other points in the sample cube or the points in the adjacent cubes of the sample cube. These points are obviously close to the sample point, and the neighborhood points of the sample point also exist in these points. Therefore, when searching for the neighborhood of the sample, it is no longer necessary to traverse the entire point cloud data. We only need to search some specific small cubes, thereby improving the search speed. In addition, our method has priority level, which thus increases accuracy of the found k neighborhood. Also, density search after distance search saves time. Therefore, this method not only accurately finds k neighborhood, but also improves efficiency. At the same time, this way lays the foundation for an accurate calculation of normal vector.

B. EVALUATING IMPORTANCE

The traditional method to retain feature points is relatively simple, for example, only the curvature differences between the sample and the neighborhood points are considered. In order to accurately retain the detail feature points of the overall point cloud, a new method for measuring the importance of the point is employed. We calculate four characteristic operators to reflect the importance of the point: the normal vector difference, the projection distance, the spatial distance, and the curvature difference.

Actually, the normal vector of the point reflects the tangent plan where the point is located. If two points have the same tangent plane, their normal vectors are the same and the normal vector difference between these two points is zero. In contrast, the larger the difference between the two normal vectors is, the less likely they are on the same tangent plane. Furthermore, the neighborhood points reflect the plane in which the sample is located. Therefore, when the sum of the difference between the normal vector of the sample point and the normal vector of its neighbor points is larger, the surface fitted by the sample and its neighbor points is more convex at the sample point. So this sample point is likely to be a feature point, in other words, the importance of the sample is high. Based on this, we calculate the normal vector n_p of the sample p, the normal vector $n_j(1 \leq j \leq k)$ of the $p's$ neighborhood point $p_j(1 \leq j \leq k)$, and then we calculate the difference between n_p and n_j, finally add up these differences.

In addition that the normal vector difference can reflect the importance of the point, the projection distance from the point to the plane which formed by its neighborhood points also reflects the point's concavity and convexity, thus reflects the importance of the point. When the surface is smooth, the sample point and the neighborhood points are almost on the same plane, the projection distance from the neighbor point to the tangent plane of the sample is almost close to zero. At the same time, the larger the projection distance is, the more the sample curvature changes. So, in order to reflect the change of the sample curvature, we get the k neighborhood $p_j(1 \leq j \leq k)$ of the sample p, the tangent plane p_{plan} of p, and then we calculate the projection distance for $p_j(1 \leq j \leq k)$ to p_{plan}, finally add up these projection distances.

Moreover, the spatial distance reflects the relationship between two points. In point cloud data set, when the spatial distance of the sample point and its neighborhood is large, there are two cases: Case one, this sample point is in sharp position. Thus, this sample point is a detail feature point, and the importance of this sample point is high. Case two, sampling of point cloud is not uniform, resulting in a sparse area near the sample point. In the process of surface reconstruction, holes will appear in the sparse area. In order to ensure the integrity of the reconstructed surface in the future, this sample point should be retained. Hence, we calculate the spatial distance between the sample p and each neighborhood of p, and then we accumulate these spatial distances in order to reflect the importance of the sample p.

Curvature is a parameter that most intuitively reflects the sharpness of the sample point. When the difference between the curvature at the point and the curvature at its neighbor is very large, this point is very likely in sharp position. When a point in sharp position, it is likely a detail feature point. Hence, we calculate the difference between the curvature at the sample p and the curvature at each $p's$ neighborhood. Then we accumulate these curvature differences in order to reflect the sharpness of the sample p.

In summary, the normal vector difference, the projection distance, the spatial distance, and the curvature difference all can reflect the importance of the point. In order to reduce the error accumulation, the above four characteristic operators are added up to get a new formula which describes the importance of the sample. The new formula is as follows.

$$\sum_{j=1}^{k}\Big[\alpha \times \left(1 - n_p^T n_j\right) + \beta \times \left|n_p^T (p - p_j)\right| + \gamma \times \left\|p - p_j\right\| + \delta \times \left|cu_p - c_j\right|\Big] \quad (9)$$

$\alpha, \beta, \gamma, \delta(\alpha > 0, \beta > 0, \gamma > 0, \delta > 0$ and $\alpha+\beta+\gamma+\delta = 1)$ are scale factors, p is the sample point, $p_j(1 \leq j \leq k)$ is the neighborhood of p, n_p is the normal vector of the sample point p, $n_j(1 \leq j \leq k)$ is the normal vector of $p_j(1 \leq j \leq k)$, cu_p is the curvature at the sample p, $c_j(1 \leq j \leq k)$ is the curvature at $p_j(1 \leq j \leq k)$.

For different types of point clouds, different combinations of scale factors can be given. For the point cloud with uniform sampling, the scale factor γ of the spatial distance operator is set to a small value. For the point cloud with non-uniform sampling, the scale factor γ of the spatial distance operator is set to a large value. For the point cloud with sharp detail features, the scale factor δ of the curvature difference operator is set to a large value. For the point cloud with general detail features, the scale factor α of the normal vector difference operator, the scale factor β of the projection distance operator, and the scale factor δ of the curvature difference operator are set to a large value, respectively. In general, when setting the scale factors, it is necessary to combine different situations and project experiences and then give different combinations.

Furthermore, we preset a threshold. A point which importance is greater than the threshold is reserved as a detail feature point. And the remaining points are simplified in the next step. For the point cloud data with a few features, we set a large threshold to ensure a high simplification rate. For the point cloud data with a large number of features and complex features, we set a small threshold to ensure that the algorithm retains much detail feature points.

C. SIMPLIFYING NON-FEATURE POINTS

Based on the importance of each point, the detail feature points have been obtained and retained. At the same time, the detail feature points describe the detail features, contour features, and structure features of the entire point cloud model. However, if fitting the surface only using the feature points, there will be a lot of holes in the surface, indicating

that some non-feature points should be preserved. For this purpose, an octree is used to simplify the non-feature points.

First, an octree is constructed for all non-feature points. Second, in each octree leaf node, for all points located in it, the average normal vector $\overline{n_{non}}$ of these points and the average curvature $\overline{cu_{non}}$ at these points are calculated. Third, in each octree leaf node, we calculate the difference between the normal vector of each point and the average normal vector. Fourth, we calculate the difference between the curvature at each point and the average curvature. Fifth, we add up the normal vector difference and the curvature difference, hereafter, the point with the smallest sum value is selected to replace the other points in the leaf node. This normal vector difference and this curvature difference are different from the normal vector difference and the curvature difference in III. B. When selecting feature points, for each point in the point cloud, we focus on the relationship between the point and its neighborhood. The relationship can determine the importance of the sample point, and then the feature points can be selected. So, we use four characteristic operators consisted of sample point and its neighborhood to filter feature points. When selecting non-feature points, we use the structure of octree, and the non-feature points in the same leaf node are spatially close. For the non-feature points in the same leaf node, we pay more attention to the relationship between the point and the whole leaf node. Therefore, the average normal vector and the average curvature are used to describe the whole leaf node. And then the non-feature points are simplified based on the method mentioned in III. C. In summary, the selection of feature points and the selection of non-feature points interest different perspectives, so different methods are used for selection.

$$(1 - n_{non\_i}{}^{T}\overline{n_{non}}) + (cu_{non\_i} - \overline{cu_{non}}) \tag{10}$$

n_{non_i} is the normal vector of a point *non_i* in a leaf node, cu_{non_i} is the curvature at a point *non_i* in a leaf node. The point *non_i* which can minimize (10) is selected to replace the other points in the leaf node.

In general, the non-feature points have a small amount of point cloud model information, hence, selecting one typical point to replace the other points in each octree leaf node simplifies point cloud to the utmost.

IV. THE OVERALL PROCESS OF THE DFPSA

Fig. 5 illustrates the process of the DFPSA. First, the space bounding box for the overall point cloud is constructed. Second, the relationship between the space bounding box and the point cloud is built. Third, we search for k neighborhood of each point based on distance and density. Fourth, we calculate the normal vector of each point and the curvature at each point using PCA. Fifth, the feature points are selected based on the four characteristic operators. At the same time, the feature points are saved to the feature points set and the non-feature points are saved to the non-feature points set. Sixth, the non-feature points are simplified using an octree.

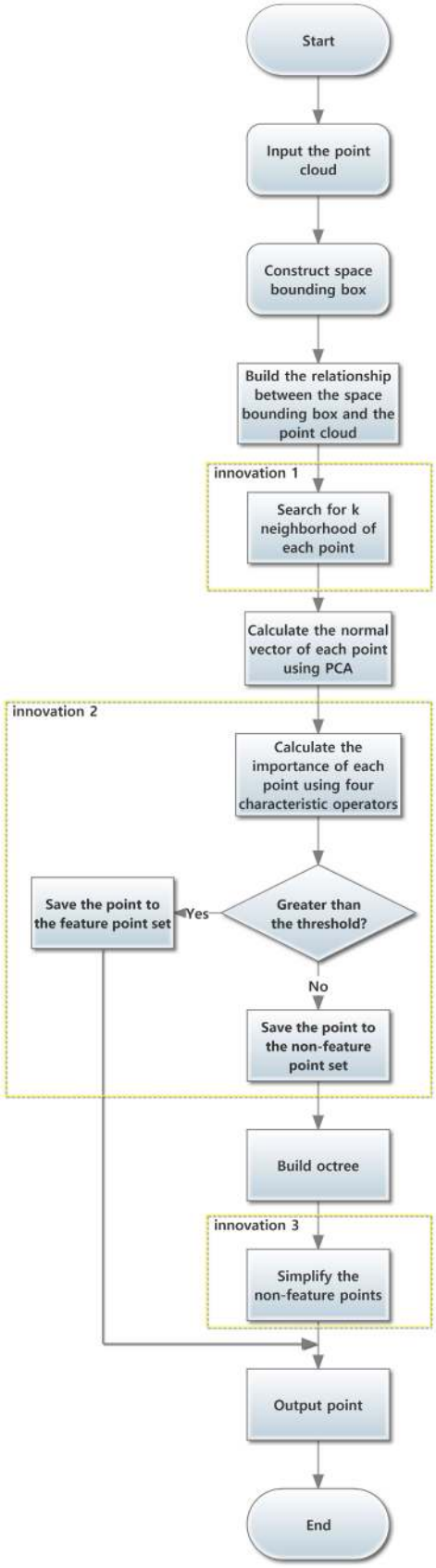

FIGURE 5. Flowchart of the DFPSA.

Seventh, we output the feature points set and the simplified non-feature points.

V. EXPERIMENT AND ANALYSIS

In order to evaluate the performance of the DFPSA, we have applied the algorithm to bunny data set (has 35947 points and a few detail features), horse data set (has 48485 points and a few detail features), gargo50K data set (has 25038 points and many detail features), and elephant data set (has 24955 points and many detail features). We also have verified the effectiveness of the DFPSA. Moreover, the simplified results of the proposed algorithm have been compared with the results of five simplified methods: the simplified method based on Gaussian spheres, the simplified method based on octree coding, the k-means clustering simplified method based on boundary reservation [10], the uniform simplification method [16], and the conformal geometric algebra method [20]. The data processing platform is windows10 on laptop PC 1.7GHz processor and 4GB memory.

A. SETTING PARAMETERS

Equation (9) is the core formula for selecting feature points. It has four parameters α, β, γ and δ. Different parameters have different effects on different types of point clouds.

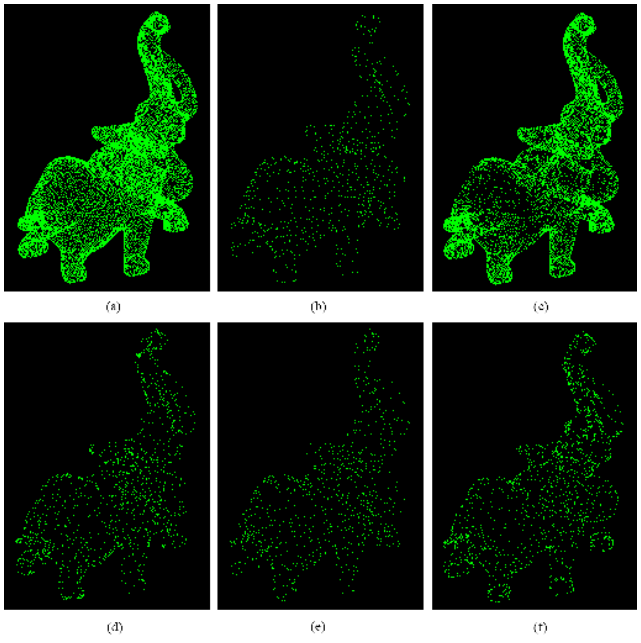

FIGURE 6. The effect of different scale factors on the same point cloud data set. (a) Original data, total number of the points = 24955, (b) Without screening feature points, treat all points as non-feature points, $\alpha = 0, \beta = 0, \gamma = 0, \delta = 0$, total number of the points = 1793, (c) $\alpha = 1, \beta = 0, \gamma = 0, \delta = 0$, total number of the points = 17582, (d) $\alpha = 0, \beta = 1, \gamma = 0, \delta = 0$, total number of the points = 2710, (e) $\alpha = 0, \beta = 0, \gamma = 1, \delta = 0$, total number of the points = 1793, (f) $\alpha = 0, \beta = 0, \gamma = 0, \delta = 1$, total number of the points = 3026.

Fig. 6 shows the effect of different scale factors on elephant data set. Fig. 6b regards all points as non-feature points, that is to say that no feature points are selected. Fig. 6c reserves most of the points in the legs, nose and ivory while its simplification rate is not high. Although Fig. 6d retains a few feature points, we can see that it still retains some features in the back, soles of feet and nose. The simplification result of Fig. 6e is exactly the same as that of Fig. 6b, that is to say, spatial distance does not screen out feature points. This is because elephant data set is a point cloud with uniform sampling, so spatial distance has little effect on feature selection. It can be clearly seen that Fig. 6f has obvious contour details, ear edge, foot edge and nose edge are completely reserved. This is because the curvature difference can select sharp detail feature points. To sum up, for the data set with uniform sampling and sharp features, the scale factor γ of the spatial distance operator is set to a small value, the scale factor δ of the curvature difference operator is set to a large value, the scale factor α of the normal vector difference operator and the scale factor β of the projection distance operator are set to moderate values.

According to the actual application project and experience, the algorithm parameters have been set as follows: $k = 7$, $\alpha = 0.1$, $\beta = 0.1$, $\gamma = 0.2$, $\delta = 0.6$, the maximum recursion depth of octree $= 5$. For bunny data set and horse data set, the threshold of the screening feature points $= 0.60$. For gatgo50K data set and elephant data set, the threshold $= 0.45$.

TABLE 1. The simplification rate (in percent) of simplification algorithms.

Simplification rate / %	Bunny	Horse	Gargo50K	Elephant
The DFPSA	81.3	83.3	49.7	66.0
The simplified method based on Gaussian spheres	76.4	79.3	35.5	65.6
The simplified method based on octree coding	91.6	94.5	90.4	90.0
The k-means clustering simplification method [10]	51.6	60.3	7.6	36.6
The uniform simplification method [16]	87.4	91.7	84.4	88.6
The conformal geometric algebra method [20]	84.9	85.2	84.6	84.4

B. SIMPLIFICATION RESULTS

Table. 1 shows the simplification rates of six simplified algorithms which are applied to the four data sets. For different types of data sets, both the simplified method based on octree coding and the uniform simplification method [16] achieve high simplification rates. Whereas the DFPSA and the simplified method based on Gaussian spheres both have high simplification rates to the point cloud with a small number of feature points, on the contrary, have a low simplification

rates to the point cloud with a large number of feature points to ensure the integrity of the point cloud features. Particularly, the simplification rate of our algorithm is higher than that of the simplified method based on Gaussian spheres. In addition, the k-means clustering simplification method [10] has the worst simplification ratio.

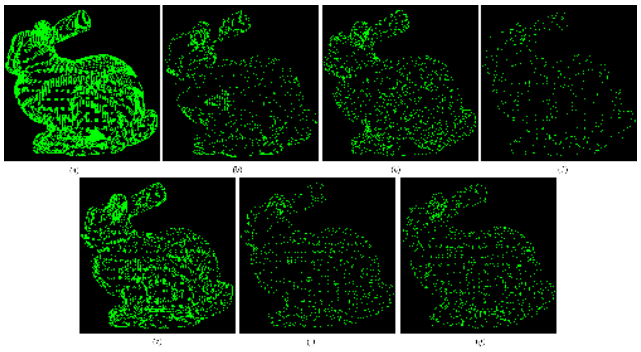

FIGURE 7. Simplified results of bunny. (a) Original data, total number of the points = 35947, (b) Our algorithm, total number of the points = 6730, (c) The simplified method based on Gaussian spheres, total number of the points = 8491, (d) The simplified method based on octree coding, total number of the points = 3005, (e) The k-means clustering simplification method [10], total number of the points = 17385, (f) The uniform simplification method [16], total number of the points = 4539, (g) The conformal geometric algebra method [20], total number of the points = 5434.

Fig. 7 shows the simplified results of bunny data set. In Fig. 7d and Fig. 7f, although the simplification rate of each model is respectively as high as 91.6% and 87.4%, the results only retain the contour points of the rabbit and the detail features are seriously lost. In Fig. 7b, Fig. 7c, Fig. 7e, and Fig. 7g, the simplification rate is 81.3%, 76.4%, 51.6%, and 84.9% respectively. Each result retains many detail features. In particular, compared with Fig. 7c, Fig. 7e, and Fig. 7g, Fig. 7b retains more points at the detail feature parts, such as ears, neck, and bottom. At the same time, at relatively smooth areas, such as back, the number of points is significantly less. Besides, the simplification rate of our result is 81.3%, which meets the criteria of the simplification and is higher than the simplified method based on Gaussian spheres and the k-means clustering simplification method [10]. Moreover, there is little difference between the simplification rate of our result and that of the conformal geometric algebra method [20].

In order to verify the validity of the algorithm, we have used Geomagic Studio to fit the results of the point cloud simplification. The reconstruction results are shown in Fig. 8. It can be seen that these six methods all have good reconstruction results at the smooth region, such as at the region of back. However, at some places with many detail features, the reconstruction results of the DFPSA, the simplified method based on Gaussian spheres, and the k-means clustering simplification method [10] are more complete and finer than others. Especially at the ear part of the rabbit, the result of the method based on Gaussian spheres has obvious empty hole, in contrast, the result of the DFPSA and the k-means clustering

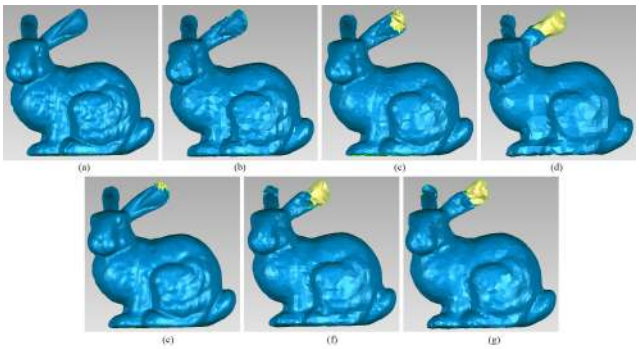

FIGURE 8. Reconstruction results of bunny. (a) Original data, (b) Our algorithm, (c) The simplified method based on Gaussian spheres, (d) The simplified method based on octree coding, (e) The k-means clustering simplification method [10], (f) The uniform simplification method [16], (g) The conformal geometric algebra method [20].

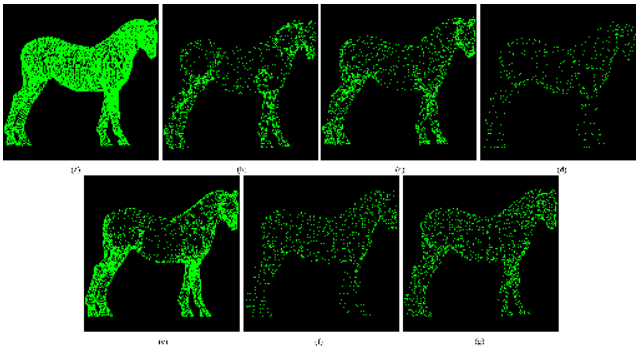

FIGURE 9. Simplified results of horse. (a) Original data, total number of the points = 48485, (b) Our algorithm, total number of the points = 8107, (c) The simplified method based on Gaussian spheres, total number of the points = 10058, (d) The simplified method based on octree coding, total number of the points = 2648, (e) The k-means clustering simplification method [10], total number of the points = 19271, (f) The uniform simplification method [16], total number of the points = 4032, (g) The conformal geometric algebra method [20], total number of the points = 7194.

simplification method [10] have few errors. In addition, based on Table. 1, the simplification rate of the method based on Gaussian spheres is basically as the same as the simplification rate of the DFPSA, and the simplification rate of the DFPSA is 1.5 times that of the k-means clustering simplification method [10]. Therefore, considering simplification rates and simplification effects synthetically, the reconstruction result of the DFPSA is more excellent than other five methods.

The type of horse data set is the same as that of bunny data set, they are both the data sets with a few features. In combination with Table. 1, Fig. 9, and Fig. 10, it can be seen that the simplification rates of the simplified method based on octree coding and the uniform simplification method [16] are high, but the fitting effects of legs are not good. The simplification rate of the k-means clustering simplification method [10] is the worst and the fitting effect of back is not good either, as shown in Fig. 10e. The DFPSA, the simplified method based on Gaussian spheres, and the conformal geometric algebra method [20] have reasonable simplification rates and good fitting effects. Moreover, in Table. 1, it can be seen that the DFPSA simplifies more points than the simplified method

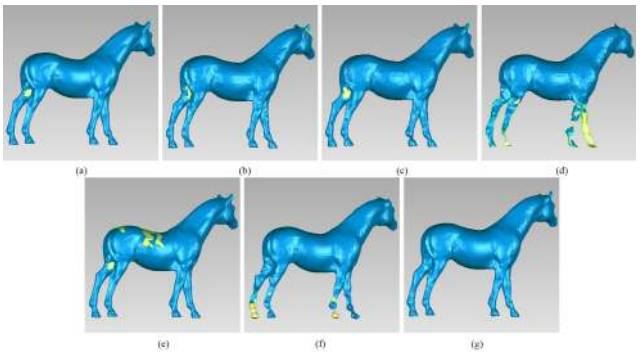

FIGURE 10. Reconstruction results of horse. (a) Original data, (b) Our algorithm, (c) The simplified method based on Gaussian spheres, (d) The simplified method based on octree coding, (e) The k-means clustering simplification method [10], (f) The uniform simplification method [16], (g) The conformal geometric algebra method [20].

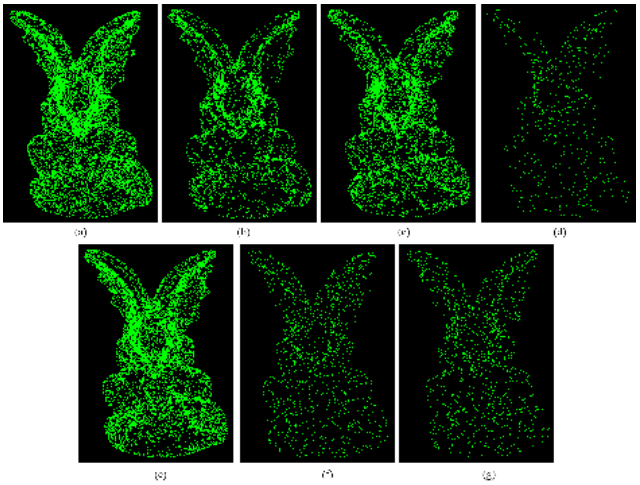

FIGURE 11. Simplified results of gargo50K. (a) Original data, total number of the points = 25038, (b) Our algorithm, total number of the points = 12604, (c) The simplified method based on Gaussian spheres, total number of the points = 16138, (d) The simplified method based on octree coding, total number of the points = 2409, (e) The k-means clustering simplification method [10], total number of the points = 23146, (f) The uniform simplification method [16], total number of the points = 3905, (g) The conformal geometric algebra method [20], total number of the points = 3861.

based on Gaussian spheres, and the conformal geometric algebra method [20] simplifies more points than the DFPSA. However, it is clearly that the difference of simplification rates about these methods is very small. In Fig. 9b, we can see that the DFPSA retains more points at the local detail feature position, retains less points at the smooth position, which not only guarantees the characteristics of the simplified model, but also ensures the simplification rate. In summary, the DFPSA has good simplification effect at both smooth and feature positions.

Fig. 11 shows the simplified results of using simplified algorithms on gargo50K data set. Based on Table. 1, it can be seen that the simplified method based on octree coding simplifies much points, and its simplification rate is as high as 90.4%. The simplification rate of the uniform simplification method [16] is 84.4%. The simplification rate of the conformal geometric algebra method [20] is 84.6%. The simplification rate of the k-means clustering simplification method [10] is only 7.6%. Whereas the DFPSA and the simplified method based on Gaussian spheres retain many points, and their simplification rates are respectively 49.7% and 35.5%. Actually, there is a very important premise of the point cloud simplification, that is the simplified points must be able to reflect the model described by the original point cloud. The results of the simplified method based on octree coding, the uniform simplification method [16] and the conformal geometric algebra method [20] have a high simplify level, but it is obvious that for the data set with lots of detail feature points, the simplified results are often bad. In Fig. 11d, Fig. 11f and Fig. 11g, only the outline of point cloud can be distinguished and it is impossible to identify what object is described by the simplified point cloud model. The result of the k-means clustering simplification method [10] completely reflects the original model, but the simplification rate is too low, which does not meet the criteria of the high degree of simplification. Compared with Fig. 11a, the number of the points in Fig. 11e has hardly decreased. In Fig. 11b and Fig. 11c, not only the outline of point cloud is saved, but also the features are retained. Especially in Fig. 11b, the DFPSA greatly reduces the points at the smooth area and retains the details completely.

The reconstruction results of gargo50K are shown in Fig. 12. In Fig. 12d, Fig. 12f and Fig. 12g, the reconstruction results are very bad, only the outline can be distinguished. The surface of each model is blurred and there are large holes at the wing part of each model. On the contrary, the detail features are obvious in Fig. 12b, Fig. 12c, and Fig. 12e. Each fitting model is almost the same as the fitting model using the original point cloud. In Fig. 12c, the result of the method based on Gaussian spheres has one big empty hole in the wing part. In Fig. 12c and Fig. 12e, the bottom of the model fits incomplete. Whereas in Fig. 12b, the surface is smooth and there are few mistakes, the reconstruction result is almost the same as Fig. 12a. In addition, the DFPSA has a smaller number of points compared with the method based on Gaussian spheres and the uniform simplification method [16]. Hence, in combination with simplification rates and simplification effects, the DFPSA is better than other methods.

Elephant data set has a large number of complex detail features at ear, nose, and ivory areas. Fig. 13 shows the simplification results of elephant data set. In Combination with Table. 1, the simplified method based on octree coding and the uniform simplification method [16] highly simplify point cloud. However, in Fig. 13d and Fig. 13f, it is difficult to distinguish ears and ivory. The k-means clustering simplification method [10], the DFPSA, the simplified method based on Gaussian spheres, and the conformal geometric algebra method [20] retain many points at detail feature position, but a few points at smooth position. In Fig. 13b, Fig. 13c, Fig. 13e, and Fig. 13g, elephant model is described very clearly. In Fig. 13b, we can see that back part has a few points and legs, ears, and ivory parts have many points. This also

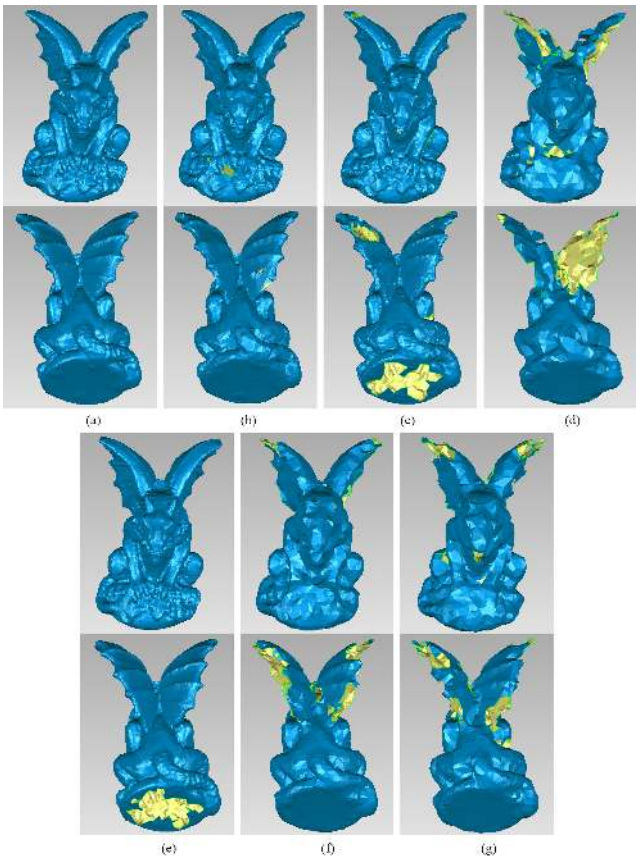

FIGURE 12. Reconstruction results of gargo50K. (a) Original data, (b) Our algorithm, (c) The simplified method based on Gaussian spheres, (d) The simplified method based on octree coding, (e) The k-means clustering simplification method [10], (f) The uniform simplification method [16], (g) The conformal geometric algebra method [20].

reflects the core idea of our algorithm: on the premise of preserving features, simplify as many non-feature points as possible to achieve a high reduction rate.

The reconstruction results of elephant are shown in Fig. 14. Fig. 14d and Fig. 14f have many errors, the elephant's nose and ivory can no longer be distinguished. Fig. 14c, Fig. 14e, and Fig. 14g have some holes at ears and body parts. Although Fig. 14b has a small hole in ivory part, it does not have much effect on the whole model. And compared with the other five algorithms, the reconstruction effect of the DFPSA is the best. In conclusion, the DFPSA has advantages in simplifying point cloud with multiple features.

We have experimented with these six methods in the same system environment. Each method run 20 times on each data set. The average running time is shown in table. 2. It can be seen in table. 2 that for the four data sets, the uniform simplification method [16] takes very short time, but in combination with the above experimental results, it sacrifices simplification accuracy. Although the simplification speed of the method is fast and the simplification rate of the method is high, the simplified point cloud cannot completely replace the original point cloud. This algorithm is suitable for point clouds with a few feature points, or for situations requiring only simplification speed and simplification rate, but not

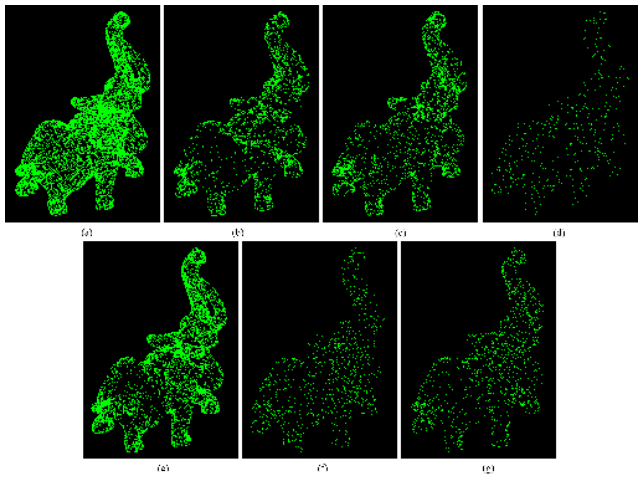

FIGURE 13. Simplified results of elephant. (a) Original data, total number of the points = 24955, (b) Our algorithm, total number of the points = 8483, (c) The simplified method based on Gaussian spheres, total number of the points = 8591, (d) The simplified method based on octree coding, total number of the points = 2496, (e) The k-means clustering simplification method [10], total number of the points = 15833, (f) The uniform simplification method [16], total number of the points = 2851, (g) The conformal geometric algebra method [20], total number of the points = 3887.

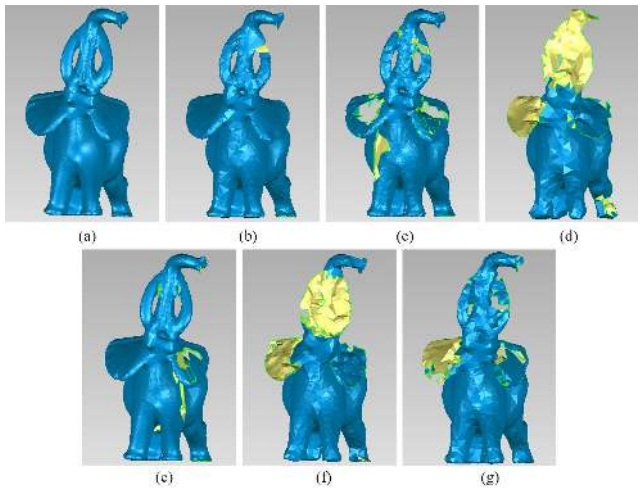

FIGURE 14. Reconstruction results of elephant. (a) Original data, (b) Our algorithm, (c) The simplified method based on Gaussian spheres, (d) The simplified method based on octree coding, (e) The k-means clustering simplification method [10], (f) The uniform simplification method [16], (g) The conformal geometric algebra method [20].

simplification accuracy. The k-means clustering simplification method [10] also has a high simplification speed, but in combination with the above experimental results and table. 1, we can find that the simplification rate of this method is low, which will have a great impact on the operation of point cloud in the future. The simplification speed of the conformal geometric algebra method [20] is also high. However, the simplification of feature parts and the simplification of non-feature part are similar, the simplification of complex features is not ideal. The simplified method based on octree coding has a high simplification rate but the poor speed. And in combination with the above experiments and analyses, this method is not a good choice. For bunny data set and horse data

TABLE 2. The running time (in second) of simplification algorithms.

Simplification rate / %	Bunny	Horse	Gargo50K	Elephant
The DFPSA	56.159	64.407	22.383	26.220
The simplified method based on Gaussian spheres	34.271	37.432	27.299	29.222
The simplified method based on octree coding	66.749	75.826	44.700	38.430
The k-means clustering simplification method [10]	9.429	14.746	11.691	9.602
The uniform simplification method [16]	6.297	10.174	8.358	6.867
The conformal geometric algebra method [20]	9.742	16.215	13.886	9.076

set with a few feature points, the simplified algorithm based on Gaussian sphere runs fast, the DFPSA runs slow. This is because in the DFPSA, for a data set with a few detail feature points, a large number of non-feature points are processed by octree, which makes the overhead of constructing the octree is large, so the overall operation efficiency is reduced. In contrast, for a data set with many feature points, such as gargo50K data set and elephant data set. The number of non-feature points is small, and the overhead of constructing octree is small. Therefore, we can see in table. 2 that the running time of the DFPSA on gargo50K data set and elephant data set is shorter than that of the simplification method based on Gaussian spheres. Furthermore, it can be aware from table. 1 that the simplification rate of our algorithm is higher than that of the method based on Gaussian spheres. To summarize, the DFPSA runs faster on the data set with much detail feature points than on the data set with a few detail feature points. Accordingly, the DFPSA is suitable for the point cloud with a large number of detail features.

C. SIMPLIFICATION RESULTS AT SIMILAR SCALES

Section V. B is the comparative experiment based on the parameters recommended in the reference papers. In order to further verify the effectiveness of the DFPSA, we adjust the parameters to ensure that the simplification rate of each algorithm is similar, and then compare the simplification results of the DFPSA with those of the other five algorithms.

Fig. 15 shows the simplified results of bunny data set while retaining the similar number of points. Fig. 16 shows the corresponding fitting results. It can be seen in Fig. 15a and Fig. 15b, a few points are retained at the smooth part while many points are retained at the contour position. The simplification results of Fig. 15c, Fig. 15d, Fig. 15e and

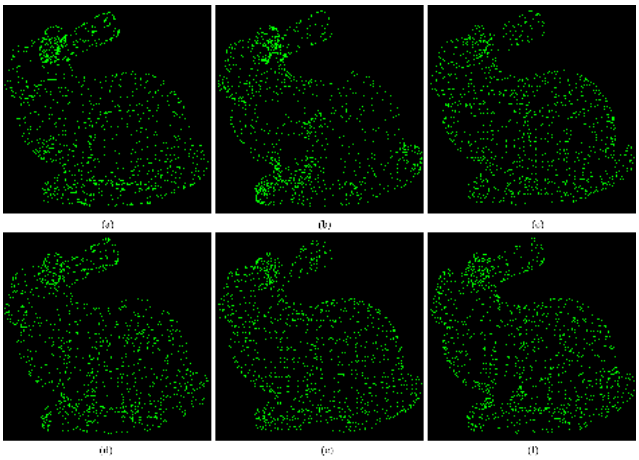

FIGURE 15. Simplified results of bunny. (a) Our algorithm, total number of the points = 4566, (b) The simplified method based on Gaussian spheres, total number of the points = 4391, (c) The simplified method based on octree coding, total number of the points = 4359, (d) The k-means clustering simplification method [10], total number of the points = 4519, (e) The uniform simplification method [16], total number of the points = 4539, (f) The conformal geometric algebra method [20], total number of the points = 4748.

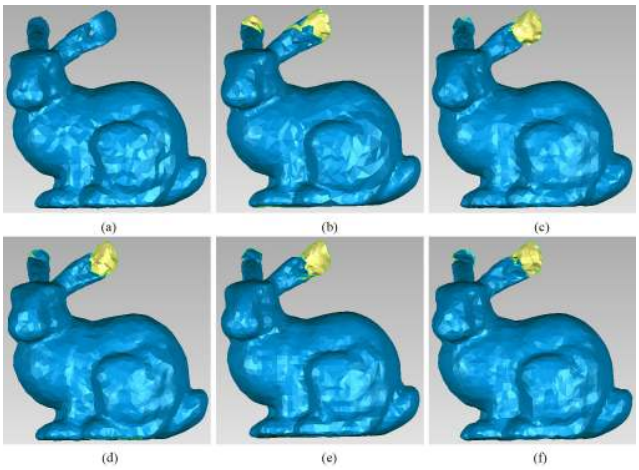

FIGURE 16. Reconstruction results of bunny. (a) Our algorithm, (b) The simplified method based on Gaussian spheres, (c) The simplified method based on octree coding, (d) The k-means clustering simplification method [10], (e) The uniform simplification method [16], (f) The conformal geometric algebra method [20].

Fig. 15f are uniform. It is obviously in Fig. 16 that these six algorithms have good reconstruction results at smooth area, but at the narrow feature location such as ear, the fitting effect of the DFPSA is better than other algorithms.

The simplified results of elephant data set are shown in Fig. 17. In Fig. 17a, Fig. 17b and Fig. 17d, the DFPSA, the simplified method based on Gaussian spheres and the k-means clustering simplification method [10] save a few points at the smooth area and much points at the feature areas, such as legs, ears, nose and ivory. Because the simplification rate of Fig. 17 is high, we can see that in Fig. 18, these six algorithms all have errors in reconstruction process. Nevertheless, the reconstruction result of Fig. 18a is better than those of the other five algorithms, the errors in Fig. 18a are

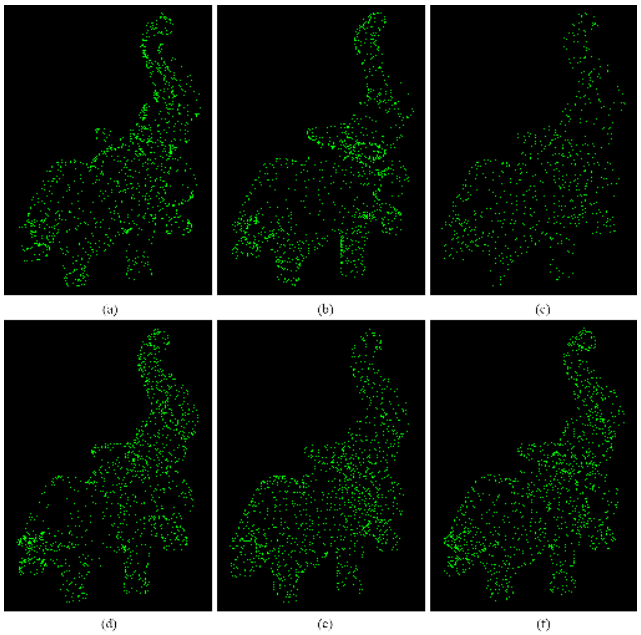

FIGURE 17. Simplified results of elephant. (a) Our algorithm, total number of the points = 2872, (b) The simplified method based on Gaussian spheres, total number of the points = 2640, (c) The simplified method based on octree coding, total number of the points = 2496, (d) The k-means clustering simplification method [10], total number of the points = 2899, (e) The uniform simplification method [16], total number of the points = 2851, (f) The conformal geometric algebra method [20], total number of the points = 2938.

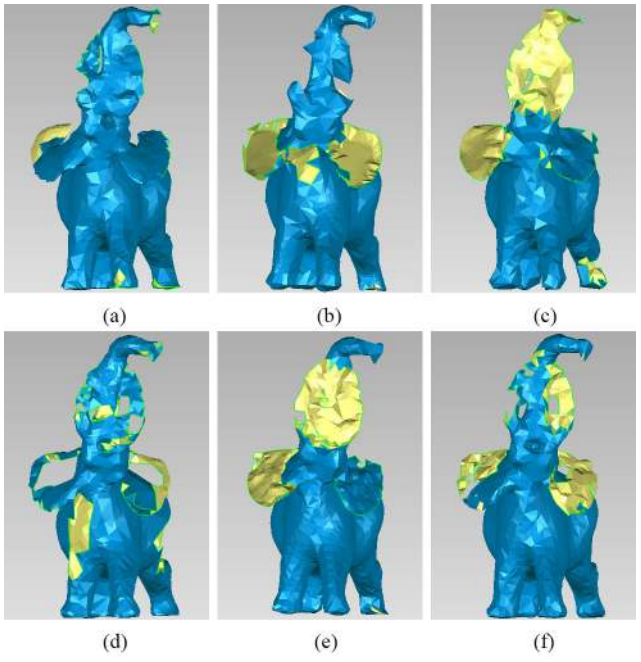

FIGURE 18. Reconstruction results of elephant. (a) Our algorithm, (b) The simplified method based on Gaussian spheres, (c) The simplified method based on octree coding, (d) The k-means clustering simplification method [10], (e) The uniform simplification method [16], (f) The conformal geometric algebra method [20].

obviously less than errors in other figures. This is because the DFPSA can not only greatly simplify non-feature points, but also maintain the features of original point cloud vigorously.

Generally, instead of simplifying the original point cloud directly, the DFPSA considers feature points and non-feature points separately. Thus, the DFPSA can greatly retain the detail feature points. Combining with the simplification rate, the simplification results, the simplification time and the reconstruction results, it can be seen that the DFPSA is able to simplify point cloud and it is more suitable for the simplification of point cloud with complex features.

VI. CONCLUSION

This study presents a new simplification point cloud algorithm based on detailed feature points. First, a new k neighborhood search method based on distance and density is proposed to find the k neighborhood of each point as accurately as possible. Second, feature points are selected based on geometric features of points. Third, feature points and non-feature points are simplified in different ways. The DFPSA greatly simplifies non-feature points while retaining detail feature points.

In order to verify the effectiveness of the DFPSA, we have carried out a series of experiments. Compared with two classical simplified algorithms and three recently proposed simplified algorithms, the DFPSA has good simplification effect and reasonable simplification time. For data sets with a small number of features such as bunny data set and horse data set, when the simplification rate is about 80%, a very good fitting effect can be obtained. For complex point cloud data sets such as gargo50K data set and elephant data set, the DFPSA retains much feature points, and compared with other simplified algorithms, it has the best reconstruction effect at the same scale. According to table. 2, we can see that the simplification speed of the DFPSA for data sets with a large number of complex feature points is shorter than it for data sets with a small number of feature points. It is confirmed that the DFPSA is suitable for the point cloud data set with lots of detail features.

We can learn from the reconstruction experiment in the section V that the DFPSA has a good simplification result at the detail feature positions, the contour positions, and the sharp positions. The algorithm also has good effect for simplifying narrow areas, such as rabbit ears, horse legs, gargoyles wings, elephant nose, and elephant ivory. In addition, according to setting different simplified threshold and setting different simplified parameter proportion, the DFPSA can flexibly simplify different types of point cloud data sets.

REFERENCES

[1] X. Liu and Z. Zhang, "Effects of LiDAR data reduction and breaklines on the accuracy of digital elevation model," *Surv. Rev.*, vol. 43, no. 323, pp. 614–628, Oct. 2011. doi: 10.1179/003962611X13117748892317.

[2] H. C. Jo, J. Kim, K. Lee, H.-G. Sohn, and Y. M. Lim, "Non-contact strain measurement for laterally loaded steel plate using LiDAR point cloud displacement data," *Sens. Actuator A, Phys.*, vol. 283, pp. 362–374, Nov. 2018. doi: 10.1016/j.sna.2018.09.012.

[3] G. Wang, J. Wu, R. He, and S. Yang, "A point cloud-based robust road curb detection and tracking method," *IEEE Access*, vol. 7, pp. 24611–24625, 2019. doi: 10.1109/ACCESS.2019.2898689.

[4] C. Rodríguez-Moreno, J. F. Reinoso-Gordo, E. Rivas-López, A. Gómez-Blanco, F. J. Ariza-López, and I. Ariza-López, "From point cloud to BIM: An integrated workflow for documentation, research and modelling of architectural heritage," *Surv. Rev.*, vol. 50, no. 360, pp. 212–231, 2018. doi: 10.1080/00396265.2016.1259719.
[5] L. Comba, A. Biglia, D. R. Aimonino, and P. Gay, "Unsupervised detection of vineyards by 3D point-cloud UAV photogrammetry for precision agriculture," *Comput. Electron. Agricult.*, vol. 155, pp. 84–95, Dec. 2018. doi: 10.1016/j.compag.2018.10.005.
[6] D. P. Luebke, "A developer's survey of polygonal simplification algorithms," *IEEE Comput. Graphics Appl.*, vol. 21, no. 3, pp. 24–35, May/Jun. 2001. doi: 10.1109/38.920624.
[7] S.-M. Hur, H.-C. Kim, and S.-H. Lee, "STL file generation with data reduction by the delaunay triangulation method in reverse engineering," *Int. J. Adv. Manuf. Technol.*, vol. 19, no. 9, pp. 669–678, May 2002. doi: 10.1007/s001700200112.
[8] F. Sun, Y.-K. Choi, Y. Yu, and W. Wang, "Medial meshes—A compact and accurate representation of medial axis transform," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 3, pp. 1278–1290, Mar. 2016. doi: 10.1109/TVCG.2015.2448080.
[9] B.-Q. Shi, J. Liang, and Q. Liu, "Adaptive simplification of point cloud using k-means clustering," *Comput.-Aided Des.*, vol. 43, no. 8, pp. 910–922, Aug. 2011. doi: 10.1016/j.cad.2011.04.001.
[10] J. F. Chang, L. M. Zhao, and H. B. Wang, "Research on k-means clustering point cloud reduction algorithm based on boundary reservation," *Eng. Surveying Mapping*, vol. 27, no. 7, pp. 60–65, 2018. doi: 10.19349/j.cnki.issn1006-7949.2018.07.013.
[11] H. Song and H.-Y. Feng, "A global clustering approach to point cloud simplification with a specified data reduction ratio," *Comput.-Aided Des.*, vol. 40, no. 3, pp. 281–292, Mar. 2008. doi: 10.1016/j.cad.2007.10.013.
[12] W. Shi, J. Li, Y. Liu, D. Zhu, and D. Yang, "Dynamic obstacles rejection for 3D map simultaneous updating," *IEEE Access*, vol. 6, pp. 37715–37724, 2018. doi: 10.1109/ACCESS.2018.2836192.
[13] P. Goswami, F. Erol, R. Mukhi, R. Pajarola, and E. Gobbetti, "An efficient multi-resolution framework for high quality interactive rendering of massive point clouds using multi-way kd-trees," *Vis. Comput.*, vol. 29, no. 1, pp. 69–83, Jan. 2013. doi: 10.1007/s00371-012-0675-2.
[14] Z. W. Shao and P. Xi, "Data reduction for point cloud using octree coding," *J. Eng. Graph.*, vol. 31, no. 4, pp. 73–76, Apr. 2010.
[15] S. W. Song, J. Liu, and C. Q. Yin, "Data reduction for point cloud using octree coding," in *Proc. ICIC*, Liverpool, England, 2017, pp. 376–383.
[16] L. Renzhong, Y. Man, L. Yangyang, and Z. Huanhuan, "An uniform simplification algorithm for scattered point cloud," *Acta Optica Sinica*, vol. 37, no. 7, Jul. 2017, Art. no. 0710002.
[17] W. Xuan, X. Hua, X. Chen, J. Zou, and X. He, "A new progressive simplification method for point cloud using local entropy of normal angle," *J. Indian Soc. Remote Sens.*, vol. 46, no. 4, pp. 581–589, Apr. 2018. doi: 10.1007/s12524-017-0730-6.
[18] Y. Zang, B. Yang, F. Liang, and X. Xiao, "Novel adaptive laser scanning method for point clouds of free-form objects," *Sensors*, vol. 18, no. 7, Jul. 2018. doi: 10.3390/s18072239.
[19] H. Han, X. Han, F. Sun, and C. Huang, "Point cloud simplification with preserved edge based on normal vector," *Optik*, vol. 126, no. 19, pp. 2157–2162, Oct. 2015. doi: 10.1016/j.ijleo.2015.05.092.
[20] S. Yuan, S. Zhu, D.-S. Li, W. Luo, Z.-Y. Yu, and L.-W. Yuan, "Feature preserving multiresolution subdivision and simplification of point clouds: A conformal geometric algebra approach," *Math. Meth. Appl. Sci.*, vol. 41, no. 11, pp. 4074–4087, Jul. 2018. doi: 10.1002/mma.4616.
[21] N. Wei, K. Y. Gao, R. Ji, and P. Chen, "Surface saliency detection based on curvature co-occurrence histograms," *IEEE Access*, vol. 6, pp. 54536–54541, 2018. doi: 10.1109/ACCESS.2018.2872168.
[22] A. Abzal, M. Saadatseresht, and M. Varshosaz, "Development of a novel simplification mask for multi-shot optical scanners," *ISPRS J. Photogramm. Remote*, vol. 142, pp. 12–20, Aug. 2018. doi: 10.1016/j.isprsjprs.2018.05.010.
[23] M. Connor and P. Kumar, "Fast construction of k-nearest neighbor graphs for point clouds," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 4, pp. 599–608, Jul./Aug. 2010. doi: 10.1109/TVCG.2010.9.
[24] B. Xiong, M. He, and H. Yu, "Algorithm for finding k-nearest neighbors of scattered points in three dimensions," *J. Comput. Aided Des. Comput. Graph.*, vol. 16, no. 7, pp. 909–912 and917, Jul. 2004.
[25] J. H. Friedman, F. Baskett, and L. J. Shustek, "An algorithm for finding nearest neighbors," *IEEE Trans. Comput.*, vol. C-24, no. 10, pp. 1000–1006, Oct. 1975. doi: 10.1109/T-C.1975.224110.
[26] M. Pauly, M. Gross, and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," in *Proc. Conf. Vis.*, Nov. 2002, pp. 163–170.
[27] W. M. Huang, X. W. Peng, P. ZH Wen, and X. J. Wu, "Simplification of scattered point cloud with geometric feature reservation," *Comput. Eng. Appl.*, vol. 45, no. 28, pp. 70–168, 186, Oct. 2009. doi: 10.3778/j.issn.1002-8331.2009.28.050.
[28] D. OuYang and H.-Y. Feng, "On the normal vector estimation for point cloud data from smooth surfaces," *Comput.-Aided Des.*, vol. 37, no. 10, pp. 1071–1079, Sep. 2005. doi: 10.1016/j.cad.2004.11.005.
[29] B. Li, R. Schnabel, R. Klein, Z. Q. Cheng, G. Dang, and S. Y. Jin, "Robust normal estimation for point clouds with sharp features," *Comput. Graph.*, vol. 34, no. 2, pp. 94–106, Apr. 2010. doi: 10.1016/j.cag.2010.01.004.
[30] H. Yu, R. Wang, J. Chen, L. Liu, and W. G. Wan, "Saliency computation and simplification of point cloud data," in *Proc. ICCSNT*, Changchun, China, Dec. 2012, pp. 1350–1353.
[31] X. X. Yao, J. Guo, J. Hu, and Q. X. Cao, "Using deep learning in semantic classification for point cloud data," *IEEE Access*, vol. 7, pp. 37121–37130, 2019. doi: 10.1109/ACCESS.2019.2905546.

CHUNYANG JI received the B.S. degree in software institute from Jilin University, Changchun, China, in 2017, where she is currently pursuing the master's degree. Her research interests include computer vision and pattern recognition, image processing, and point cloud simplification.

YING LI received the B.S., M.S., and Ph.D. degrees from Jilin University. From 2000 to 2006, she was a Teacher with Jilin University, where she has been a Professor with the College of Computer Science and Technology, since 2006. She is currently the Director of the Computer Spatial Information Processing Technology Laboratory and an Academic Committee Member and a Degree Committee Member with the College of Computer Science and Technology, Jilin University. She has published over 40 articles in journals and international conferences. Her research interests include artificial intelligence, machine learning, computer vision and pattern recognition, image processing, computational intelligence, and big data. She is also a Fellow of the China Computer Federation.

JIAHAO FAN received the B.S. degree in computer science and technology and the master's degree from Jilin University, Changchun, China, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His research interests include swarm intelligence algorithm, machine learning, artificial intelligence, computer vision and pattern recognition, and image processing.

SHUMEI LAN began her career as a Teacher, since 1986. She has been a Vice Professor with the College of Computer Science and Technology, Jilin University, since 2004, where she has been engaged in teaching and research, since 2000. She has published nearly 20 articles in domestic core journals. Her major research interests include image processing, machine learning, and artificial intelligence.