

RESEARCH ARTICLE

A New Simplification Algorithm for Point Cloud Based on the Vertical Plane Constraint and Moving Window

WEIHUA LI^{1,2}, LIANGLIN LIU², AND CHUNHUI PENG²¹School of Surveying Mapping and Geographic Information, Tongji University, Shanghai 200092, China²School of Architectural Engineering, Jinggangshan University, Ji'an, Jiangxi 343000, China

Corresponding author: Weihua Li (lwh2311@tongji.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 41771482, and in part by the Natural Science Research Project of Jinggangshan University under Grant JZ2002.


ABSTRACT Using three-dimensional spatial information, this method constructs the detection condition, compares the conditions to be detected, formed by the points in the file to be simplified, then determines the redundancy of the points. Moving windows are used to promote the operation of the algorithm and generate many tiny approximate vertical planes, from which the simplified points will be generated. By selecting experimental data on rabbit and horse and comparing the methods based on mesh and curvature, the proposed method increased the simplification rate of rabbit and horse from 3.022% and 11.123% (mesh method), and 5.704% and 15.316% (curvature method), to 83.387% and 84.296%, respectively, while the standard deviation was reduced from 0.01051 and 0.0157 (mesh method), and 0.0817 and 0.0013 (curvature method), to 0.02179 and 0.01507, respectively. For the simplification of multiple objects, the proposed method increased the simplification rate from 89.113% and 91.826% (mesh method), and 84.79% and 88.91% (curvature method), to 93.458% and 96.916%, respectively, reducing the time by approximately 20 s.

INDEX TERMS Vertical plane constraint, point cloud simplification, normal vector angle entropy, moving window, feature points.

I. INTRODUCTION

Point cloud datasets with high precision and density contain rich geographic information; however, a large amount of redundant information exists, indicating that subsequent processing will consume more time and hardware costs [1]. Among the existing point cloud simplification algorithms, some improve simplification accuracy at the expense of simplification speed, whereas others improve simplification speed at the expense of simplification accuracy. Consequently, it is difficult to find a balance between simplification accuracy and speed [2], [3]. The principal simplification algorithms are currently based on the grid, curvature, and cluster. Grid simplification methods [4], [5], [6], [7], [8], [9] differ in terms of the definition of whether the point belongs to the

grid or the removal methods of redundant points, and many of these research methods are only improved in local aspects. Another type of algorithm for point cloud simplification is based on curvature. Multiple studies have investigated this method [10], [11], [12], [13], [14], the difference between them mainly having to do with the calculation method of curvature, the setting of k-order neighborhoods, and the algorithm of removing redundant points. Abdul Rahman El Sayed et al. [15] proposed a simplified algorithm based on weighted graph representation, which first used the significant characteristics of each shape vertex to identify the geometric region and identify the feature points in the region. Other methods have been proposed [16], [17], [18], [19] based on feature point retention to simplify the point cloud, in which the feature point retention optimization has been limited in some aspects. Numerous works [20], [21], [22], [23], [24], [25], [26], [27] have also used the clustering method to simplify

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

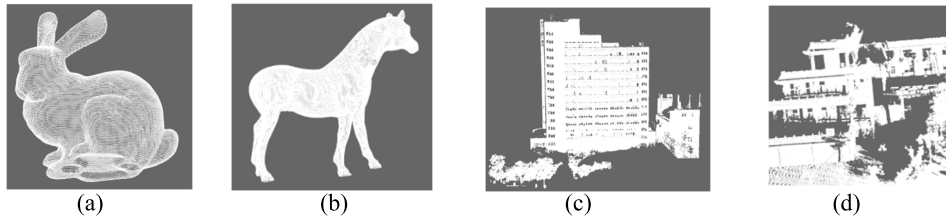


FIGURE 1. The experimental data. (a) Rabbit, (b) Horse, (c) Whu-TLS Campus, and (d) Whu-TLS park.

point clouds from different aspects, which was found to have an effect on simplification, but the number of calculations remained large. Furthermore, additional studies [28], [29], [30], [31] have evaluated the clustering method to simplify the point cloud, but their judgment criteria of clustering has differed, in that the criteria are greatly affected by the neighborhood setting.

Most point cloud simplification algorithms are based on points in the k -neighborhood to calculate the features of the sample points. During the operation, each point repeatedly participates in multiple operations, and the number of operations is always several times the number of point clouds. When performing the k -neighborhood correlation operation, there is a large gap in accuracy if the size of k is selected differently. Accordingly, this study proposes a simplified algorithm that can set the judging requirement from the original spatial distribution and geometric characteristics of the point cloud. In this method, each point is not calculated repeatedly; only the seed points are replaced to judge each point gradually, and a simplified dataset can then be obtained.

II. METHOD

A. EXPERIMENTAL DATA, CONCEPT DEFINITIONS, AND ALGORITHM FRAMEWORK

1) EXPERIMENTAL DATA

To verify the simplification effect and speed of this algorithm and other methods, Stanford rabbit and horse point cloud imagery files were selected as the experimental data. There was a total of 35947 points in the rabbit file and 48485 points in the horse file; the original graphics are shown in Figure 1 (a) and Figure 1 (b). In order to verify that the algorithm's ability to transition from one object to another in the simplifying process, additional files from the public terrestrial laser scanner (TLS) dataset of Wuhan University, Whu-TLS Campus and Whu-TLS Park, were selected as experimental data. The original figures are shown in Figure 1 (c) and Figure 1 (d); the files had 284629 and 368118 points, respectively.

2) CONCEPT DEFINITIONS

Here, suppose P is a point cloud dataset, P_{text} is a point cloud dataset with the name "text", $P_{\text{text}}(i)$ is a point in the dataset P_{text} , and $P_{\text{text}}(i)_{-x}$, $P_{\text{text}}(i)_{-y}$, $P_{\text{text}}(i)_{-z}$ are the x , y , z coordinates of point $P_{\text{text}}(i)$.

Several terms are also defined:

(1) Seed point. The seed point is used to construct the detection condition. One seed point corresponds to one detection condition and another seed point corresponds to

one vertical plane. At the beginning, the first point of the original data file to be simplified is taken as the seed point. The subsequent seed points are derived from the seed point according to the algorithm and stored in the seed point dataset. In the future, the seed point will be taken out from this dataset in sequence.

(2) Next point. The next point is related to finding the serial number in the copied file of original data file to be simplified through the coordinates of the seed point. In the copied file of original data file to be simplified, the next serial of the serial number is the next point.

(3) Detection condition. The detection condition is composed of the seed point and its next point in the serial number of the copied file of the data file to be simplified (because the detected redundant points will be deleted every time in the original data file to be simplified, the next point in the order of the original file may be deleted). The specific composition method is as explained in the first part of Section 2.1, and a seed point corresponds to one detection condition.

(4) Vertical plane. The vertical plane passes through the connecting line between the seed point and the next point in the serial number of the copied file of the data file to be simplified, and is perpendicular to the xoy plane; a seed point corresponds to one vertical plane.

(5) Condition to be detected. The condition to be detected is composed of the seed point and each remaining point of the original data file to be simplified (because the detected redundant points will be deleted every time from the original data file to be simplified). The specific composition method is shown in the first part of Section 2.1. Each remaining point in the current original data file to be simplified and the seed point constitute the conditions to be detected; therefore, one seed point corresponds to multiple conditions to be detected. When a seed point changes, the conditions to be detected of all points in the current original file to be simplified will be changed.

Several point cloud datasets are also defined. P_{original} , the initial dataset which is input and will be simplified; $P_{\text{original-copy}}$ is the copy dataset of P_{original} ; P_{detect} is the dataset of points extracted by each seed point according to the algorithm, because P_{detect} will be cleared after the algorithm about each seed point is running, the points detected by each seed point will be stored circularly in P_{detect} ; P_{seed} is used to store the seed points obtained from P_{detect} ; P_{simply} is used to store the simplified points obtained from P_{detect} , P_{remain} is used to store the remaining points after the whole algorithm ends, and P_{simply1} is used to store the final simplified point dataset.

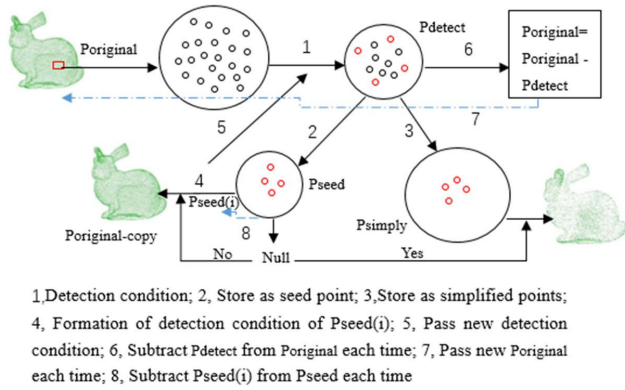


FIGURE 2. The framework of algorithm.

3) ALGORITHM FLOW

Firstly, the first point of the original data file to be simplified is used as the seed point, the detection condition is constructed in $P_{original-copy}$, the conditions to be detected of each point are constructed in $P_{original}$, and the qualified points are stored in P_{detect} . The new seed points and simplified points are extracted in Steps 2 and 3 and stored in P_{seed} and P_{simply} respectively. In Step 6, the points in P_{detect} shall be deleted from $P_{original}$ and make P_{detect} empty. Then take down a seed point from P_{seed} according to the sequence number, construct the detection condition in $P_{original-copy}$, construct the conditions to be detected for each point in the $P_{original}$, which has deleted the points in P_{detect} , and repeat the above operations. Before taking out next seed point, delete the current seed point from P_{seed} . In this loop, if P_{seed} is empty, the operation ends, and the simplified data set P_{simply} is obtained, and then make P_{remain} added to P_{simply} to obtain the final result $P_{simply1}$.

B. ABTAINMENT OF QUALIFIED POINTS WITH ONE SEED POINT RUNNING

1) FORMING THE DETECTION CONDITION AND CONDITIONS TO BE DETECTED

This is about the situation of one seed point. Because the sequence number of the seed point $P_{seed}(i)$ in the seed point dataset is different from that in the copy file of the original file, firstly the sequence number of the seed point should be searched in the copy file of the original file, assuming it as j . Then the corresponding point of $P_{seed}(i)$ in the copy file in the original file is $P_{original-copy}(j)$, and the next point of the seed point in the copied file of the original file is $P_{original-copy}(j+1)$. Figure 3 shows the structure of the detection condition and conditions to be detected of a certain seed point $P_{seed}(i)$. The vertical plane is passing through the line between points $P_{seed}(i)$ and $P_{original-copy}(j+1)$ and is perpendicular to the xoy plane. The projection of point $P_{seed}(i)$ on the xoy plane is $P_{seed}(i)'$; the next point of $P_{seed}(i)$ in the copy file of the original file is $P_{original-copy}(j+1)$, whose projection on the xoy plane is $P_{original-copy}(j+1)'$; the other points in the original file $P_{original}$ are $P_{original}(n)$, where, $n = 1, 2, 3, \dots, N$, N is the number of points in $P_{original}$; and the

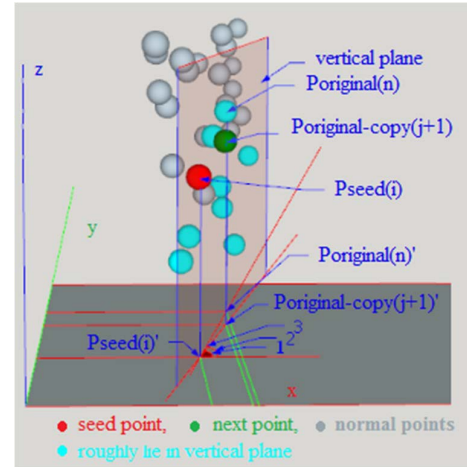


FIGURE 3. Schematic of plane feature constraint.

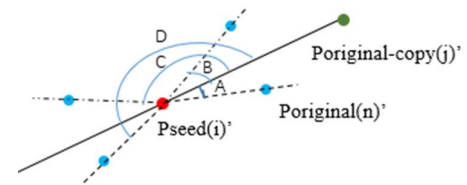


FIGURE 4. Relative position relationship of $P_{original}(n)$ and $P_{seed}(i)$.

projection on the xoy plane is $P_{original}(n)'$, then the detection condition of the seed point $P_{seed}(i)$ will be established.

The angle between the connecting line of $P_{seed}(i)'$ and $P_{original-copy}(j+1)'$ and the x-axis is $\angle 1$, the angle between the connecting line of $P_{seed}(i)'$ and $P_{original}(n)'$ and the x-axis is $\angle 2$, the angle between the connecting line of $P_{seed}(i)'$ and $P_{original-copy}(j+1)'$, and the connecting line of $P_{seed}(i)'$ and $P_{original}(n)'$ is $\angle 3$. Additionally,

$$\angle 3 = \angle 1 - \angle 2 \quad (1)$$

If the point $P_{original}(n)$ crosses the x-axis and the connecting line of $P_{seed}(i)'$ and $P_{original}(n)'$, in a counterclockwise direction, then $\angle 3$ is less than 0. If the point $P_{original}(n)$ crosses the x-axis and moves counterclockwise, and stays between the x-axis and the connecting line of $P_{seed}(i)'$ and $P_{original}(n)'$, then $\angle 3$ is greater than 0. Except for this relationship that the point $P_{original}(n)$ is on the vertical plane, there are four positional relationships between the point $P_{original}(n)$ and the vertical plane, as shown in Figure 4.

Here, angle A, angle B, angle C and angle D are the relative angles between the connecting line of $P_{seed}(i)'$ and $P_{original}(n)'$ and the vertical plane when the relative position relationship of $P_{original}(n)$ and $P_{seed}(i)$ is different.

Assuming that the distance between the point $P_{original}(n)$ and the vertical plane is the same in all four cases, and suppose that the connecting line of $P_{seed}(i)'$ and $P_{original-copy}(j+1)'$ is parallel to the x-axis and judged by the quadrant angle, there is,

$$\angle A = -\angle B, \angle C = \pi - \angle B, \angle D = \pi + \angle B \quad (2)$$

therefore,

$$-\tan(\angle A) = \tan(\angle B) = -\tan(\angle C) = \tan(\angle D) \quad (3)$$

and,

$$|\tan(\angle A)| = |\tan(\angle B)| = |\tan(\angle C)| = |\tan(\angle D)| \quad (4)$$

Therefore, no matter what kind of positional relationship, the degree of distance between the point $P_{\text{original}}(n)$ and the vertical plane can be measured through only using $|\tan(\angle A)|$. It should be noted that $|\tan(\angle A)|$ is not easy to obtain; however, $|\tan(\angle 1)|$, $|\tan(\angle 2)|$ in Figure 3 are easy to obtain. If $|\tan(\angle 1)| = |\tan(\angle 2)|$, the point is on the vertical plane; however, the greater the difference between the $|\tan(\angle 1)|$, $|\tan(\angle 2)|$, the greater the distance between the point and the vertical plane, and the farther the point is from the vertical plane. On the contrary, the closer the point is to the vertical plane. Because $\tan(\angle 3) \neq \tan(\angle 1) - \tan(\angle 2)$, it is necessary to define another quantity to measure the distance between the point and the vertical plane. If this quantity is ΔD , then

$$\Delta D = ||\tan(\angle 1)| - |\tan(\angle 2)|| \quad (5)$$

If ΔD is larger, it means that the distance between the point $P_{\text{original}}(n)$ and the vertical plane is also larger. If ΔD is smaller, it means that the distance between the point and the vertical plane is also smaller. If ΔD is zero, the point $P_{\text{original}}(n)$ is on the vertical plane. Therefore, the points within the threshold range will be selected by setting the threshold value of ΔD .

Therefore, the detection condition $\tan(\angle 1)$ is:

$$\tan(\angle 1) = \frac{P_{\text{original-copy}}(j+1)_{-y} - P_{\text{seed}(i)_{-y}}}{P_{\text{original-copy}}(j+1)_{-x} - P_{\text{seed}(i)_{-x}}} \quad (6)$$

In special cases, if $P_{\text{original-copy}}(j+1)_{-x} - P_{\text{seed}(i)_{-x}} = 0$, this indicates that the vertical plane is perpendicular to the x-axis, and the denominator is meaningless in that case, thus it will be treated as taking the point $P_{\text{original-copy}}(j+2)$ from the copy file of the original file to participate in the calculation. Further, it will be treated this way again if the situation continues to occur.

The conditions to be detected for $\tan(\angle 2)$ are:

$$\tan(\angle 2) = \frac{P_{\text{original}(n)_{-y}} - P_{\text{seed}(i)_{-y}}}{P_{\text{original}(n)_{-x}} - P_{\text{seed}(i)_{-x}}} \quad (7)$$

At this time, assuming that the current number of points in P_{original} is N , then $n = 1, 2, 3 \dots N$, the seed points are unchanged, and each point in current P_{original} will form the condition to be detected with the seed points $P_{\text{seed}(i)}$.

In special cases, if $P_{\text{original}(n)_{-x}} - P_{\text{seed}(i)_{-x}} = 0$, it means that the connecting line of the point $P_{\text{original}(n)'}$, $P_{\text{seed}(i)'}$ is perpendicular to the x-axis and the denominator is meaningless in that case, thus it is treated as storing the point $P_{\text{original}(n)}$ in P_{seed} and P_{simply} and continuing to judge the next condition to be tested. Further, it will be treated this way again if the situation continues to occur.

2) JUDGEMENT

Because the object may have multiple ups and downs, if a distance constraint is not added, the vertical plane may detect distant points that should be retained owing to the features. Therefore, when the seed point $P_{\text{seed}(i)}$ is taken out of P_{seed} , the distance condition should be added. If the distance between point $P_{\text{seed}(i)}$ and $P_{\text{original}}(n)$ is $d(P_{\text{seed}(i)}, P_{\text{original}}(n))$ then:

$$d(P_{\text{seed}(i)}, P_{\text{original}}(n)) = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2} \quad (8)$$

where,

$$\Delta x = P_{\text{original}(n)_{-x}} - P_{\text{seed}(i)_{-x}},$$

$$\Delta y = P_{\text{original}(n)_{-y}} - P_{\text{seed}(i)_{-y}},$$

$$\Delta z = P_{\text{original}(n)_{-z}} - P_{\text{seed}(i)_{-z}}.$$

At this time, assuming that the current number of points in P_{original} is N , then $n = 1, 2, 3 \dots N$, and the seed points are unchanged. Further, each point in the current P_{original} will calculate the distance between points $P_{\text{seed}(i)}$ and $P_{\text{original}}(n)$.

Two thresholds must be set, ΔD and $d(P_{\text{seed}(i)}, P_{\text{original}}(n))$. The threshold of ΔD is defined as the weight of distance between the detected point and the vertical plane. The smaller the value, the more constrained the point is near the vertical plane; while the closer the extracted point is within the vertical plane, the higher the accuracy. If the value is large, the extracted points may fluctuate, and the extraction accuracy will be low if they are not close to the vertical plane. The $d(P_{\text{seed}(i)}, P_{\text{original}}(n))$ threshold should not be too large, because the larger this value is, the easier it is to introduce error. However, neither should it be too small, because the number of redundant points extracted each time would also be small, thus more search time would be required. The judgment criteria is as follows:

$$\left. \begin{array}{l} \Delta D \leq T \\ d(P_{\text{seed}(i)}, P_{\text{original}}(n)) \leq C_r \end{array} \right\} \quad (9)$$

where, T is the threshold of the value of ΔD , and C_r is the distance threshold between point $P_{\text{original}}(n)$ and point $P_{\text{seed}(i)}$.

If point $P_{\text{original}}(n)$ satisfies Equation 9, it is considered an eligible point with $P_{\text{seed}(i)}$ as the seed point, and these points will be stored in another dataset, P_{detect} .

C. EXTRACTION OF SIMPLIFIED POINTS AND NEW SEED POINTS

This is about the situation of one seed point. All points $P_{\text{original}}(n)$, which are detected by taking $P_{\text{seed}(i)}$ as the seed point and conforming to Equation 9, are stored in P_{detect} , and it is assumed that the point cloud distribution of P_{detect} is as shown in Figure 5. Because the detected eligible points are almost in a vertical plane and limited by distance, this is a point cloud distribution of the approximate vertical plane in a small area. Only the peripheral points are feature points, or perhaps all the points inside are not feature points; however, the points in the middle area must be redundant points.

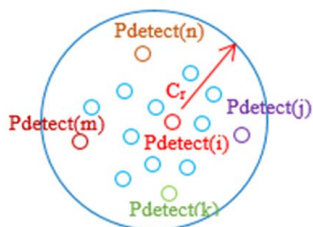


FIGURE 5. Distribution of redundant points when $P_{seed}(i)$ is taken as the detection point.

Therefore, only the points in the four extreme directions in this distribution are retained, that is, the points with the largest and smallest y coordinate and the points with the largest and smallest z coordinate when $|\tan(\angle 1)| \geq 1$ (or the points with the largest and smallest x coordinate and the points with the largest and smallest z coordinate when $|\tan(\angle 1)| < 1$) are retained. As shown in Figure 5, $P_{detect}(j)$ and $P_{detect}(m)$ are extreme points in the y-direction or x-direction, respectively, and $P_{detect}(k)$ and $P_{detect}(n)$ are extreme points in the z-direction, respectively. These four points are selected as simplified points in a small area and stored in the point cloud dataset P_{simply} , as well as in P_{seed} . The resulting dataset P_{simply} is used to store the simplified points, and the dataset P_{seed} is used to store the points that will be used as seed points in the future.

To reduce the number of calculations of the algorithm, the detected point cloud dataset P_{detect} obtained each time in accordance with Equation 9 is deleted from the original point cloud dataset $P_{original}$.

D. MOVING WINDOW PROMOTES THE OPERATION OF THE WHOLE ALGORITHM

This is about the situation of all seed points. The point cloud dataset $P_{original}$, used for simplification, copies an equivalent dataset before the algorithm runs, which is $P_{original-copy}$. After point $P_{seed}(i)$ runs as a seed point, the four points $P_{detect}(j)$, $P_{detect}(k)$, $P_{detect}(m)$, and $P_{detect}(n)$ are generated, which are stored in the simplified point cloud datasets P_{simply} and P_{seed} , used for storing the seed points. The next seed point is taken out in sequence from P_{seed} . Assuming that point $P_{detect}(j)$ is taken, at this time, the point $P_{detect}(j)$ has been stored in P_{seed} , and it is assumed that the corresponding point serial number in P_{seed} is d, that is, the points $P_{detect}(j)$ and $P_{seed}(d)$ are the same point. Then, it is assumed that the corresponding serial number of the point in $P_{original-copy}$ is r, that is, the point $P_{detect}(j)$, $P_{seed}(d)$ and $P_{original-copy}(r)$ are the same point, so the next point of $P_{detect}(j)$ in $P_{original-copy}$ is $P_{original-copy}(r + 1)$.

$P_{detect}(j)$ will find its next point $P_{original-copy}(r + 1)$ in $P_{original-copy}$ to form the detection conditions. The points in the point cloud dataset $P_{original}$ that delete P_{detect} each time, and then the qualified points will be detected as P_{detect} of $P_{seed}(d)$. Then, similar to the operation after $P_{seed}(d)$ detects P_{detect} , points in the four extreme directions are extracted in P_{detect} and stored in P_{simply} and P_{seed} . Because the seed point

constitutes the detection condition with its next point, which is the next point of the seed point in the dataset $P_{original-copy}$, the vertical plane of the next detection condition may be in the same vertical plane or at any angle with the original vertical plane; thus, the next detection vertical plane may operate at any angle, as shown in Figure 6.

To avoid repeatedly removing the points in P_{seed} as detection points, they are deleted from P_{seed} after each seed point operation. Assuming that the $P_{detect}(j)$ operation is completed here, point $P_{detect}(j)$ (i.e. point $P_{seed}(d)$) will be deleted from P_{seed} . If dataset P_{seed} is empty, the run ends. At that time, there may be a few individual points left in the original cloud dataset $P_{original}$, which are likely to be scattered feature points. Therefore, it is necessary to retain them, assuming that these points are P_{remain} , and the final simplification result is $P_{simply1}$, as shown by:

$$P_{simply1} = P_{simply} + P_{remain} \tag{10}$$

E. ADJUSTMENT OF THE SIMPLIFICATION ALGORITHM FOR MULTIPLE OBJECTS

If the point cloud dataset $P_{original}$ contains multiple objects, that is, the point cloud dataset of each object is displayed in blocks and separated by a certain distance, after the points of one object are detected, the seed point cannot be extended to another object. The distance between the seed point $P_{seed}(i)$ and the next point $P_{original-copy}(j + 1)$ (refer to the first few lines of Section 2.2, Part 1) is $d(P_{seed}(i), P_{original-copy}(j + 1))$:

$$d(P_{seed}(i), P_{original-copy}(j + 1)) = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2} \tag{11}$$

where,

$$\begin{aligned} \Delta x &= P_{original-copy}(j + 1)_x - P_{seed}(i)_x, \\ \Delta y &= P_{original-copy}(j + 1)_y - P_{seed}(i)_y, \\ \Delta z &= P_{original-copy}(j + 1)_z - P_{seed}(i)_z. \end{aligned}$$

If $d(P_{seed}(i), P_{original-copy}(j + 1)) > C_r$, $P_{seed}(i)$ will not be taken as a seed point, it will be deleted from P_{seed} , and point $P_{original-copy}(j + 1)$ will be put into $P_{seed}(i)$ to participate in the operation as a seed point. Let the number of such cases be N, where N is the set value. If the number of such cases reaches N, this point $P_{original-copy}(j + 1)$ will jump to the next k point, and k is a settable value. Assuming $k = 100$, the point $P_{original-copy}(j + 1)$ jumps to $P_{original-copy}(j + 101)$, deletes $P_{original-copy}(j + 1)$ from P_{seed} , and places $P_{original-copy}(j + 101)$ into P_{seed} . In this manner, a reasonable seed point can be quickly selected when two objects are in a certain interval.

If $d(P_{seed}(i), P_{original-copy}(j + 1)) \leq C_r$ but P_{detect} is empty, it can also be treated as a jump to quickly replace the seed points, which are generally on the ground or a road surface. Considering the situation, the algorithm can be improved to simplify the points of a single or multiple objects.

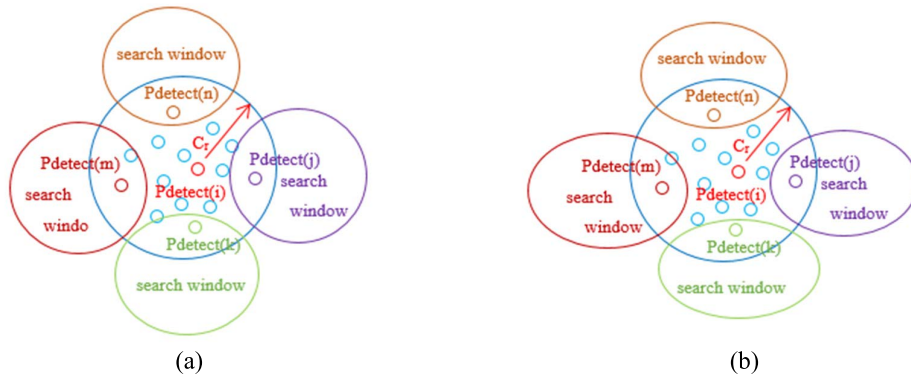


FIGURE 6. Generation of seed points and the search scope: (a) the search of the next seed point occurs approximately on the same plane of the original seed point, and (b) the search of the next seed point may be at any angle.

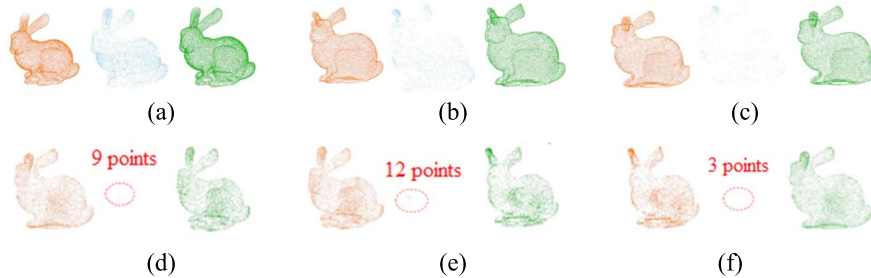


FIGURE 7. The simplification effects when the threshold T and C_r are set to different value. (a) $T = 0.0006$, $C_r = 5$, (b) $T = 0.0008$, $C_r = 6$, (c) $T = 0.001$, $C_r = 7$, (d) $T = 0.006$, $C_r = 8$, (e) $T = 0.007$, $C_r = 9$, (f) $T = 0.01$, $C_r = 10$.

III. EXPERIMENT AND DISCUSSION

A. PARAMETER SETTING

The algorithm in the second section is realized by C++ and PCL, and the experiment of optimal parameters is carried out using rabbit data. For the setting of two thresholds, T (the ΔD threshold) and C_r (the distance threshold), six groups are selected here: (1) $T = 0.0006$, $C_r = 5$, (2) $T = 0.0008$, $C_r = 6$, (3) $T = 0.001$, $C_r = 7$, (4) $T = 0.006$, $C_r = 8$, (5) $T = 0.007$, $C_r = 9$, (6) $T = 0.01$, $C_r = 10$. The algorithm was used to process the rabbit data based on these six groups of parameters, and the relevant data are shown in Table 1.

Figure 5 (a) is figure (a) in figure 5. T and C_r are the values of two thresholds, $P_{\text{original-copy}}$ is the number of points of the original rabbit data, P_{simply} is the number of initially simplified points after the algorithm operates, P_{remain} is the remaining number of points after the algorithm operates, and P_{simply1} is the final number of simplified points. Entropy is calculated according to the formula in literature 10, and time (s) is the running time of the algorithm, ρ is the simplification rate, and the calculation formula is

$$\rho = \frac{P_{\text{original-copy}} - P_{\text{simply1}}}{P_{\text{original-copy}}} \quad (12)$$

The effects of different parameters on the rabbit are depicted below.

Orange (left) is the point cloud of P_{simply} , blue (middle) is the remaining point cloud of P_{remain} , and green (right) is the final point cloud of P_{simply1} , which satisfies the relationship $P_{\text{simply1}} = P_{\text{simply}} + P_{\text{remain}}$. As shown in Figure 7,

as the thresholds T and C_r increased, the number of points in P_{simply1} , P_{simply} , and P_{remain} decreased, but the number of points in P_{remain} occasionally increased. A larger value of T and C_r means that a detection point can search for more qualified points, but only the points with the extreme directions y (or x) and z should be selected. Accordingly, more redundant points are deleted, and there is a greater possibility of deleting the feature points. If more points are detected at a seed point, more redundant points will be removed, the number of point clouds in the subsequent search will be smaller, and the calculation time will be shorter. However, the angle information entropy of the normal vector of the point cloud will decrease faster. The parameters in Figure 7(e) show that the simplification rate reached 83.4%, calculation time was shorter, and information entropy was slightly reduced. Based on the data comparison and quality of the figures, the parameters shown in Figure 7(e) are more appropriate.

B. SIMPLIFIED COMPARISON OF DIFFERENT METHODS

To compare the simplification effect and efficiency of this method with other methods, rabbit and horse data files were selected, and the mesh and curvature simplification methods were chosen. The simplified effect and three-dimensional reconstruction figures are as follows.

It can be seen from the above figures that the method used in this study has a good simplification effect on rabbits and horses. Figure 8(c) shows that the rabbit is evenly simplified after being divided into a grid, and many feature points at

TABLE 1. Different parameters and corresponding data.

Figure 5	T	C _r	P _{original-copy}	P _{simplify}	P _{remain}	P _{simplify1}	Entropy	Time(s)	ρ
(a)	0.0006	5	35947	24308	4422	28730	2.2464	241.705	0.201
(b)	0.0008	6	35947	23555	1476	25031	2.2460	159.542	0.304
(c)	0.001	7	35947	20559	465	21024	2.2378	102.423	0.415
(d)	0.006	8	35947	7299	9	7308	2.2456	28.263	0.797
(e)	0.007	9	35947	5960	12	5972	2.2453	28.781	0.834
(f)	0.01	10	35947	4290	3	4293	2.1784	27.682	0.881



FIGURE 8. The simplification of rabbit. (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

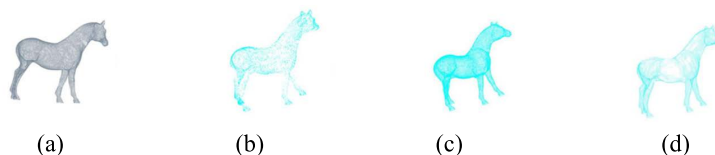


FIGURE 9. The simplification of Horse. (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

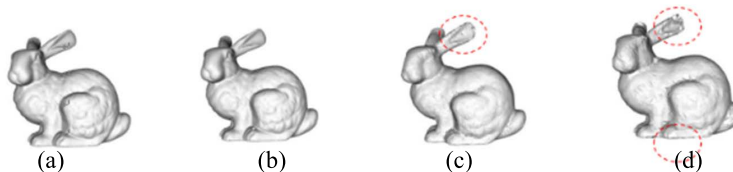


FIGURE 10. The three-dimensional reconstruction of rabbit. (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

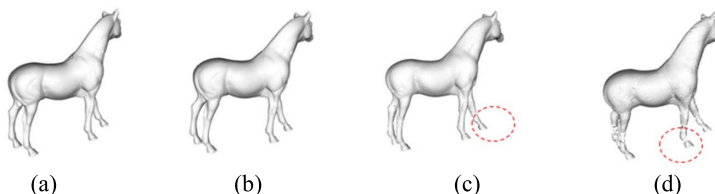


FIGURE 11. The three-dimensional reconstruction of Horse. (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

the corner of the leg are filtered out. In contrast, Figure 8 (b) not only simplifies the point cloud, but also retains the existing feature points, and Figure 8(d) has redundant points. Figure 9(c) also shows a large number of redundant points; in Figure 9 (d), many points in the middle belly of the horse were filtered out. It can also be seen from the three-dimensional reconstruction effect that the effect of the proposed method in this study is closest to the original, while the other two methods have local holes or deformation after reconstruction, as shown by the red circles in Figures 10 and 11.

Using the calculation method of mean value and standard deviation of curvature provided in the literature [17], the mean value and standard deviation of each simplified dataset, in combination with other relevant data, are listed in Table 2.

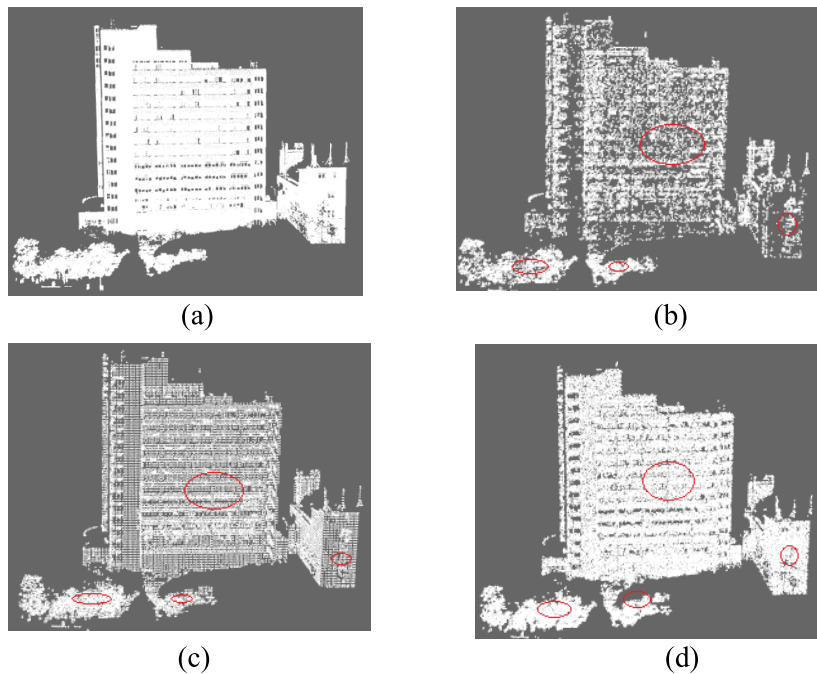
It can be seen from the table that when the simplification time of the rabbit and horse is similar to that of other methods, the simplification rate of this method is the highest, and the mean value of information entropy of the angle between normal vectors is the largest. When the mean curvature of this method is larger or closer to that of the other methods, the standard deviation obtained by this method is less than that of the other methods, which shows that the point cloud distribution after the simplification of this method is more uniform.

C. THE SIMPLIFICATION OF POINT CLOUD OF MULTIPLE OBJECTS

To verify the simplification effect of this algorithm on a point cloud composed of multiple objects, the point cloud

TABLE 2. The simplified data of two kinds of data by different methods.

methods	data	$P_{\text{original-copy}}$	$P_{\text{simplify1}}$	Entropy	Time(s)	$\rho(\%)$	mean value	standard deviation
Proposed method	rabbit	35974	5972	2.2453	28.781	83.387	0.02518	0.02179
	Horse	48485	7614	2.3075	31.803	84.296	0.02099	0.01507
grid method	rabbit	35974	7058	2.0831	29.252	80.365	0.02068	0.0323
	Horse	48485	13007	2.3011	31.329	73.173	0.01275	0.0314
curvature method	rabbit	35974	8022	2.1632	26.004	77.683	0.02299	0.02996
	Horse	48485	15036	2.0233	32.289	68.988	0.01	0.01637

**FIGURE 12.** The simplification of multiple objects (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

of the public dataset, the whu TLS campus, and the whu TLS park were selected as the experimental data. The data size was 8228 KB, with 284629 points, and 10285 KB with 368118 points respectively; the original image is shown in Figure 12(a). The simplified effect of the proposed algorithm on the dataset, which is the whu TLS campus, is shown in Figure 12(b), where the file size was 308 KB, with 18621 points and a 93.458% simplification rate; the red dashed circle represents the objects. Figure 12(b) shows that the method can not only greatly simplify the points of multiple objects but also retain most feature points. However, Figure 12(c) only simplifies multiple objects uniformly without considering the retention of feature points. Therefore, the simplified objects are relatively uniform, but it can be observed from the figure that there are still a large number of redundant points on the objects. In Figure 12(d), there are also a large number of redundant points, and the curvature of the points is calculated

based on the adjacent points. However, when the two objects have a certain distance, the calculated curvature value has a large error, and it can also be seen from the data in Table 3 that the calculation time of the proposed method is less, but the degree of simplification is higher.

The simplified effect of the proposed algorithm on the dataset, which is a whu TLS park, is shown in Figure 13. It can be observed from Figure 13(b) that the simplification effect of the method in this study is the best; the effect in Figure 13(c) is uniformly simplified, and it is difficult to retain the characteristic points of the objects. While the effect in Figure 13(d) depends on the distribution of adjacent points, the error of curvature calculated by adjacent points is large when points cross different objects; therefore, the simplification accuracy is difficult to ensure. It can also be seen from the data in Table 3 that the simplification effect of the proposed method is the best.

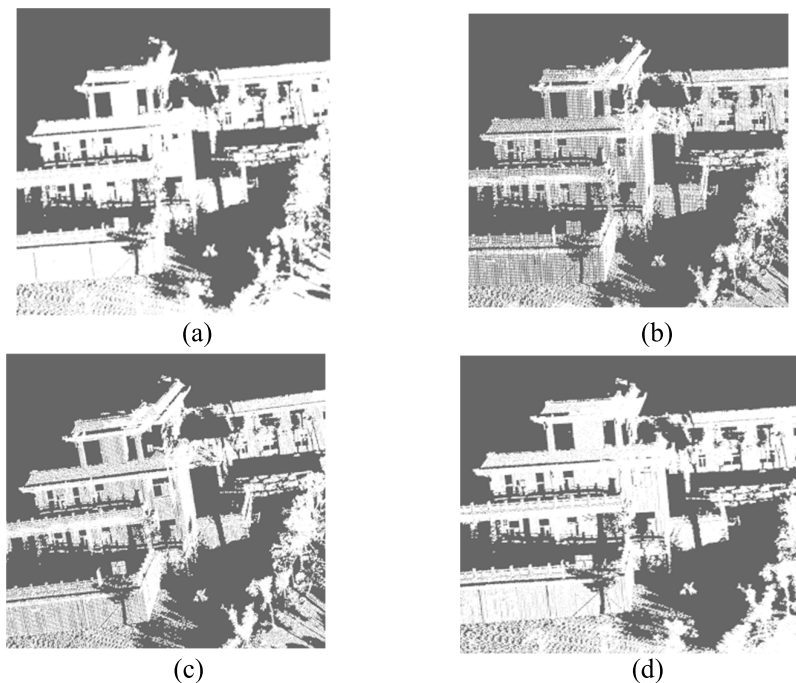


FIGURE 13. The simplification of multiple objects (a) original, (b) the proposed method, (c) grid method, (d) curvature method.

TABLE 3. The simplified data of two kinds of data by different methods.

methods	data	$P_{original-copy}$	$P_{simplify1}$	Time(s)	$\rho(\%)$
Proposed method	whu TLS campus	284629	18621	120.685	93.458
	whu TLS park	368118	11344	128.527	96.916
grid method	whu TLS campus	284629	30988	142.538	89.113
	whu TLS park	368118	30089	153.685	91.826
curvature method	whu TLS campus	284629	43292	154.621	84.79
	whu TLS park	368118	40853	156.581	88.91

IV. CONCLUSION

Experiments show that the method proposed in this paper can run effectively. Through two experimental datasets and comparison with the other two methods, it is proven that the proposed method has a high simplification rate and simplification accuracy, and the time required is generally less than that of the other methods. Using the public datasets of Whu-TLS campus and Whu-TLS park for testing, the proposed method was shown to have a better simplification effect on the point cloud dataset composed of multiple objects. By paying attention to the selection of the T threshold and Cr threshold, all datasets could be simplified with a T threshold of about 0.007, and the Cr threshold was generally 7–9 times of the average spacing of point clouds. The proposed method can be applied not only for a single object, but also for the simplification of a large number of points such as TLS and vehicle LiDAR. Our next study will assume the basis of this algorithm to develop a simplified method that is more suitable for TLS and vehicle LiDAR.

DATA AVAILABILITY STATEMENT

All data, models, or code generated or used during the study are available from the corresponding author by request.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

ACKNOWLEDGMENT

The authors thank Wu for providing them with some animal point cloud data and some guidance.

REFERENCES

- [1] M. Gong, Z. Zhang, and D. Zeng, “A new simplification algorithm for scattered point clouds with feature preservation,” *Symmetry*, vol. 13, no. 3, p. 399, Feb. 2021, doi: 10.3390/sym13030399.
- [2] M. Huang, X. Wu, X. Liu, T. Meng, and P. Zhu, “Integration of constructive solid geometry and boundary representation (CSG-BRep) for 3D modeling of underground cable wells from point clouds,” *Remote Sens.*, vol. 12, no. 9, p. 1452, May 2020, doi: 10.3390/rs12091452.
- [3] C. Ji, Y. Li, J. Fan, and S. Lan, “A novel simplification method for 3D geometric point cloud based on the importance of point,” *IEEE Access*, vol. 7, pp. 129029–129042, 2019, doi: 10.1109/ACCESS.2019.2939684.

- [4] D. P. Luebke, "A developer's survey of polygonal simplification algorithms," *IEEE Comput. Graph. Appl.*, vol. 21, no. 1, pp. 24–35, May/June 2001, doi: [10.1109/38.920624](https://doi.org/10.1109/38.920624).
- [5] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, 2007.
- [6] Y.-J. Liu and M. M.-F. Yuen, "Optimized triangle mesh reconstruction from unstructured points," *Vis. Comput.*, vol. 19, pp. 23–37, Mar. 2003, doi: [10.1007/s00371-002-0162-2](https://doi.org/10.1007/s00371-002-0162-2).
- [7] H. Tang, H. Z. Shu, J. L. Dillenseger, X. D. Bao, and L. M. Luo, "Moment-based metrics for mesh simplification," *Comput. Graph.*, vol. 31, no. 5, pp. 710–718, Oct. 2007, doi: [10.1016/j.cag.2007.05.001](https://doi.org/10.1016/j.cag.2007.05.001).
- [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](https://doi.org/10.1109/TVCG.2015.2448080).
- [9] 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.
- [10] 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](https://doi.org/10.1007/s12524-017-0730-6).
- [11] H. Han, X. Han, F. Sun, and C. Huang, "Point cloud simplification with preserved edge based on normal vector," *Optik, Int. J. Light Electron Opt.*, vol. 126, no. 19, pp. 2157–2162, Oct. 2015.
- [12] K. Zhang, S. Qiao, X. Wang, Y. Yang, and Y. Zhang, "Feature-preserved point cloud simplification based on natural quadric shape models," *Appl. Sci.*, vol. 9, no. 10, p. 2130, May 2019, doi: [10.3390/app9102130](https://doi.org/10.3390/app9102130).
- [13] V. Markovic, Z. Jakovljevic, and Z. Miljkovic, "Feature sensitive three-dimensional point cloud simplification using support vector regression," *Tehnički Vjesnik*, vol. 26, no. 4, pp. 985–994, 2019, doi: [10.17559/TV-20180328175336](https://doi.org/10.17559/TV-20180328175336).
- [14] 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. Methods Appl. Sci.*, vol. 41, no. 11, pp. 4074–4087, Jul. 2018, doi: [10.1002/mma.4616](https://doi.org/10.1002/mma.4616).
- [15] A. R. E. Sayed, A. E. Chakik, H. Alabboud, and A. Yassine, "An efficient simplification method for point cloud based on salient regions detection," *RAIRO, Oper. Res.*, vol. 53, no. 2, pp. 487–504, Apr. 2019, doi: [10.1051/ro/2018082](https://doi.org/10.1051/ro/2018082).
- [16] G. Wang, L. Wu, Y. Hu, and M. Song, "Point cloud simplification algorithm based on the feature of adaptive curvature entropy," *Meas. Sci. Technol.*, vol. 32, no. 6, Jun. 2021, Art. no. 065004, doi: [10.1088/1361-6501/abd497](https://doi.org/10.1088/1361-6501/abd497).
- [17] Y. Yang, M. Li, and X. Ma, "A point cloud simplification method based on modified fuzzy C-means clustering algorithm with feature information reserved," *Math. Problems Eng.*, vol. 2020, pp. 1–13, Oct. 2020, doi: [10.1155/2020/5713137](https://doi.org/10.1155/2020/5713137).
- [18] Q.-N. Zhang, Z.-C. Huang, Z. Xu, and H.-B. Shang, "Study on sampling rule and simplification of LiDAR point cloud based on terrain complexity," *J. Indian Soc. Remote Sens.*, vol. 46, no. 11, pp. 1773–1784, Nov. 2018, doi: [10.1007/s12524-018-0831-x](https://doi.org/10.1007/s12524-018-0831-x).
- [19] S. Liu, J. Liang, M. Ren, J. He, C. Gong, W. Lu, and Z. Miao, "An edge-sensitive simplification method for scanned point clouds," *Meas. Sci. Technol.*, vol. 31, no. 4, Apr. 2020, Art. no. 045203, doi: [10.1088/1361-6501/ab5e00](https://doi.org/10.1088/1361-6501/ab5e00).
- [20] 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](https://doi.org/10.1016/j.cad.2007.10.013).
- [21] 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](https://doi.org/10.1016/j.cad.2011.04.001).
- [22] N. Leal, E. Leal, and S.-T. German, "A linear programming approach for 3D point cloud simplification," *IAENG Int. J. Comput. Sci.*, vol. 44, no. 1, pp. 60–67, 2017.
- [23] Y. Li, H. Liu, Y. Tao, and J. Liao, "Reasoning mechanism: An effective data reduction algorithm for on-line point cloud selective sampling of sculptured surfaces," *Comput.-Aided Des.*, vol. 113, pp. 48–61, Aug. 2019, doi: [10.1016/j.cad.2019.04.002](https://doi.org/10.1016/j.cad.2019.04.002).
- [24] S. Qing, X. Tao, Y. Tatsuo, Z. Yujie, Y. Wenting, and Z. Hang, "Point cloud simplification algorithm based on particle swarm optimization for online measurement of stored bulk grain," *Int. J. Agricult. Biol. Eng.*, vol. 9, no. 1, pp. 71–78, 2016, doi: [10.3965/j.ijabe.20160901.1805](https://doi.org/10.3965/j.ijabe.20160901.1805).
- [25] T. Whelan, L. Ma, E. Bondarev, P. H. N. de With, and J. McDonald, "Incremental and batch planar simplification of dense point cloud maps," *Robot. Auto. Syst.*, vol. 69, pp. 3–14, Jul. 2015, doi: [10.1016/j.robot.2014.08.019](https://doi.org/10.1016/j.robot.2014.08.019).
- [26] J. Wei, M. Xu, and H. Xiu, "A point clouds fast thinning algorithm based on sample point spatial neighborhood," *J. Inf. Process. Syst.*, vol. 16, no. 3, pp. 688–698, Jun. 2020, doi: [10.3745/JIPS.01.0057](https://doi.org/10.3745/JIPS.01.0057).
- [27] Z. Kang, R. Zhong, A. Wu, Z. Shi, and Z. Luo, "An efficient planar feature fitting method using point cloud simplification and threshold-independent BaySAC," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 1842–1846, Dec. 2016.
- [28] W. Cheng, W. Lin, X. Zhang, M. Goesele, and M.-T. Sun, "A data-driven point cloud simplification framework for city-scale image-based localization," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 262–275, Jan. 2017, doi: [10.1109/TIP.2016.2623488](https://doi.org/10.1109/TIP.2016.2623488).
- [29] M. Shoaib, J. Cheong, Y. Kim, and H. Cho, "Fractal bubble algorithm for simplification of 3D point cloud data," *J. Intell. Fuzzy Syst.*, vol. 37, no. 6, pp. 7815–7830, Dec. 2019, doi: [10.3233/JIFS-182742](https://doi.org/10.3233/JIFS-182742).
- [30] P. Chmelar, L. Rejfk, T. N. Nguyen, and D.-H. Ha, "Advanced methods for point cloud processing and simplification," *Appl. Sci.*, vol. 10, no. 10, p. 3340, May 2020, doi: [10.3390/app10103340](https://doi.org/10.3390/app10103340).
- [31] Y. Li, Q. Hu, M. Wu, J. Liu, and X. Wu, "Extraction and simplification of building façade pieces from mobile laser scanner point clouds for 3D street view services," *ISPRS Int. J. Geo-Inf.*, vol. 5, no. 12, p. 231, Dec. 2016, doi: [10.3390/ijgi5120231](https://doi.org/10.3390/ijgi5120231).



WEIHUA LI received the master's and Ph.D. degrees, where he is mainly engaged in the work and research related to the processing of laser radar point cloud. He has a good understanding of the application and development of point cloud in practice, and has some research results. He has presided over and participated in many research projects and published some research results.



LIANGLIN LIU received the master's and Ph.D. degrees, where he is mainly engaged in the work and research related to structural engineering and laser radar point cloud processing. He has a good research foundation and research results for the point cloud research and has solved a certain number of problems in the processing process. He presided over and participated in many research projects and published a certain number of research results.



CHUNHUI PENG received the master's and Ph.D. degrees, where he is mainly engaged in the work and research related to environmental remote sensing and LIDAR point cloud processing. He has a good understanding of point cloud research and has achieved a certain number of research results, solved a certain number of problems, presided over and participated in a number of research projects, and published a certain number of research results.