IOP Publishing | Measurement Science and Technology
Meas. Sci. Technol. **31** (2020) 045203 (11pp) https://doi.org/10.1088/1361-6501/ab5e00

An edge-sensitive simplification method for scanned point clouds

Shifan Liu[1], Jin Liang[1], Maodong Ren[2], JingBin He[1], Chunyuan Gong[1], Wang Lu[1] and Zehua Miao[1]

[1] School of Mechanical Engineering, State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, People's Republic of China
[2] Innovation Lab, XTOP 3D Technology (Shenzhen) Co., Ltd, Shenzhen 518060, People's Republic of China

E-mail: liangjin@mail.xjtu.edu.cn

Received 16 August 2019, revised 20 November 2019
Accepted for publication 2 December 2019
Published 14 January 2020

Abstract

Due to the huge number of points on three-dimensional point clouds captured by optical scanning devices, point-based simplification is a crucial step in model reconstruction. However, the loss of edge features of industrial parts after such simplification reduces reconstruction accuracy. This paper presents an edge-sensitive, point-based simplification method to eliminate redundant points and preserve more edge feature details. Firstly, a new geometrical descriptor is created for each point to generate a geometrical domain. A clustering scheme is then designed by applying two different clustering algorithms, to split the point cloud in the geometrical domain and the spatial domain respectively. The proposed method is capable of preserving edge features well, while reducing the original number of points to 10% or even 5%. The proposed method is compared with other simplification methods and the experimental results indicate that it performs better in simplifying industrial parts.

Keywords: point cloud simplification, edge feature preserving, clustering algorithm

(Some figures may appear in colour only in the online journal)

1. Introduction

In the last decade, three-dimensional scanning technologies have been widely used in model reconstruction [1]. However, the volume of data of raw point clouds is huge, which reduces the efficiency of downstream surface reconstruction. As a result, point cloud simplification is a particularly important step, and it has attracted more and more attention in recent years.

On the other hand, with the development of the automobile manufacturing and equipment manufacturing industries, the accurate surface reconstruction of industrial parts, such as sheet metal parts and cast parts, is becoming a current pursuit [2]. Most of these parts have edge regions or rounded corners for stress equilibrium. These features are important in the analysis of the machining process, and should be preserved during simplification. As a result, the purpose of this paper is to investigate an edge-sensitive, point-based simplification method.

1.1. Previous work

Point cloud simplification is concerned with reducing the number of redundant points and preserving geometric features, so as to provide a better representation of the underlying surface. In early research, many researchers focused on the moving least squares (MLS) method, volume data, and iterative simplification. MLS is used to construct local surfaces implicitly [3, 4], and points are projected to the surface for down sampling. Kobbelt *et al* [5] simplified point clouds by extracting feature-sensitive surfaces based on volume data. Lipman *et al* [6] proposed a locally optimal projection (LOP) operator and applied it to raw scanned data with complex shapes. Huang *et al* [7] developed a weighted locally optimal projection (WLOP) operator based on LOP, which has proven to be less sensitive to noise and has the advantage of producing an evenly distributed point cloud. To reduce the computational complexity of WLOP, Yang *et al* [8] focused on the decomposition of a point cloud and created multiple output results by iterating over each subset. However, LOP and its variations [9, 10] still suffer from high computational costs.

Recently, clustering algorithms have been widely applied to point-based simplification; they split the point cloud into a number of clusters and replace each of them with several points. Wu and Kobbelt [11] introduced a global optimization scheme. In their work, the global minimal set of splats is computed based on normal vectors and spatial extent to replace the entire surface. Zhao *et al* [12] applied a hierarchical clustering method to partitioning the point cloud and selected a representative point for each cluster. Yu *et al* [13] proposed an adaptive simplification method based on a hierarchical tree to simplify point-based models efficiently. The method may lead to the loss of important edge feature information, due to ignoring geometrical characteristics while clustering data points. To improve the efficiency, a *k*-means clustering algorithm [14] was employed to partition point cloud into sphere clusters iteratively, and the selection of initial centroids was exploited. Benhabiles *et al* [15] combined a clustering method and a coarse-to-fine approach to simplify point clouds. Each point of a coarse cloud created by a clustering algorithm is assigned a weight quantifying its importance, and a simplified point cloud is then created by classifying those points into high-curvature or planar regions. Classical clustering methods were tested in a manner insensitive to edges, because they split a point cloud according to the spatial positions of points rather than geometrical characteristics. Many relevant methods have been used to solve the simplification problem. Park *et al* [16] applied mixed-integer quadratic programming techniques to simplification, and they performed well on both indoor and outdoor data points. However, it is not suitable for huge numbers of data points due to its prohibitive computational cost. Han *et al* [17] set an importance for each point based on normal vectors and simplified point clouds by detecting edge features and removing redundant points, which preserves edge features well. Xin *et al* [18] exploited a data simplification method based on the back propagation (BP) neural network, which is affected by the number and the quality of training samples. Bahirat *et al* [19] introduced a curvature-sensitive surface simplification operator, in which points are selected based on the importance of their curvature. Although the algorithm preserves finer details, it is applied only to data points obtained from a depth camera. Chen *et al* [20] proposed a centroidal Voronoi tessellation method to progressively improve the resampling quality by interleaving the optimization of resampling points. However, this algorithm does not have a good performance on point clouds with edge features due to the inaccurately estimated tangent planes. Whelan *et al* [21] presented a method for incrementally growing planar segments and an accurate method to eliminate most of the redundant planar points in dense point clouds. It is time-efficient and suitable for real-time operation, but it is not suitable for the point clouds of industrial parts. Cheng *et al* [22] took the simplification as a weighted *k*-cover problem and simplified the point cloud by using an adaptive exponential weight function based on the visibility probability of 3D points.

Edge areas are crucial features in analysis of the machining process of industrial parts, such as sheet metal parts and cast parts. In addition, some industrial parts are likely to have defects in edge areas, such as burrs or flash. To analyze and improve the machining process, the edge features with both thin and thick contours should be preserved. To preserve more edge features, we create a clustering scheme by applying two different clustering algorithms to the splitting of the point cloud, according to spatial distances and geometrical characteristics respectively, which provides non-sphere clusters to split-edge regions.

1.2. Our work

In this paper we focus on developing a simplification method for scanned point clouds. The method makes the points dense in the areas of high curvature, and sparse in the relatively planar areas. The proposed method has three main parts. First, a feature descriptor of each point is created, to describe geometrical characteristics. The point cloud is then grouped into *k* clusters by a *k*-means clustering method. After obtaining an initial clustering result, the high-curvature clusters are partitioned into some sub-clusters by applying a fuzzy *c*-means (FCM) algorithm and a *k*-means clustering algorithm alternately. The sub-clusters with few points are merged with their neighboring clusters after each iteration. Finally, when the termination condition is satisfied, every cluster will be represented by only one point. There are three main contributions in this paper, as follows.

(1) A new feature descriptor is created to represent the geometrical characteristics of each point, according to the distribution of its neighboring normal vectors. A geometrical domain is then generated, based on the descriptor. As a result, the characteristics of the point cloud can be presented in both spatial and geometrical domains.
(2) A new metric is created in the geometrical domain. An FCM algorithm with the new metric is then proposed, to partition the high-curvature clusters into smaller sub-clusters based on feature descriptors, which is capable of obtaining non-spherical clusters and performs well on the splitting of edge regions.
(3) Two different clustering algorithms are used to group data points, based on Euclidean distance and geometrical characteristics, respectively. The clustering scheme is sensitive to both the edge with a small curvature and the edge with a thick contour, which means that our method is able to preserve small edge details and rounded corner features well.

In section 2 we introduce the principles of the proposed method and the error estimation method. The performance of the proposed method is evaluated by experiment and discussed in section 3. Section 4 concludes the paper.

2. Proposed simplification method

To simplify the point cloud, we create a clustering scheme to partition the point cloud into clusters. Two clustering algorithms are used in our simplification method. The goal of the

k-means clustering algorithm is to gather points whose spatial positions are close, and the goal of the FCM algorithm is to gather points with similar geometrical characteristics. Actually, the k-means clustering algorithm can be regarded as a special FCM algorithm, where its membership function is a signum function and its metric is simple Euclidean distance. Although the k-means clustering algorithm is able to partition a 3D point cloud into k clusters robustly and efficiently, it does not perform well in high-curvature areas. As a result, the FCM algorithm is applied on clustering points in the same cluster generated by the k-means clustering algorithm, based on their geometrical characteristics, due to its insensitivity to noise and changeable metrics.

The point-based simplification algorithm proposed in this paper is performed in several steps. First, if points do not have normal vectors, principal component analysis (PCA) will be used to estimate normal vectors [23]. Second, the geometrical descriptor of each point is calculated according to three eigenvalues of a symmetric matrix built based on unit normal vectors of neighboring points. Third, the point cloud is split into k initial clusters by the k-means clustering algorithm, and the clusters containing high-curvature points are selected according to a selection condition. The FCM clustering algorithm, whose metric is created by combining geometrical descriptors with normal vectors, is then used to partition these clusters further, and the k-means clustering algorithm is used to split the FCM results into small sub-clusters. In order to distinguish the edge-like points from the outliers, clusters with few members named after singular clusters are removed and their members are merged with neighboring clusters after each iteration. Finally, the point cloud is simplified by selecting only one point in each sub-cluster. Figure 1 shows the framework of the whole simplification process. The input point cloud is denoted as a point set $P = \{p_1, p_2, \ldots, p_N\}$ in this paper.

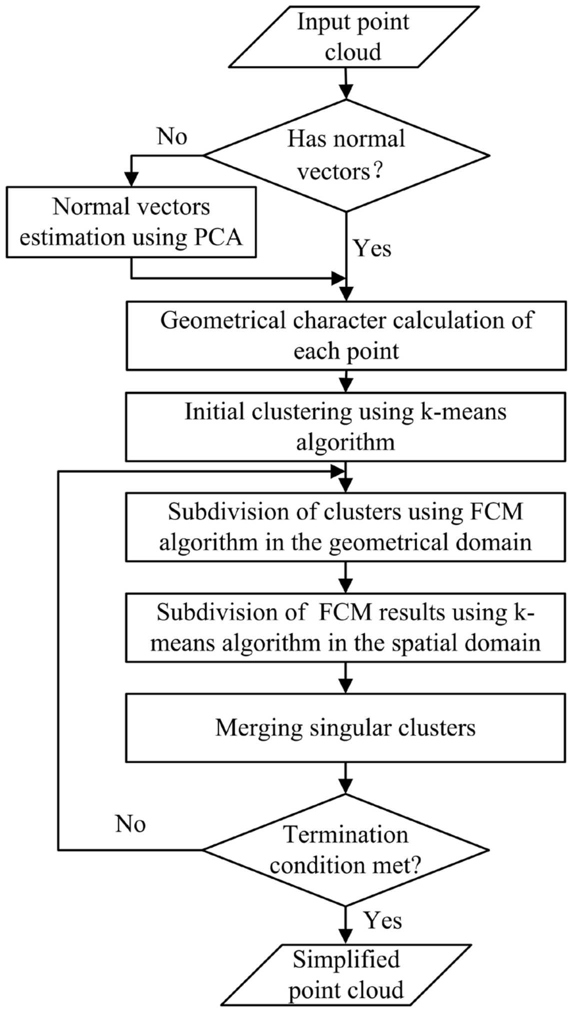

Figure 1. Simplification process.

2.1. Generation of geometrical descriptor

Geometrical descriptors are used to describe geometrical features during the preprocessing of the point clouds. Most geometrical descriptors are computed based on the distribution of the points, ignoring the normal vectors [24]. Therefore, we select the principal curvature as the descriptor. Although the two principal curvature directions can be estimated by PCA after projecting neighboring normal vectors onto the tangent plane [25], it is a complex and time-consuming procedure. To address this problem, we build a 3×3 matrix to describe the two principal curvatures according to the distribution of neighboring normal vectors. We dubbed the mean distance d_{mean} of the point cloud P as a basic length, and we define a search sphere for each point p_i in set P with a constant radius of r ($r = 3d_{\mathrm{mean}}$ in this paper). The point p_j in the search sphere, which meets $\mathbf{n}_i^T\mathbf{n}_j > 0$, is regarded as a neighboring point of p_i, where n_i is the unit normal vector of p_i and n_j is the unit normal vector of p_j. This selection method can eliminate the false neighboring points when the model has two opposing flat parts in close proximity. For the point p_i, the unit normal vectors of all its neighboring points are used to create a matrix, and due to the symmetry of the matrix $n_i^T n_j$, the matrix can be represented as

$$F(\mathbf{n}_n) = \sum_{p_j \in P_i} (\mathbf{n}_n^T \mathbf{n}_j)^2 = \mathbf{n}_n^T (\sum_{p_j \in P_i} \mathbf{n}_j \mathbf{n}_j^T) \mathbf{n}_n, \quad (1)$$

where P_i is the set of neighboring points and n_n is the normal vector of the normal plane. For the point p_i, this matrix $F(n_n)$ is the sum of square distances from normal vectors to the normal plane, and it represents the distribution of neighboring normal vectors. The two unit principal curvature direction vectors are represented as $\mathbf{v}_1 = \arg\min_{\mathbf{n}_n} F(\mathbf{n}_n)$ and $\mathbf{v}_2 = \arg\min_{\mathbf{n}_n} F(\mathbf{n}_n)$. Therefore, the matrix $M = \sum_{p_j \in P_i} \mathbf{n}_j \mathbf{n}_j^T$ represents the distribution of neighboring normal vectors implicitly. Equivalently, the two smaller eigenvalues of matrix M represent two principal curvatures indirectly. The geometrical descriptor $h_i = (\alpha_i, \beta_i)$ of point p$_i$ is computed by

$$\alpha_i = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}, \qquad \beta_i = \frac{\lambda_2}{\lambda_1 + \lambda_2 + \lambda_3}, \quad (2)$$

where λ_1, λ_2, λ_3 ($\lambda_1 > \lambda_2 > \lambda_3$) are the three eigenvalues of the matrix M. Here, α_i represents the non-planar degree of

p_i; in other words, the higher this value is, the more edge-like or corner-like the local region is. Next, β_i represents the edge-like degree of p_i or, equivalently, the higher this value is, the more likely it is that p_i is located at an edge region. As a result, descriptor h_i describes the geometrical feature of p_i by combining α_i and β_i.

The descriptor is created based on normal vectors of neighboring points. Therefore, when the point cloud has too much noise, it is necessary to smooth the normal vectors before creating h_i. The details of the smoothing of normal vectors are described in [9].

2.2. Initial *k*-means clustering method

After the generation of the geometrical descriptor, the point cloud can be represented both in the spatial domain and the geometrical domain. To group the points having similar spatial positions and similar geometrical characteristics, the *k*-means clustering algorithm and FCM algorithm are applied. On the uniform data set, the *k*-means clustering algorithm is more robust and accurate than the FCM algorithm [26]. As a result, the *k*-means clustering algorithm is used to partition the point cloud into k groups, denoted as $G = \{g_1, g_2, \ldots, g_k\}$. Each point is grouped into the cluster with the nearest centroid iteratively.

The choice of initial cluster centroids has a great effect on the *k*-means clustering result. Some of the centroids are likely to gather together when using a common random selection method, resulting in a non-uniform initialization result. In contrast to that method, we select k ($k \ll N$) initial cluster centroids in the following steps.

(1) Select one point in the point cloud randomly.
(2) For the selected point, search its neighboring points and mark them all.
(3) Select the next point from the unmarked points randomly.
(4) Repeat steps 2 and 3 until k points are selected as initial cluster centroids.

Our cluster centroids initialization method is capable of generating uniform cluster centroids.

2.3. Fuzzy *c*-means clustering method

After initial *k*-means clustering, the points in the cluster located at an edge region have different geometrical characteristics, which should be grouped further. Generally, the scanned point cloud is approximately uniform, since all the multi-view scans are from the same scanner. However, unlike with the spatial position, the distribution of characteristics in the geometrical domain is non-uniform, limiting the performance of the *k*-means clustering algorithm. On the other hand, there might be some noise points or outliers in the point cloud due to light disturbance during the data acquisition phase. To address these problems, the FCM algorithm is used due to its insensitivity to non-uniform data and noise [27, 28]. In addition, we create a new metric of the FCM algorithm based on both geometrical characteristics and normal vectors to achieve non-sphere clustering results, capable of partitioning high-curvature clusters into edge and planar areas well.

We denote the geometrical descriptors of points in an initial group g_t ($t = 1, 2, \ldots, k$) as set $H = \{h_1, h_2, \ldots, h_n\}$, where n is the cardinality of the data g_t. The FCM algorithm transforms the classical classification into a fuzzy optimization by introducing a fuzzy weight and membership function. To group the data set, the FCM algorithm defines an objective function

$$J_{\mathrm{FCM}} = \sum_{i=1}^{n} \sum_{k=1}^{c} (u_{ik})^m d(h_i, v_k), \tag{3}$$

where c is the number of fuzzy clusters, u_{ik} is the membership value of the ith member h_i on the kth fuzzy cluster and $\sum_{k=1}^{c} u_{ik} = 1$, v_k is the centroid of the kth fuzzy cluster, $d(h_i,\ v_k)$ is the metric and m is the fuzzification parameter. The FCM algorithm updates the membership values and the cluster centroids iteratively to minimize the objective function. The Euclidean distance between h_i and v_k is the most common metric. However, most rounded corners and edge regions in sheet metal parts or cast parts are symmetrical, and the points on the two sides of an edge region are likely to grouped into the same cluster when using the classical metric because they have similar geometrical descriptors. As a result, we add the unit normal vector into the geometrical descriptor, and it changes into $h_i = (\alpha_i,\ \beta_i,\ n_i)$. A new metric between $h_i = (\alpha_i,\ \beta_i,\ n_i)$ and $v_k = (\alpha_i^v, \beta_i^v, n_i^v)$ is then defined as

$$d(h_i, v_k) = \frac{e^{l(a-\cos b)} + 1}{e^{l(a-\cos b)}} \sqrt{(\alpha_i - \alpha_k^v)^2 + (\beta_i - \beta_k^v)^2}, \tag{4}$$

where $a = \frac{\boldsymbol{n}_i^T \cdot \boldsymbol{n}_k^v}{|\boldsymbol{n}_i| |\boldsymbol{n}_k^v|}$, l is an empirical value which is set to 25 in this paper, and b is a user-defined angle threshold to control the contribution of the normal vector direction to the metric, which is set to 60° in this paper.

Similar to the *k*-means clustering algorithm, the initialization of cluster centroids is vital for the FCM algorithm. To accelerate the clustering process, the selection of c initial cluster centroids consists of three steps, as follows.

(1) Compute the center point of the cluster v_k and select the point whose metric distance to v_k is a maximum as the first cluster centroid.
(2) Select the point whose metric distance to the first cluster centroid is a maximum as the second cluster centroid.
(3) Select c–2 points iteratively, and each selected point v_k meets the condition defined as

$$v_k = \arg\max_{h_i} (\min d(h_i, v_j)), (i = 1, 2, \cdots, n; j = 1, 2, \cdots, k-1). \tag{5}$$

After the initialization of centroids, the membership value of each point graded on every centroid is calculated by

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left[\frac{d(h_i, v_k)}{d(h_i, v_j)}\right]^{1/(m-1)}}, \tag{6}$$

and each point belongs to the group with maximum membership value in the initial clustering. The centroids are updated

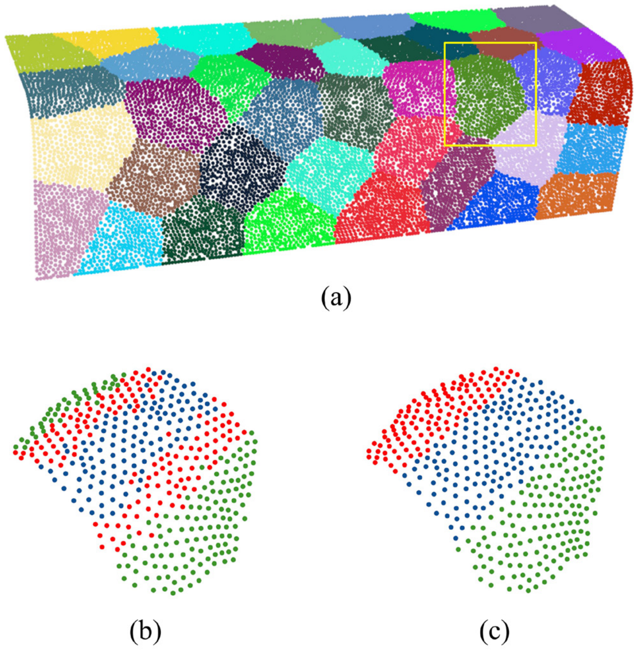

Figure 2. Comparison between the classical metric and the proposed metric: (a) the initial k-means clustering result of a sheet metal part (one cluster is displayed in one color); (b) the FCM clustering result of the initial cluster in the yellow box, using the classical metric ($c = 3$); and (c) the FCM clustering result of the initial cluster in the yellow box, using the proposed metric ($c = 3$, $b = 60°$).

by $v_k = \frac{\sum_{i=1}^{n} u_{ik}^{m} h_i}{\sum_{i=1}^{n} u_{ik}^{m}}$, and the membership values are updated by formula (6) until the movement of each centroid in the spatial domain is lower than a specified minimum threshold, which is set to 10^{-6} in this paper. Figure 2 shows the clustering results achieved by using the classical metric and the proposed metric. We set the number of fuzzy clusters to three, and points in the same cluster are in the same color. Due to the symmetry of the edge area, the points on the two sides have similar geometrical characteristics. Thus, the classical metric cannot partition them well and group them into the same cluster, although they are discontinuous in the spatial domain. In contrast, the proposed metric with angle control can distinguish these points, and makes the sub-clusters continuous in the spatial domain. It can be seen that the FCM algorithm using the proposed metric is capable of splitting the two sides of the edge region.

2.4. Alternate clustering

In order to gather points with similar geometrical and spatial characteristics, FCM clustering and k-means clustering are applied in the geometrical domain and spatial domain alternately. The clustering process of each initial cluster consists of four steps, as follows.

(1) Applying the FCM algorithm to the geometrical domain, to gather points having similar geometrical characteristics.
(2) Applying the k-means algorithm to the spatial domain, to split the FCM results into smaller sub-clusters. In an FCM cluster, the two points whose spatial positions are the farthest are selected as the new initial centroids of the k-means clustering algorithm.
(3) Merging a group with few members with its neighboring groups after each iteration. In order to preserve geometrical features, we should distinguish edge points from outliers. For the whole point cloud, the distribution of real geometrical characteristics is continuous. However, in local regions, both outliers and edge points are likely to cause discontinuity of the distribution of geometrical characteristics, and it is difficult to distinguish them in the present single cluster. Due to this local geometrical discontinuity, these outliers or edge points will be grouped into clusters with few members or even only one member. Thus, we merge the cluster with its neighboring cluster, in three steps. First, the group named after the singular cluster is removed from the clustering result. Then, for each point in the singular cluster, the clusters containing its neighboring points are found as its neighboring clusters. Finally, the membership values of the point graded on its neighboring clusters are calculated, and the point is grouped into the neighboring cluster with maximum membership value. If the point is an edge point, it will be grouped into the neighboring high-curvature region cluster whose members have similar geometrical characteristics to it. If the point is an outlier, it will be grouped into the singular cluster again in the next update because its geometrical characteristic differs from all its neighboring points. If all sub-clusters subdivided from the same cluster are merged into one cluster, these singular clusters and their members will be removed directly. Figure 3 shows the process and the result of merging. After subdivision, the initial three clusters (figure 3(c)) are partitioned into nine clusters (figure (d)). Before merging, the red vertex is in a singular cluster, and after merging, it is merged into its neighboring edge cluster colored in blue. It can be seen that the edge point is merged into its neighboring edge cluster whose members have similar geometrical characteristics to it. Although the merging process is likely to remove sharp points, resulting in sharp feature loss, it is suitable for the point clouds of sheet metal parts and cast parts because most of these industrial parts have few sharp regions, in order to avoid stress concentrations.
(4) We select the sub-clusters with a maximum metric higher than the user-defined threshold ε as the next initial clusters, and repeat steps 1 to 3 until all clusters' maximum metrics are lower than the threshold ε.

The fuzzification of parameter m determines the fuzzy degree of the boundaries between fuzzy clusters. In this paper we set m to 2 and the number of fuzzy clusters c to 3. The cluster whose members number fewer than 3 is regarded as a singular cluster.

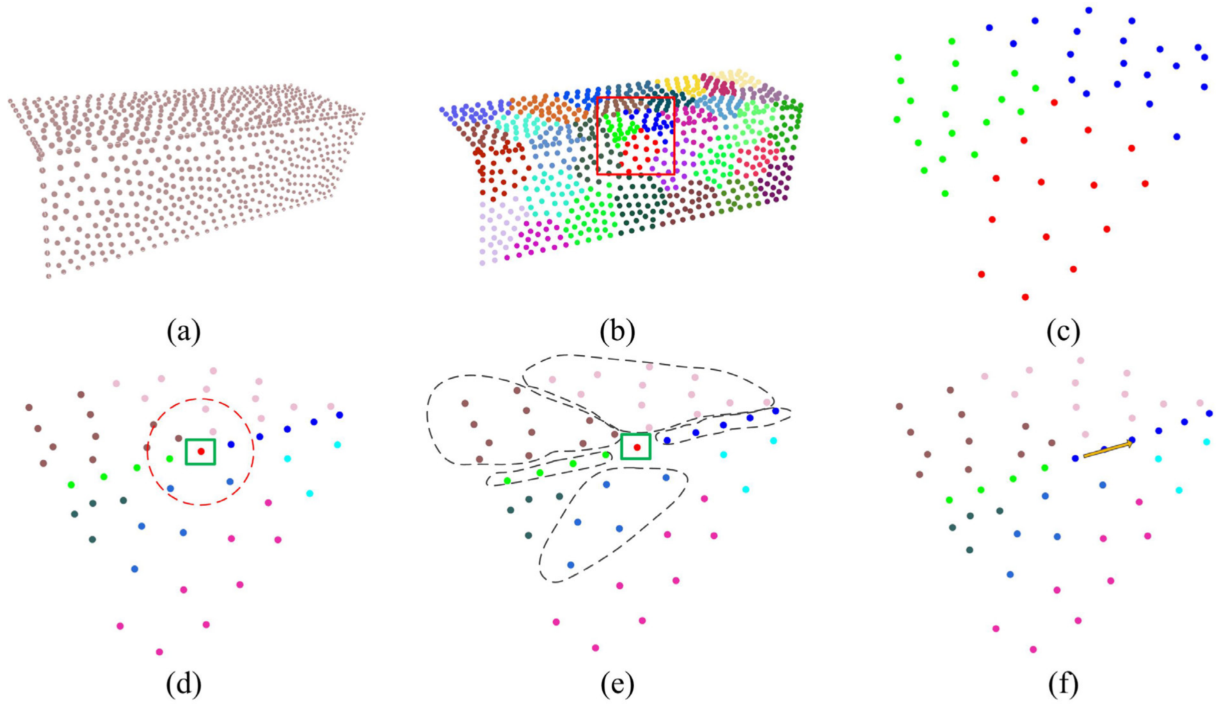

Figure 3. The process of merging a singular cluster into its neighboring clusters: (a) an edge point cloud; (b) initial clusters after initial *k*-means clustering; (c) the three initial clusters in the red box of (b). (d) A singular cluster containing only one edge point in the green box after fuzzy clustering, with its neighboring points in the red circle. (Each initial cluster is partitioned into three sub-clusters. The green, red, and blue clusters are edge clusters.) (e) Neighboring clusters of the singular cluster containing neighboring points. (f) Merging the edge point into the neighboring edge cluster with maximum membership value.

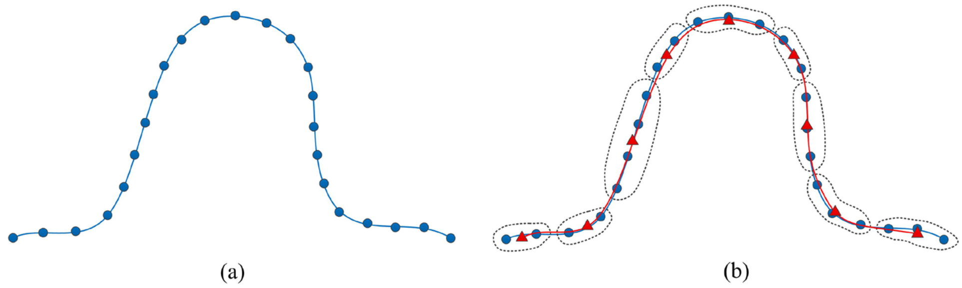

Figure 4. Simplification by replacing each cluster by its centroid: (a) initial points marked in blue circles and origin surface marked in a blue curve; (b) centroids marked in red triangles and simplified surface marked in a red curve.

2.5. Final simplification

After clustering, the point cloud is partitioned into many clusters, and in each cluster, the points have similar spatial and geometrical characteristics. Therefore, we select a representative point for each cluster to simplify the point cloud. In this paper we replace each cluster by its centroid [12, 14]. Figure 4 shows the final simplification process. There is an error between the origin surface and the simplified surface, and the quality of the simplification result can be measured by estimating the error.

If the user wants the simplification result to be a proper subset of the origin point dataset, the point which is nearest to the centroid is selected to represent the cluster, rather than the centroid.

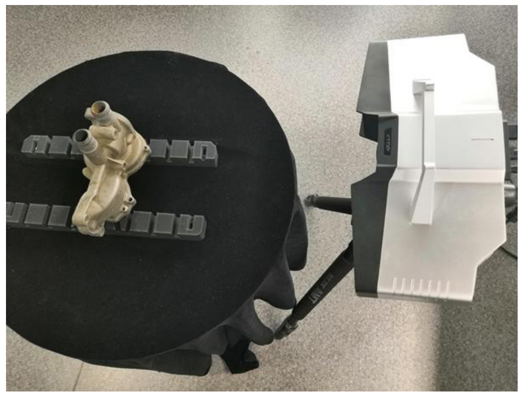

Figure 5. Elements of experiment: XTOP system and the scanned model.

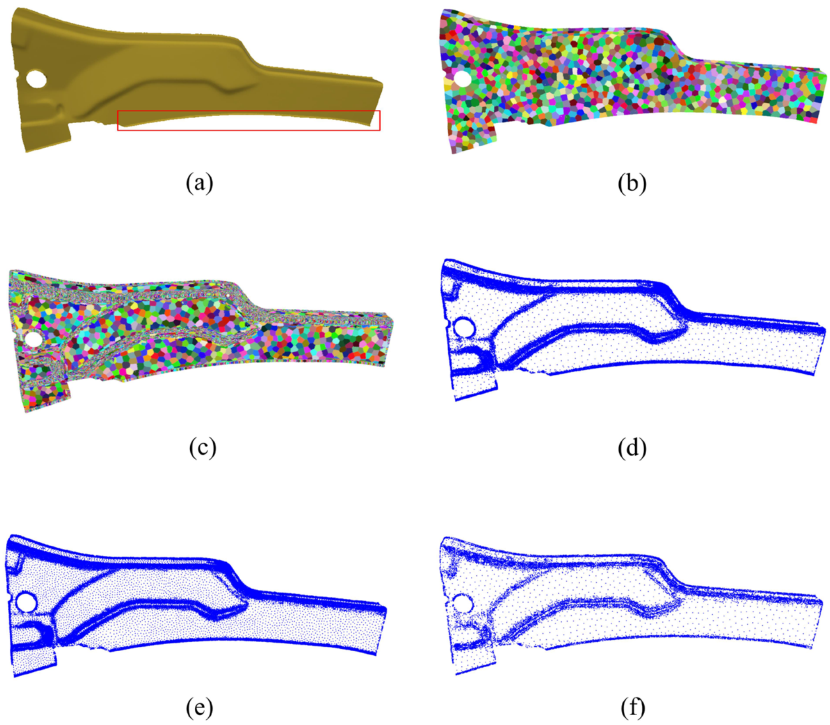

Figure 6. Simplification of a sheet metal part using the proposed method: (a) the input point cloud (361 760 points) (the sheet metal part has noticeable flash and burrs, such as the region in the red box); (b) initial k-means clustering result ($k = 1000$); (c) final clustering result ($\varepsilon = 0.0001$); (d) final simplification result of (c) (30 430 points); (e) final simplification result (39 798 points, $k = 5000$, $\varepsilon = 0.0001$); (f) final simplification result (18 023 points, $k = 1000$, $\varepsilon = 0.0005$).

2.6. Error estimation

To quantify simplification accuracy, we must measure the geometrical error between the simplified point cloud Q and the original point cloud P. We denote the surface of P as S and the surface of Q as S'. Similar to [3, 29, 30], we estimate the simplified error by measuring the maximum error and the mean error between the surface S and the surface S'. In accordance with [3], an up-sampled point cloud of points on S is created to compute point-to-surface distances. Owing to the approximate uniform distribution of the scanned point cloud, we use the original point clouds directly to measure error in this paper. For each p in the origin point cloud P, the maximum error $E_{\max}(S,S')$ and the mean error $E_{\text{mean}}(S,S')$ can be defined as

$$E_{\max}(S,S') = \max_{p\in P} d(p,S'), \tag{7}$$

$$E_{\text{mean}}(S,S') = \frac{1}{N}\sum_{p\in P} d(p,S'), \tag{8}$$

where N is the cardinality of the origin point set, and $d(p,S')$ is the error distance, defined as the distance between point p and its projection point p' on the surface S'. For the mesh surface S', the simplified point cloud is triangulated using the surface reconstruction method of Delaunay triangulation.

3. Experiment and analysis

In this section, we design several experiments to present the simplification result of the proposed method and compare it with other simplification methods. We program the proposed algorithm in Visual Studio 2013 on a desktop computer with 3.6 GHz CPU and 8 GB RAM. In this paper, we use three point clouds as experiment models. The sheet metal part model, cast part model and blade model are acquired using the XTOP system, which is a structured light scanning system, as shown in figure 5.

3.1. Results of our method

Figure 6 is an example of the proposed simplification process. The point cloud is a sheet metal part with edge regions and noticeable burrs and flash. More edge and flash features should be preserved, because these features are crucial information for the analysis of the blanking process. It can be seen that the initial clusters in the edge regions and flash regions are subdivided into a few small sub-clusters, resulting in a denser distribution in the high-curvature areas. Figure 6(d) shows that the final simplification result preserves more points in the edge regions and flash regions. Figure 6(e) illustrates that the number of initial clusters determines the number of points in planar regions. Figure 6(f) shows that the threshold

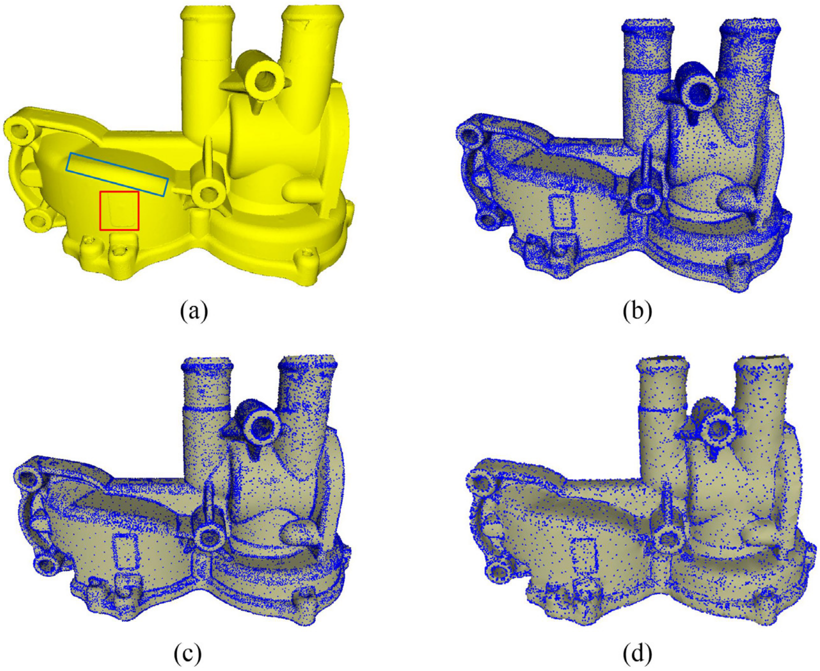

Figure 7. Simplification of a cast part model using the proposed method: (a) the input point cloud (951 301 points) (a rectangular step in the red box and a rounded edge with a thick contour in blue box); (b)–(d) simplification results by reducing the original number to 70 771, 49 518, and 19 473.

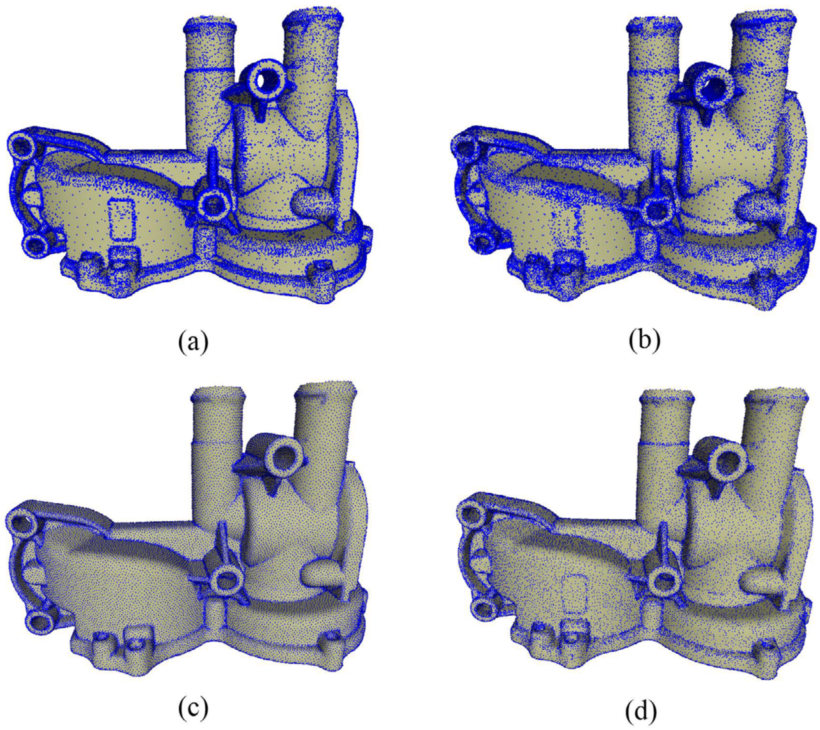

Figure 8. Comparison of four simplification methods on preservation of edge features: (a) the proposed method (70 273 points); (b) Shi *et al* (69 495 points); (c) the HC method (71 386 points); (d) Han *et al* (70 396 points).

ε determines the degree of subdivision. With the increase of ε, fewer points in high-curvature regions are preserved and the details of edge and flash feature are more vague. Meanwhile, the distribution of preserved points is more uniform and balanced. Therefore, the number of points in the final simplified point cloud is determined by the two input parameters.

Figure 7 shows a cast part model with edge regions and rounded corners. The original model was reduced to approximately 7.5%, 5%, and 2% after simplification. There are two edge types in this model. The rectangular step shown in the red box in figure 7(a) is an obscure edge feature, the height of which is less than 1.0 mm, and the rounded corner in the blue

Table 1. Comparison of preserved number and simplification time.

Models	Original number	Proposed method: Preserved number	Proposed method: Running time (s)	Shi *et al*: Preserved number	Shi *et al*: Running time (s)	HC: Preserved number	HC: Running time (s)	Han *et al*: Preserved number	Han *et al*: Running time (s)
Sheet metal part	361 760	30 423	7.146	30 580	6.416	30 426	6.612	30 749	9.460
Cast part	951 301	41 274	18.027	45 455	22.697	42 361	18.583	42 808	30.948
Blade	1175 978	34 933	22.368	34 732	24.737	35 146	20.822	35 279	33.184

Table 2. Comparison of simplification error (unit: mm).

Models	Proposed method: E_{max}	Proposed method: E_{mean}	Shi *et al*: E_{max}	Shi *et al*: E_{mean}	HC: E_{max}	HC: E_{mean}	Han *et al*: E_{max}	Han *et al*: E_{mean}
Sheet metal part	0.0969	0.003 22	0.114	0.003 36	0.173	0.006 94	0.102	0.005 14
Cast part	0.262	0.008 81	0.579	0.0101	0.673	0.0180	0.371	0.0132
Blade	0.101	0.007 85	0.142	0.0121	0.184	0.0154	0.126	0.0146

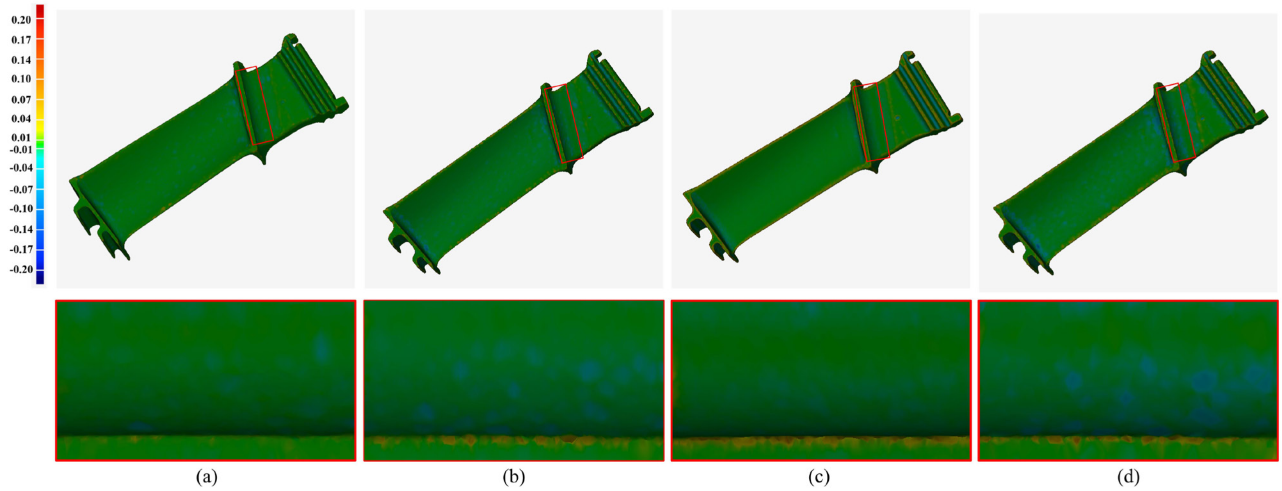

Figure 9. Error estimation results and the edge area enlargement of four simplification methods: (a) the proposed method; (b) the method of Shi *et al*; (c) the HC method; (d) the method of Han *et al*.

box is an edge with a thick contour. Both types of edge feature should be preserved after simplification. Figures 7(b) and (c) show that the edge features and rounded corners are well preserved by applying the proposed method when the simplification ratio is lower than 10%. The two types of edge feature are both recognized and preserved. Figure 7(d) shows that when the simplification ratio is too low, although the density of the edge areas is still higher than the planar areas, some features have been missed. As a result, the original model should not be simplified excessively.

These examples show that the proposed method is capable of simplifying scanned industrial parts, preserving the more crucial feature details while reducing the original number to 5%.

3.2. Comparison with other methods

To illustrate the simplification effect of the edge area, we compare the proposed method with three other simplification methods on edge features. The two edge types shown in figure 7(a) in section 3.1 are crucial features for industrial parts and should be preserved. All four simplification methods reduce the original number to similar simplified numbers. The simplification results are shown in figure 8. The hierarchical clustering (HC) method [12] produces the most uniform and smoothest simplification result; however, both the small edge feature and the rounded corner features are lost. Although the method of Shi *et al* [14] preserves rounded corner features, the small edge feature is vague because this method does not take the local geometrical characteristic into consideration. Conversely, the method of Han *et al* [17] preserves the small edge feature, but cannot recognize the rounded corner because it treats the feature as a planar area due to the thick edge contour. The proposed method retains both the small edge and the edge with a thick contour, so preserving more edge feature information. It can be seen that the proposed method is edge-sensitive for the scanned point cloud.

We compare our algorithm with the other three methods in accuracy and running time. The comparison of the preserved number and computation efficiency is shown in table 1 and the comparison of error estimation is shown in table 2. It can be seen that the proposed algorithm provides a more accurate simplification result when the four algorithms preserve similar simplified numbers. The HC method produces the smoothest results with the highest efficiency, but it gives the largest edge feature loss among the four methods. For the sheet metal model, the proposed algorithm is time-consuming compared with the method proposed by Shi *et al* because it spends time creating a geometrical domain. However, it is computationally efficient when applied to the cast part model and blade model because the proposed algorithm gathers points with both similar geometrical characteristics and similar spatial positions into the same cluster, leading to a smaller iteration number. Taking the cast part model as an example, the maximum iteration number of the proposed algorithm is 2, whereas the maximum iteration number of the algorithm proposed by Shi *et al* is 6. As a result, the new clustering scheme can produce an accurate result efficiently at edge areas, due to its low iteration number. The maximum error produced by the method of Han *et al* is lower than that of Shi *et al* and the HC method, which illustrates that it has an ability to preserve small and sharp edge features due to retaining edge points. However, the mean error is high in the method of Han *et al* because it removes points in rounded corners. In addition, this method is time-consuming. Experimental results show that the proposed algorithm is more accurate when the model has more edge features, and has good time efficiency.

Figure 9 shows the error estimation results intuitively. In each figure, the edge region in the red box is enlarged and shown below the blade model. The HC algorithm produces the smoothest result among the four methods and it produces the best result in relative planar areas. However, it blurs the edge features and produces the worst result among the four methods in edge area regions. The proposed method and the method proposed by Shi *et al* produce similar simplification results in the planar areas, due to their similar *k*-means clustering processes. In the edge areas, the proposed method produces a result with less feature loss, due to partitioning clusters according to geometrical characteristics. The method of Han *et al* preserves more edge features than that of Shi *et al* and the HC method, but performs the worst in rounded corner areas. Among the four methods, the proposed method provides the best simplification result, with mean error and maximum error the lowest. In addition, the distribution of the simplification error is uniform, which indicates that even in the edge regions, the simplification error is also low.

These experimental results demonstrate that the proposed method is edge-sensitive and performs well in simplifying scanned industrial parts.

4. Conclusion

This paper proposes an edge-sensitive, point-based simplification method by designing a clustering scheme using *k*-means clustering and FCM clustering algorithms. The proposed method is capable of splitting edge regions from planar regions and eliminating redundant points effectively.

A geometrical descriptor is created to describe the local geometrical characteristic of each point by building a matrix based on the normal vectors of neighboring points. A new metric of FCM is then created to avoid an inaccurate fuzzy clustering result by introducing normal vector direction restriction, making full use of the descriptor's sensitivity to edge regions. The clustering scheme is designed by extending *k*-means clustering and FCM clustering algorithms to split the point cloud in the spatial domain and local geometrical domain respectively. The FCM clustering algorithm is used to partition clusters based on geometrical feature, and the *k*-means clustering algorithm provides an initial coarse clustering result and splits the FCM results into smaller sub-clusters in the spatial domain. In order to distinguish edge points from outliers, a merging strategy is designed to improve the clustering result after each iteration.

Experimental results using three industrial part models show that our simplification method can be applied to scanned point clouds with edge features, and is especially suitable for industrial parts with small edge regions or big rounded corners, such as sheet metal and cast parts. It is a reliable and robust simplification method with high simplification accuracy.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1105800).

ORCID iDs

Shifan Liu https://orcid.org/0000-0001-7852-1576
Chunyuan Gong https://orcid.org/0000-0001-9466-2280

References

[1] Nan L L and Wonka P 2017 PolyFit: polygonal surface reconstruction from point clouds *IEEE Int. Conf. Computer Vision* vol 1 pp 2353–61
[2] Anwer N and Mathieu L 2016 From reverse engineering to shape engineering in mechanical design *CIRP Ann.– Manuf. Technol.* **65** 165–8
[3] Pauly M, Gross M and Kobbelt L P 2002 Efficient simplification of point-sampled surfaces *IEEE Computer Society Press Conf. Proc.* pp 163–70
[4] Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D and Silva C T 2001 Point set surfaces *IEEE Computer Society Press Conf. Proc.* pp 21–8
[5] Kobbelt L P, Botsch M, Schwanecke U and Seidel H P 2001 Feature sensitive surface extraction from volume data *Computer Graphics and Interactive Techniques – SIGGRAPH Conf. Proc.* pp 57–66
[6] Lipman Y, Cohen-Or D, Levin D and Tal-Ezer H 2007 Parameterization-free projection for geometry reconstruction *ACM Trans. Graphic.* **26** 22
[7] Huang H, Li D, Zhang H, Ascher U and Cohen-Or D 2009 Consolidation of unorganized point clouds for surface reconstruction *ACM Trans. Graphic.* **28** 176
[8] Yang H Y, Hao K R and Ding Y S 2015 Resampling to speed up consolidation of point clouds *Math. Probl. Eng.* **2015** 646410
[9] Huang H, Wu S, Gong M, Cohen-Or D, Ascher U and Zhang H 2013 Edge-aware point set resampling *ACM Trans. Graph.* **32** 1–12

[10] Preiner R, Mattausch O, Arikan M, Pajarola R and Wimmer M 2014 Continuous projection for fast L1 reconstruction *ACM Trans. Graph.* **33** 47
[11] Wu J and Kobbelt L 2004 Optimized sub-sampling of point sets for surface splatting *Comput. Graph. Forum* **23** 643–52
[12] Zhao P C, Wang Y and Hu Q W 2016 A feature preserving algorithm for point cloud simplification based on hierarchical clustering *IEEE Int. Geoscience and Remote Sensing Symp.* pp 5581–4
[13] Yu Z, Wong H S, Peng H and Ma Q 2010 ASM: an adaptive simplification method for 3D point-based models *Comput. Aided Des.* **42** 598–612
[14] Shi B Q, Liang J and Liu Q 2011 Adaptive simplification of point cloud using *k*-means clustering *Comput. Aided Des.* **43** 910–22
[15] Benhabiles H, Aubreton O, Barki H and Tabia H 2013 Fast simplification with sharp feature preserving for 3D point clouds *IEEE 11th Int. Symp. on Programming and Systems* pp 47–52
[16] Park H S, Wang Y, Nurvitadhi E, Hoe J C, Sheikh Y and Chen M 2013 3D point cloud reduction using mixed-integer quadratic programming *IEEE Computer Vision Pattern Recognition Workshops Conf. Proc.* pp 229–36
[17] Han H Y, Han X, Sun F S and Huang C Y 2015 Point cloud simplification with preserved edge based on normal vector *Optik* **126** 2157–62
[18] Xin M, Li B, Yan X, Chen L and Wei X 2018 A robust cloud registration method based on redundant data reduction using back propagation neural network and shift window *Rev. Sci. Instrum.* **89** 024704
[19] Bahirat K, Raghuraman S and Prabhakaran B 2017 Real-time, curvature-sensitive surface simplification using depth images *IEEE Trans. Multimedia* **20** 1489–98
[20] Chen Z, Zhang T, Cao J, Zhang Y J and Wang C 2018 Point cloud resampling using centroidal Voronoi tessellation methods *Comput. Aided Des.* **102** 12–21
[21] Whelan T, Ma L, Bondarev E, With P H N and McDonald J 2015 Incremental and batch planar simplification of dense point cloud maps *Robot. Auton. Syst.* **69** 3–14
[22] Cheng W, Lin W, Zhang X, Goesele M and Sun M T 2016 A data-driven point cloud simplification framework for city-scale image-based localization *IEEE Trans. Image Process.* **26** 262–75
[23] Mitra N J, Nguyen A and Guibas L 2004 Estimating surface normals in noisy point cloud data *Int. J. Comput. Geom. Appl.* **14** 261–76
[24] Bueno M, González-Jorge H, Martínez-Sánchez J and Lorenzo H 2017 Automatic point cloud coarse registration using geometric keypoint descriptors for indoor scenes *Autom. Constr.* **81** 134–48
[25] Liu S F, Liang J, Gong C Y and Pai W Y 2018 Registration method of point clouds using improved digital image correlation coefficient *Opt. Eng.* **57** 113104
[26] Mahdaoui A, Bouazi A, Hsaini A M and Sbai E H 2017 Comparison of *k*-means and fuzzy *c*-means algorithms on simplification of 3D point cloud based on entropy estimation *Proc. Int. Conf. on Applied Mathematics* vol 33 pp 201–26
[27] Wu K L and Yang M S 2002 Alternative *c*-means clustering algorithms *Pattern Recogn.* **35** 2267–78
[28] Cui X, Li S P, Yan X T and He X H 2018 A curvature-based weighted fuzzy *c*-means algorithm for point clouds de-noising *Int. Conf. on Graphic and Image Processing* p 106153
[29] Cignoni P, Rocchini C and Scopigno R 1998 Metro: measuring error on simplified surfaces *Comput. Graph. Forum* **17** 167–74
[30] Miao Y, Pajarola R and Feng J 2009 Curvature-aware adaptive re-sampling for point-sampled geometry *Comput. Aided Des.* **41** 395–403