EPCS: Endpoint-based Part-aware Curve Skeleton Extraction for Low-quality Point Clouds

Chunhui Li  
Northwest University  
China, Xi’an, 710127  
lichunhui2020@stumail.nwu.edu.cn

Mingquan Zhou  
Northwest University  
China, Xi’an, 710127

Zehua Liu  
Northwest University  
China, Xi’an, 710127

Yuhe Zhang  
Northwest University  
China, Xi’an, 710127  
zhangyuhe0601@nwu.edu.cn

November 18, 2022

ABSTRACT

The curve skeleton is an important shape descriptor that has been utilized in various applications in computer graphics, machine vision, and artificial intelligence. In this study, the endpoint-based part-aware curve skeleton (EPCS) extraction method for low-quality point clouds is proposed. The novel random center shift (RCS) method is first proposed for detecting the endpoints on point clouds. The endpoints are used as the initial seed points for dividing each part into layers, and then the skeletal points are obtained by computing the center points of the oriented bounding box (OBB) of the layers. Subsequently, the skeletal points are connected, thus forming the branches. Furthermore, the multi-vector momentum-driven (MVMD) method is also proposed for locating the junction points that connect the branches. Due to the shape differences between different parts on point clouds, the global topology of the skeleton is finally optimized by removing the redundant junction points, re-connecting some branches using the proposed MVMD method, and applying an interpolation method based on the splitting operator. Consequently, a complete and smooth curve skeleton is achieved. The proposed EPCS method is compared with several state-of-the-art methods, and the experimental results verify its robustness, effectiveness, and efficiency. Furthermore, the skeleton extraction and model segmentation results on the point clouds of broken Terracotta also highlight the utility of the proposed method.

Figure 1: The results of the LBC [15], $L_1$-medial [6], MdCS [16] and EPCS methods on the horse model from the Terracotta dataset, which is a raw scan point cloud.
1 Introduction

The skeleton is a typical shape descriptor for 3D models, and concisely and intuitively reveals the geometry, topology, and symmetry structure of the described shape [1]. Skeletons also play a critical role in shape completion [2], shape segmentation [3], [4], and model classification and recognition [5]. The extraction of the skeleton is also of great significance for the restoration of broken cultural relics, especially for symmetry models like vases; furthermore, the skeleton can also be employed to reconstruct the surfaces of cultural relics with missing fragments [2], [6], [7].

The skeletons extracted by existing methods can be roughly classified into three main groups, namely surface skeletons [8], [9], [10], curve skeletons [2], [6], and centerlines [11], [12], [13]. The main difference between these three types of skeletons is that they are suitable for different shapes. Specifically, surface skeletons are 2D shapes representing the whole shape of the original point cloud, and are more suitable for shapes for which the surface structure is more important than the whole topology, e.g., chairs and tables [8], [9]. Centerlines are usually used in the medical and biological fields, and are well suited for describing the structures of blood vessels or intestines [12]. Curve skeletons are the most widely used because they contain highly concise topology information about objects [2], [5], [6], [14], [15], [16]. This study focuses on the extraction of the curve skeleton, for which both the shape of the object and the topological structure between the parts of the object are taken into consideration.

Raw scan point clouds are usually obtained via reconstruction from a series of 2D images [17], [18] or scanning by a 3D scanner [19], [20]. Due to many restrictions of the scanning process, the raw scan points are usually disorganized, lack inherent structural or directional information, and contain noises, and outliers [21]; in addition, due to the limited scanning angle, raw scan point clouds generally contains holes which usually cause data missing. As a result, raw scan point clouds are usually of low-quality, particularly for those without any processing such as noise reduction or hole filling. Therefore, it is necessary for the algorithms to be robust to highly non-uniform sampling patterns, diverse levels of noise, and even missing data [2]. The extraction of skeletons based on a Voronoi diagram (VD) is a typical strategy, but the model must have a polygonal surface [22], [23]. Alternatively, curvature flow-based methods have also achieved satisfactory performance [20], [24], [25]; however, the computation of the curvature and other geometric invariants is time-consuming, and only dense point clouds without missing data and noise can be handled. The third category of extraction methods is realized by some geometric features of the skeleton, such as the rotational symmetry axis (ROSA) [2], the local separators [26], which can be applied to the raw scan point clouds. Several state-of-the-art methods utilize a robust central axis to fit each part of the model to iteratively obtain the curve skeleton, such as ROSA [2], $L_1$-median [6]. These methods usually take a resampled model as the input to reduce the computational cost, such as [6], [16]; however, incomplete skeletons may be obtained. To tackle this problem, in the present study, all parts of the point cloud are first located by detecting the endpoints of the parts, and the skeletal curve of each part is regarded as the branch of the final skeleton. Furthermore, the essence of skeletonization is to appropriately divide the part into layers [16], [27], [28], and the center point of each layer is considered as the skeletal point.

Therefore, the endpoint-based part-aware curve skeleton (EPCS) extraction algorithm is proposed, and Figure 1 presents the result of the EPCS algorithm on a raw scan point cloud. First, the novel random center shift (RCS) method is introduced to detect the endpoints, which are utilized as the seed points for the generation of the skeletal points in the next stage. Subsequently, region growing [29], is used to guide the division of layers, and the center points of the oriented bounding box (OBB) [30] of the layers are then regarded as the skeletal points. Because layer division is a linear sequential process, the skeletal curves can be directly obtained by connecting the skeletal points in sequence. However, the skeletal curves obtained in this stage are independent branches; therefore, the multi-vector momentum-driven (MVMD) method is proposed, which considers both the local distance and direction information and iteratively connects the branches, thus forming the complete skeleton. Because global information about the given point cloud is not considered when connecting the branches, and the distribution of skeletal points is non-uniform and not smooth due to the shape differences between different parts, the global topology of the skeleton is finally optimized by: removing the redundant junction points, re-connecting some branches using the proposed MVMD method, and applying an interpolation method based on the 1-4 splitting operator [31] to smooth the skeletal curves. Consequently, a complete and smooth curve skeleton is obtained.

Different from existing methods that consider the topology of the skeleton after the rough skeleton is obtained, such as [2], [6], [15], [16], the proposed method preliminarily considers the skeletal branches via endpoint recognition, which effectively avoids missing branches. Since region growing can realize the layering of the model directly, and the layering results can be used as the input of skeletal point computation, region growing strategy has been widely used in the skeleton extraction task such as [32] and [33]. Meanwhile, we use a small radius constraint to enhance the region growing strategy, thus avoiding the influence of noises and outliers, e.g., the colored points in Figure 2 (n) are all the points which are visited in the region growing process; whereas the noises and outliers are marked gray far from the model, they are not involved in the calculation. Furthermore, the advantage of using the center point of OBB as the skeletal point is that OBB is insensitive to missing data, this is because even if the OBB is constructed by using
the point set with missing data, the center point of the OBB will not change significantly; moreover, OBB is robust to the noises and outliers [30] due to the computation of the principal directions of the point set when constructing the OBB, thus reducing the impact of noises and outliers on the bounding box constructing. Therefore, the proposed EPCS performs well on the raw scan point cloud with holes, noises, and outliers.

The main contributions of this study are summarized as follows:

- A novel EPCS method is proposed for curve skeleton extraction, which is robust to noises, outliers, and missing data, and it can be applied to low-quality point clouds. The proposed EPCS requires no preprocessing, such as mesh reconstruction, Voronoi graph generation, or the computation of geometric invariants.
- A novel RCS method is proposed for the detection of the endpoints on point clouds, which can effectively avoid missing branches during skeleton extraction.
- The MVMD method is proposed for locating the junction points that connect the branches, which also significantly improves the completeness and smoothness of the skeletal curves in the connected regions.
- The proposed EPCS outperforms current state-of-the-art methods with respect to effectiveness, robustness, and efficiency.

2 Related work

According to Tagliasacchi et al. [1], there are three types of skeletons, namely surface skeletons [8], [9], [10], curve skeletons [2], [6], and centerlines [11], [12], [13], among which curve skeletons are believed to directly promote the restoration of cultural relics. Curve skeletons are 1D, and many ideas for extraction have been developed. In this section, we only review the methods of extracting curve skeleton.

The use of a VD to extract the skeleton is a typical idea. Ogniewicz et al. [22] proposed a skeletonization method based on the VD of boundary points. Subsequently, based on a given graph and a 3D polygonal surface, Lu et al. [23] used centroidal Voronoi tessellation (CVT) to achieve skeleton fitting and conduct model segmentation in a point cloud data set. While both the method proposed by Lu et al. [23] and the proposed EPCS method can obtain the skeleton and conduct model segmentation, the EPCS method does not require a given graph and a 3D polygonal surface.

Alternatively, Au et al. [20] used curvature flow to achieve Laplacian smoothing and complete the geometry contraction process. Subsequently, Chuang et al. [24] published an approach for the tracking of a hierarchical finite element system with a surface evolving under mean curvature flow (MCF), which is characterized by a time advantage. Considering the skeletonization problem via MCF, Tagliasacchi et al. [34] generated mean curvature skeletons. Vahrenkamp et al. [25] also constructed the mean curvature skeleton to identify object regions that are suitable for grasping by robotic hands; the mean curvature skeleton was found to achieve good results and performance on watertight surface meshes. However, these methods can only be used on models that do not contain missing data and that are processed after scanning, and require the input point cloud to be clean and dense enough to generate the correct curve skeleton.

In recent years, the object of numerous skeleton extraction studies has been the tree model [35], [36], [37], [38], [39]. Livny et al. [35] presented an automatic approach that robustly reconstructs the skeletal structures of trees via the active laser scanning of real-world vegetation data. Fu et al. [36] extracted a skeleton from tree point clouds via an octree and the level set method. Wu et al. [37] extracted a skeleton to estimate the phenotypic parameters of maize plants with high accuracy. To optimize plant skeleton structure, Chaudhury et al. [38] proposed a stochastic framework supported by the biological structure of the original plant. Tabb et al. [39] described an efficient and robust method based on breadth-first search, which can extract curve skeletons from a given model of elongated objects affected by noise, which can be applied in agricultural contexts to extract the branching structure of plants.

Jalba et al. [40] proposed a method based on the GPU framework to extract the surfaces and curve skeletons of 3D shapes. Hu et al. [41] presented an approach for skeleton expression based on the set of the cross-section centroids from a point cloud model. Zhou et al. [42] utilized the k-nearest neighbors (KNN) algorithm with an iterative point contraction algorithm to extract the curve skeleton. Several methods for improving the geometric properties and thus suitable for skeletonization have been proposed. Li et al. [32] improving the use of LOP (Locally Optimal Projection) to adaptively contract medial surfaces of 3D models for skeletonization. Song et al. [43] used the distance field to guide the iterative process of $L_1$-medial to obtain a better skeleton compared to [6].

Some algorithms that have a strong correlation with the proposed algorithm are subsequently discussed. These algorithms can extract the curve skeleton from the row scan or missing data, and all have the ability to resist noise. Tagliasacchi et al. [2] proposed an algorithm based on the ROSA of an oriented point set in 2009; this algorithm can extract the curve skeleton from incomplete point clouds in which large portions of the data may be missing. Inspired by the ROSA algorithm, Cao et al. [15] subsequently presented an algorithm for curve skeleton extraction via
Laplacian-based contraction (LBC) in 2010. Because LBC is considered to be an improvement of ROSA, the proposed algorithm is only compared with LBC in this study. In 2013, Huang et al. [6] introduced the $L_1$-medial skeleton ($L_1$-medial) as a curve skeleton, and presented an iterative algorithm to obtain it. In 2020, Qin et al. [16] proposed a Mass-driven topology-aware curve skeleton (MdCS) that displays the mass distribution while retaining geometric properties.

In addition, although the $L_1$-median is robust to noises and outliers, when applied to a raw scan point cloud with holes, it will be biased towards the opposite side of the holes. Therefore, the $L_1$-medial skeleton [6] needs the re-centering process to decrease this deviation. In addition, the iterative computation of $L_1$-medial also evolves a high computational cost. Moreover, OBB and AABB are the most commonly used bounding boxes [30] that can obtain the center point of the given model with missing data. However, the deviation of the center point of AABB may be induced if the model has not been aligned with the axes, whereas the OBB does not lead to such case. Therefore, the center point of the OBB is considered as the skeletal point in this study. Additionally, the data in Table 2 can also prove that the OBB center point is superior to the $L_1$-median as a skeleton point.

Summary. The proposed algorithm differs from these existing algorithms in the following aspects. In the computation process, the proposed algorithm requires the consideration of only the distance between points; it does not need to estimate the normal [2, 15, 16] or curvature [24, 34], and does not require the construction of the connectivity between the points, like voxelization [44, 45], and meshing [34, 46]. Moreover, while the ROSA [2] and LBC [15] methods require the assumption that the input shape is a circular structure, the proposed algorithm has no such limitation. Furthermore, different from the $L_1$-medial [6] and MdCS [16] methods, the proposed EPCS algorithm utilizes the RCS method to detect the structure of the final curve skeleton via endpoint detection before obtaining the rough skeleton, thereby effectively avoiding the loss of skeletal branches.

Figure 2: The pipeline of the proposed method. Given an ostrich model (a), (b) is the raw scan point cloud, 3 endpoints marked as small red balls in (c) are recognized by RCS method. The skeletal points of the right leg are extracted by dividing the leg into layers (d), the layers are marked in red, green, and blue; and each layer corresponds to a skeletal point. Then extract the skeletal points of the left leg (e), head (f) and body (g) using the same process. The obtained skeletal branches are in (h). The junction points are computed for connecting the branches (i-k). Then an interpolation method is used to resample the skeletal points for smoothing the skeletal curves (l). Finally, a complete and smooth curve skeleton (m) is obtained. (n) shows the model segmentation result.

3 Methodology

3.1 Overview

Given an unoriented point cloud $P = \{p_i\}_{i \in J} \subset \mathbb{R}^3$ the proposed EPCS method directly extracts the curve skeleton of $P$. EPCS method contains three simple and easy-to-implement stages. The pipeline of the proposed method is illustrated in Figure 2. First, the endpoints of different parts of $P$ are recognized via the RCS method, and are marked as small red balls in Figure 2 (c). Each endpoint corresponds to a part of the model, and the skeleton of this part is a branch of the skeleton. The endpoints are then used as the seed points for layer division.

Second, each part of $P$ is divided into several layers via region growing from the seed points, as shown in Figure 2 (d-g). The skeletal points are then achieved by computing the centers of the OBB of each layer, as shown in Figure 2
The branches of the skeleton can be obtained by connecting the skeletal points in sequence, as shown in Figure 2 (h). However, to construct a complete curve skeleton, the branches should be connected by the junction points. Therefore, in the third stage, the MVMD method is proposed for both junction points generating and the branches connecting. Particularly, by considering both the local distance and direction information, junction points are generated as shown in Figure 2 (i). However, since the distance and direction information are local information whereas the global information are not considered, redundant junction points may also be generated; thus, the MVMD method is employed again to remove the redundant junction points, and thereby constructing a complete skeleton as shown in Figure 2 (j) and (k). Moreover, an interpolation method based on the splitting operator [31] is also used to smooth the obtained skeletal curves, as shown in Figure 2 (l), and resample the skeletal points to ensure the uniform sampling of the skeletal points. Consequently, a complete and smooth curve skeleton of \( P \) can be obtained, as shown in Figure 2 (m). The result of model segmentation of the \( P \) is shown in Figure 2 (n).

### 3.2 RCS for endpoint detection

The skeletal curve of each part of the given point cloud is considered a branch of the final skeleton. To avoid missing branches, a skeletal curve of each part is first generated by detecting the endpoint of each part. In this study, the endpoints refer to the outermost sharp points of the parts of the point cloud, e.g., the endpoints of an ostrich model are the outermost sharp points on the toes, and head, as shown in Figure 2 (c). The observation of endpoints indicates that the center point of their neighbors quickly shifts with the large increase of the neighborhood radius, whereas the non-endpoints have no such characteristic, as shown in Figure 3. Thus, the novel RCS method is proposed, which uses the shift distance of the center points of different scales of neighborhoods for endpoint recognition. This section details the proposed RCS method.

Given a scattered and unoriented point cloud \( P \), all the points of \( P \) are normalized in range of \([0, 1]\) to avoid the influence of the scales of the given point cloud. Then, the RCS method is performed on the normalized model \( P \), and the detailed process is as follows.

![Figure 3: The illustration of random center shift (RCS) method. The center point (blue ball) of the endpoint (yellow ball) neighbors quickly shifts with the large increase of the neighborhood radius.](image)

The RCS method starts with searching the points on the end regions of the parts of \( P \). Specifically, for each point \( p_i \in P \), its neighboring points are searched using two different scales of radii, i.e., when the search radius is \( r_1 \), the set of neighbors is denoted as \( S_1 \) (the blue points in Figure 3), and when the search radius is \( r_2 \), the set of neighbors is denoted as \( S_2 \), where \( r_1 = \delta \ell, r_2 = r_1 + \epsilon, \epsilon > 0 \), \( \ell \) is the diagonal length of the OBB of \( P \), \( p_o \) is the center point of point set \( S_c \) and \( S_c = \{ p | p \in S_2, p \notin S_1 \} \) (\( p_o \) is marked by a blue ball, \( S_c \) is the green points in Figure 3). Then, the point \( p_i \) that meets Eq. (1) constitutes the point set \( S_i \), \( S_i \) contains the points located on the end regions, such as the yellow points in Figure 4.

\[
\| p_i - p_o \| + \epsilon > r_1.
\]  

Avoiding erroneously regarding outliers as endpoints, it still requires estimating the local sampling density [47] of \( p_i \). It should be noted that the point set \( S_i \) contains all points located on the end regions of \( P \); thus, hierarchical clustering [48] is performed to divide \( S_i \) into different subsets, and each subset only contains points on the same part. For example, as shown in Figure 4, the resulting clusters are the end region points on the four feet, the tail, and the head of the wolf, respectively. The clustering process is summarized in Algorithm 1.
Figure 4: The endpoints are recognized by the RCS method and marked with red balls. The blue balls are the center points of OBB of $S_{part}^T$.

**Algorithm 1** Euclidean distance cluster

**Input:** point set: $S_t$, parameters: $\delta, \ell, \phi$

**Output:** $S_{part}$

1: $S_{part} \leftarrow \emptyset$
2: for $p \in S_t$ do
3:   $Q \leftarrow \emptyset$
4:   if $p.status \neq \text{processed}$ then
5:     $Q.add(p)$
6:     for $q \in Q$ do
7:       $q.status \leftarrow \text{`processed'}$
8:       $N = Knn_q(\delta\ell)$
9:       $Q.addAll(N)$
10:      if size($Q$) $\geq \phi$ then
11:         $S_{part}.add(Q)$
12:     end if
13:   end for
14: end if
15: end for

The subsets of $S_t$ are denoted as $S_{part}$, and $part_i$ represents the $i$-th part of $P$. The endpoints are then computed. For each $p_e \in S_{part}^E$, its neighboring points within radius $\epsilon$ are searched, and the neighboring point set is denoted $S_{part}^E$, e.g., the blue points in Figure 4. Then, for each $p_e \in S_{part}^E$, its neighboring points within radius $\epsilon$ are also searched, and the neighboring point set is $S_{part}^R$, e.g., the green points in Figure 4. To increase the robustness, the OBB of $S_{part}^R$ is then calculated, and the center point (the blue balls in Figure 4) of the OBB is denoted as $p_{part}^c$ instead of directly computing the center points of $S_{part}^R$. Given $p_{part}^c$, the endpoint $p_{p}^i$ of the $i$-th part is defined as follows:

$$p_{p}^i = \max \parallel p_{c}^i - p_p \parallel,$$

where $p_{p}^i$ is the point that has the largest Euclidean distance from $p_{c}^i$. The endpoints of the wolf model are marked with red balls in Figure 4.

### 3.3 skeletal points generation

Given the endpoints, the skeletal points of each part are generated in this stage. The endpoints of $P$ were generated in the previous stage; however, it can be seen from Figure 4 that the body part has no endpoints because it is surrounded by other parts, such as the tail and the legs. If the skeleton for each part is directly generated based on the endpoints, the skeleton of the body part will be missed; in addition, the generation of the skeleton of the body part from any endpoint is also unstable. Therefore, the skeleton of one part is generated in only one iteration, and the processed part will not be considered in the next iteration, which deals with another part; this decreases the influence between each part.

Because the skeleton of one part is generated in only one iteration, the endpoints should first be sorted. The order of the endpoints determines the processing order of the parts and the generation order of the skeletal curves (referred to as
Algorithm 2.

After sorting the endpoints, each part is divided into layers via the classic region growing strategy, as shown in Figure 2 (d), (e), (f), and (g). The entire layer division process consists of a two-layer loop; the outer loop selects an unvisited endpoint, and the inner loop computes the skeletal points of each part.

In the outer loop, the endpoint \( p^* \) that has not been visited is taken as the initial seed. Given the initial seed \( p^* \), its neighboring points within radius \( \rho \) form the point set \( S^i_{layer_1} \), which is the first layer of the \( i \)-th part. Then, the center point \( c^i_1 \) of the OBB of \( S^i_{layer_1} \) is computed, and \( c^i_1 \) is regarded as the first skeletal point of this branch.

Some pre-processing is then required by the inner loop. The average distance \( d^i_l \) between \( c^i_1 \) and all points in \( S^i_{layer_1} \) is computed, and all points in \( S^i_{layer_1} \) are pushed into \( Q_{seed} \), which stores the growing seed points. In the inner loop, each region begins from a growing seed \( p_{seed} \) that popped from \( Q_{seed} \). Then, to obtain \( S^i_{layer_{i+1}} \), the neighboring points of \( p_{seed} \) within radius \( \Delta \) that are not labeled as “visited” form the point set \( S^i_{layer_{i+1}} \), and the center point of the OBB of \( S^i_{layer_{i+1}} \) is computed as a new skeletal point \( c^i_{l+1} \). Finally, the points in \( S^i_{layer_1} \) are labeled as “visited,” and no longer participate in the subsequent processing.

Furthermore, \( d^i_{l+1} \) is also computed, which controls the iteration of the inner loop, i.e., when the region growing process reaches the adjacent part, there is a significant difference between \( d^i_{l+1} \) and \( d^i_l \), thus, when \( |d^i_{l+1} - d^i_l| / d^i_{l+1} > \Phi \), the region growing process should be terminated.

However, the computation of an appropriate value of \( \Phi \) is a non-trivial task due to the various shapes and scales of the parts. Therefore, the linear least-squares method [49] is utilized to predict the value of \( d^p_{l+1} \) of the next layer according to the values of \( d^i \) of the previous layers, and the computation of \( d^i_{l+1} \) takes into account both the shape and scale information of each part. Specifically, we define \( d^i = X\hat{w} \), where \( X = \begin{pmatrix} 1 & 2 & \cdots & l \end{pmatrix} \), \( \hat{w} = (w; b) \). Then the least square method is utilized to estimate \( \hat{w} \):

\[
\hat{w}^* = \arg\min_{\hat{w}} E_{\hat{w}}.
\]

\[
= \arg\min_{\hat{w}} (d^i - X\hat{w})^T (d^i - X\hat{w}). \tag{3}
\]

Subsequently, by setting \( \frac{\partial E_{\hat{w}}}{\partial \hat{w}} = 2X^T(X^T\hat{w} - d^i) = 0 \); then \( \hat{w} \) can be obtained. Thus, \( d^p_{l+1} \) is estimated by \( d^p_{l+1} = w(l + 1) + b \). Therefore, the value of \( |d^i_{l+1} - d^p_{l+1}| / d^i_{l+1} \) will not largely change with different shapes and scales of different parts, and \( \Phi \) is therefore more stable. Therefore, when \( d^p_{l+1} \) and \( d^i_{l+1} \) meet Eq. (4), the growing of the current region is terminated.

\[
|d^i_{l+1} - d^p_{l+1}| / d^i_{l+1} > \Phi. \tag{4}
\]

Then, a new initial seed (the endpoint) is selected to divide the layers to process another part. Ideally, each part has an endpoint; however, some parts that are surrounded by the other parts will have no endpoints, and will not be processed by the procedure described previously. For example, consider the octopus models in Figure 12. Hence, among the points that have not been labeled as “visited”, the unvisited point \( p_x \) that is farthest from the center point \( p_c \) of the OBB of the \( P \) is taken as the initial seed to continue the region growing procedure. When there is no new initial seed and all points on the point cloud have been visited, the algorithm is terminated. The region growing pseudocode is shown in Algorithm 2.

**Small Radius Constraint.** When dealing with a model like the hand shown in Figure 6 (k), a too-large value of \( \Delta \) will affect the extraction effect. The gap between different fingers will be smaller than \( \Delta \), which will lead to errors in region growing. To prevent this, \( \Delta \) is refined. The \( l \) cycle in the inner loop is divided into \( \tau \) times, the radius of each time is \( \rho = \Delta / \tau \), and the points are searched for \( \tau \) times to obtain \( S^i_{layer_{i+1}} \). With the help of small radius constraint, noises and outliers which are not considered when computing the skeletal points, as shown in Figure 2 (n).

**3.4 Curve skeleton generation**

The skeletal points were generated in a linear sequence in the previous stage, as they were generated during the region growing process. Therefore, the skeletal curves of each part can be formed by directly connecting the skeletal points in the linear sequence; thus, the skeletal curves are formed as the branches of the final curve skeleton, as shown in Figure 2 (h).
Algorithm 2 skeletal curves extraction

Input: $P, p_c$, endpoints set $p^P$, parameter $\Phi$
Output: skeletal curve set $C$

1: function EXTRACT($P, p_c, p^P, \Phi$)
2: $V \leftarrow \emptyset, C \leftarrow \emptyset, i \leftarrow 0$
3: while size$(V) < size(P)$ do
4:   if $i < size(p^P)$ then $p_x \leftarrow p_i^P$
5:   else $p_x \leftarrow \text{GetStartPoint}(P, V, p_c)$
6:   end if
7:   $i \leftarrow i + 1, l \leftarrow 0, c = \emptyset$
8:   while $p_x$ do
9:     if $l = 0$ then $S_{seg}^i \leftarrow q_1^i, d_1^i \leftarrow p_x$
10:    end if
11:   $S_{seg}^i \leftarrow q_{l+1}^i, d_{l+1}^i \leftarrow S_{seg}^i, q_l^i, d_l^i$
12:   if $l > 2$ then
13:     $d_{l+1}^p \leftarrow \text{predict}(\{d_1^i, d_2^i, \ldots, d_l^i\})$
14:    $\lambda \leftarrow |d_{l+1}^p - d_{l+1}^i|/d_{l+1}^i$
15:   else $\lambda \leftarrow 0$
16:   end if
17:   if $\lambda < 0$ then $c.add(q_l^i), V.add(S_{seg}^i)$
18:   else break
19: end if
20: $l \leftarrow l + 1$
21: end while
22: $C.add(c)$
23: end while
24: end function
25: function GETSTARTPOINT($P, V, p_c$)
26: $U \leftarrow \{p_u | p_u \in P, p_u \notin V\}$
27: $p^x \leftarrow \text{max}(\|p_u - p_c\|)$
28: if $K_{mp}(\delta \ell) > 10$ then return $p^x$
29: else $\text{GetStartPoint}(P, V.add(p^x), p_c)$
30: end if
31: end function

Given the branches, the final skeleton is constructed in this stage, which completes two main tasks. First, the proper junction points for connecting branches are located by the proposed MVMD method. Second, the global topology of the skeleton is optimized by reducing the redundant junction points and then re-connecting the branches using the proposed MVMD method. Furthermore, the distribution of skeletal points is non-uniformly sampled due to the shape differences between different parts of the models, which considerably affects the smoothness of the skeletal curves. We therefore apply an interpolation method based on the 1-4 splitting operator [47] to smoothing the skeletal curves and subsequently resampling to make sure the skeletal points are uniform. Finally, a complete and smooth curve skeleton can be obtained.

3.4.1 MVMD for locating junction points

The junction points are referred to as the points that connect at least two branches of the skeleton, and are crucial for the entire shape of the final skeleton. Therefore, both the distance and direction vectors are considered as multi-vectors for the selection of the junction points, i.e., multi-vectors are formed according to the nearest distance and direction of the current branches, both of which determine the location of the junction points. The details are as follows.

For each branch, the last skeletal point $c_{n_j}^k$ that is the farthest from the endpoint $p_j^P$ is selected as the target point, as shown in Figure 5 (a). $c_{n_j}^k$ represents the last skeletal point on the $j$-th branch, and $n_j$ denotes the number of skeletal points on the $j$-th branch. The multi-vectors are then computed. The first vector is $\vec{v}_0$, which is formed by the points $c_{n_j}^k$ and $c_{n_j-1}^k$, among which $c_{n_j-1}$ is the adjacent skeletal point of $c_{n_j}^k$ on the $j$-th branch; thus, the direction of the $j$-th branch can be retained. The other vector $\vec{d}_0$ is formed by points $c_{n_j}^k$ and $c_m^k$, among which $c_m^k$ is the $m$-th skeletal.
point on the $k$-th branch (there are $n_k$ points on the $k$-th branch) that meets $c^k_m = \min_{x=1 \sim n_k} ||c^j_{n_j} - c^k_x||$. The definitions of $\vec{v}_0$ and $\vec{d}_0$ are as follows.

$$
\vec{v}_0 = c^j_{n_j} - c^j_{n_j-1} \\
\vec{d}_0 = c^m_k - c^j_{n_j}.
$$

(5)

Given $c^j_{n_j}$, $\vec{v}_0$ and $\vec{d}_0$, the next point $c^j_{n_j+1}$ is computed using Eq. (6), which is a linear combination of $\vec{v}_0$ and $\vec{d}_0$, as shown in Figure 5 (a) and (b).

$$
\vec{v}_t = c^j_{n_j+t} - c^j_{n_j+t-1} \\
\vec{d}_t = c^{k_1}_{m_1} - c^j_{n_j+t} \\
c^j_{n_j+t+1} = c^j_{n_j+t} + \mu \left( \alpha \cdot \vec{v}_t + (1 - \alpha) \cdot \vec{d}_t \right),
$$

(6)

where $\alpha$ is balancing factor among $\vec{v}_t$ and $\vec{d}_t$ by using the concept of momentum.

By repeating the preceding procedure, the points are moved forward as driven by the multi-vectors, as shown in Figure 5 (b). The repeating process terminates when $c^j_{n_j+t+1}$ meets

$$
||c^j_{n_j+t+1} - c^{k_1}_{m_1}|| < \frac{\ell}{20},
$$

which indicates two branches have been connected (as shown in Figure 5 (b) and (c)), or when the moving step reaches 10, which means the two branches cannot be connected because they are too far away.

The main parameters controlling the moving process are $\mu$ and $\alpha$. The $\mu = \frac{\ell}{20}$ is set by default, and the value of $\alpha$ is computed by Eq. (7). Moreover, $t$ is the moving step number, and the value of $\alpha$ decreases with the increase of $t$.

$$
\alpha = \frac{1}{\sqrt{2\pi}} \exp \left( - \frac{(t/5)^2}{2} \right),
$$

(7)

thus, as $t$ gradually increases, $\alpha$ decreases; correspondingly, the weight of $\vec{v}_t$ decreases and the weight of $\vec{d}_t$ increases. In other words, in the beginning steps, the direction information largely affects the location of $c^j_{n_j+t+1}$, whereas in the last several steps, the location of $c^j_{n_j+t+1}$ is mainly affected by the distance information. Via the momentum-driven dynamic coefficient setting, the MVMD method can locate more appropriate junction points.

By performing the MVMD method, the independent branches are connected, and a complete skeleton can be obtained. Furthermore, the resulting skeleton is smoother because the branches are connected by interpolating points, as shown in Figure 5 (a), instead of by directly generating a straight line to connect the branches.
3.4.2 Global topology optimization

The previous stages can yield a complete skeleton for the given point cloud; however, a problem remains because several redundant junction points may also be generated. This is because the junction points are located by only considering the local direction and distance information. The global topology of the skeleton therefore requires optimization.

To optimize the global topology of the skeleton, all the junction points are first checked by computing the Euclidean distances between their adjacent junction points, e.g., junction points $c_{k1}$ and $c_{k2}$ in Figure 5 (d). For the sake of a clear representation, the adjacent junction point pair $\langle c_{jn}, c_{km} \rangle$ is denoted as $\langle c_{j}, c_{k} \rangle$. For $c_{k1}$ and $c_{k2}$, there are two pairs, namely $\langle c_{j1}, c_{k1} \rangle$ and $\langle c_{j2}, c_{k2} \rangle$. If the Euclidean distance between them is less than $\zeta$, one of the two junction points is redundant, and one is removed using the following procedures; on the contrary, these two junction points should be retained.

For a junction point pair that has redundant junction points, Eq. (8) is used to identify the redundant junction points, and the branch is then connected with the remaining junction point, as shown in Figure 5 (d).

\[
\begin{align*}
    d_1 &= \| c_{k1} - c_{j1} \| + \| c_{k1} - c_{j2} \|, \\
    d_2 &= \| c_{k2} - c_{j1} \| + \| c_{k2} - c_{j2} \|,
\end{align*}
\]

(8)

if $d_1 < d_2$, $c_{k1}$ is considered a common junction point among $c_{j1}$ and $c_{j2}$, and vice versa.

The topology optimization process removes the redundant junction points, as shown in Figure 5 (e); however, it also results in some independent branches that were previously connected with the redundant junction, e.g., branches 1 and 3 in Figure 5 (e). Therefore, these independent branches are re-connected with the remaining nearest junction points using the MVMD method described in Section 3.4.1. By using the MVMD method, which employs the interpolation operation to complete the connection task, a complete and smooth skeleton can be constructed.

Specifically, the goal described in Section 3.4.1 is to locate the junction point; here, the junction point is already known and denoted as $c_{b}$. Therefore, instead of $c_{m}$ and $c_{nt}$, only $c_{b}$ needs to be used to rewrite $\vec{d}_t = c_{b} - c_{nj}$ and $\vec{d}_t = c_{b} - c_{nj+1}$ With the help of $\vec{d}_n, \vec{d}_{n+1}$ complete the interpolation process, the obtained $c_{nt}$ is considered as the skeletal point, as shown in Figure 5 (e).

Figure 6: The curve skeleton extracted by EPCS from different point cloud models. The number below each model is the number of points in the model. The last column is the model segmentation result corresponding to the model in the previous column.
Due to the shape differences between different parts of the model, the distribution of the skeletal points is non-uniform, resulting in not smooth skeletal curves. By interpolating and resampling the skeletal curves, the distribution of the skeletal points can be uniform, which also makes the skeletal curves smoother. Thus an interpolation method based on the 1-4 splitting operator [47] is adopted. Specifically, for a skeletal curve, we interpolate until the distance between any two skeletal points of the branch is less than $\frac{1}{5} \eta$, as shown in Figure 2 (l). Then we down sample the skeletal points using $\eta$ as the sampling density, finally, all skeletal points are reconnected and a complete EPCS can be obtained, as shown in Figure 2 (m).

4 Result and analysis

To demonstrate the effectiveness and efficiency of the proposed EPCS algorithm, experiments were performed on various types of point clouds, including models with diverse noises (Figure 2, Figure 7, Figure 9, Figure 10, and Figure 11), models with missing data (Figure 1, Figure 2, Figure 7, Figure 9, Figure 10, and Figure 15), models with outliers (Figure 2, Figure 7, Figure 9, Figure 10, and Figure 12), raw scan point clouds (Figure 2, Figure 7, Figure 9, and Figure 10), and the real scanned Terracotta dataset (Figure 1 and Figure 15). Furthermore, to highlight the advantages of the proposed method, it was also compared with several state-of-the-art methods, including the LBC [15], $L_1$-medial [6] and MdCS [16] methods. The codes of the LBC [15], $L_1$-medial [6] and MdCS [16] methods were provided by the authors, and all the parameters were fine-tuned. To perform the experiments, a computer equipped with an AMD 3700X CPU and an Nvidia RTX 2060S GPU was used.

4.1 Parameters

The default parameters of the proposed algorithm are listed in Table 1. $\eta$ determines the spacing between the skeletal points on the curve skeleton, the larger $\eta$ is, the farther between the skeletal points, and the sparser the skeletal points. As shown in Figure 7. $\Phi$ is an important parameter for controlling the region growing process. With the help of the least-squares method, $\Phi$ converges to 0.2 for different model. Moreover, $\zeta$ is a parameter for topology optimization, which is used for merging junction points. When $\zeta$ increases, the junction points will decrease, and vice versa. From Figure 8 (a), (b) and (c), it can be seen that when $\zeta$ increases from 0 to 0.1, the three junction points marked in the yellow dotted circle are merged together, and when increasing from 0.1 to 0.2, the two junction points marked in the green circle are also merged together. The distance between the junction points is less than 0.2, so $\zeta$ increases from 0.2 to 0.3, no new merging occurs. The parameters $\epsilon$, $\delta$, $\phi$, and $k_{LD}$ are fixed in our algorithm.

| Parameter | $\Phi$ | $\eta$ | $\Delta = \tau \times \rho$ | $\zeta$ | $\epsilon$ | $\delta$ | $\phi$ | $k_{LD}$ |
|-----------|--------|--------|------------------|--------|----------|--------|--------|---------|
| Default Value | 0.2    | 0.1    | 0.1 = 4 \times 0.025 | 0.2    | 0.02    | 0.0625 | 5      | 5       |

Figure 7: $\eta$ determines the spacing between the skeletal points on the curve skeleton.

Figure 8: $\zeta$ determines the degree to which the junction point is removed.
4.2 Raw scans results

This section compares the actual performance of several algorithms on the raw scans. The difficulty of the raw scans for the curve skeleton extraction task is that, no unprocessed before, the raw scans often have holes leading to missing data, and there will be noises and outliers interference around the model. The results are shown in Figure 9.

It can be seen from Figure 9 that the LBC [15] algorithm is sensitive to outliers; although it can deal with holes, under the influence of both noises and outliers, the curve skeleton of LBC [15] is no longer available. The $L_1$-medial skeleton [6] cannot converge due to the growing radius of the neighborhood ball, resulting in the absence of skeletal branches. Outliers affect the quality of MdCS [16] since they considerably affect the symmetry of the topology check, manifesting as missing branches. For EPCS, since EPCS predicts the curve skeleton topology in advance via RCS method for endpoints detection and avoids the lack of skeleton branches, the extracted curve skeletons are more complete than other methods. In skeletal points extraction, by setting a small radius constraint on the region growing process, outliers can be excluded and the region growing process starting from the outliers cannot be initiated, thus avoiding the influence of outliers on the extraction of skeletal points. Furthermore, since the principal direction of the point set is calculated in the construction of OBB, the interference of noises and outliers can also be reduced to some extent, thereby reducing the disturbance to skeletal points. Additionally, though there are holes that cause data missing in raw scan point clouds, since OBB is not sensitive to missing data, even if there are holes that cause data missing in raw scan point clouds, it will not have a large impact on OBB, thus ensuring better centering of skeletal points. In order to further reduce the influence of noises, outliers and missing data on the curve skeleton, the MVMD method is designed to globally and locally optimize the extracted skeleton, and a smoothing method is also utilized to smooth the skeleton curves, thereby improving the quality of the curve skeleton.
4.3 Robustness Analysis

This section presents results on models disturbed by missing data, noise, and outliers, to verify the robustness of the proposal method.

4.3.1 Quantitative Analysis

**Missing data.** Figure 10 presents the results of the four algorithms when dealing with point clouds with missing data. In the experiment, \( f \) is the number of raw scan frames, and \( f = 2, 3, 4, 5 \). When \( f \) increases, the horse model is gradually complete. Because there are holes, noises, and outliers in raw scans, the test is more strict.

![Figure 10: The skeleton of raw scans with missing data.](image)

As shown in Figure 10, the \( L_1 \)-medial [6] and MdCS [16] methods yielded inadequate solutions for the multi-branch model, which indicates that the LBC [15], \( L_1 \)-medial [6], MdCS [16] methods all encountered challenges in skeleton extraction on multi-branch models with missing data. LBC [15] cannot handle models with outliers, thus unavailable skeletons are extracted. With \( f \) increasing, the \( L_1 \)-medial skeleton becomes more and more complete, and when \( f = 5 \), although the skeleton is extracted, a connection error between the tail and legs is produced. MdCS [16] performs poorly due to noises and outliers interfering with topology checking.

Since the influence of noises and outliers are taken into account in the RCS method, when \( f > 2 \), the detected endpoint is ideal and can be used as a seed point for skeletal points generation. Although the invalid endpoints are detected at the horse’s ear, the final skeleton extraction result is not affected. This is because the adaptive radii \( r_i^p \) of the invalid endpoints are significantly larger than the valid endpoints, making the region growing process starting from the invalid endpoints initiate later; however, the neighbors of the invalid endpoints have been visited by the other region growing process, and thereby the region growing starting from the invalid endpoints will not initiate, and no skeletal points are generated.

**Noise.** For the experiment on models with Gaussian noises, the standard deviation of Gaussian noise is defined as \( \sigma d_{\text{min}} \), where \( d_{\text{min}} \) is the minimum distance between points in the point cloud model. The Gaussian noise disturbance experiment is expressed as \( \text{Noise}(\sigma) \), and \( \text{Noise}(\sigma = 0) \) refers to no Gaussian noise added to the model. The experimental results are shown in Figure 11.
It can be seen from Figure 11 that when $Noise(\sigma = 0)$, the LBC [15], $L_1$-medial [6], MdCS [16], and EPCS methods can extract a complete and correct curve skeleton; however, when $Noise(\sigma = 2)$, the LBC [15] method produced a loop structure on the chest of the human model, the $L_1$-medial [6] method produced an incomplete curve skeleton because the body and the head were not connected, and the MdCS [16] method produced the worst curve skeleton with incorrect branches and connection relationships. When $Noise(\sigma = 4)$ occurred, the connectivity among the skeleton curves of the $L_1$-medial [6] method continued to deteriorate, and the MdCS [16] method yielded two adjacent skeleton curves on the abdomen of the human model. With the further increase in the scale of noise, such as $Noise(\sigma = 6)$, the LBC [15] method resulted in redundant branches. The $L_1$-medial [6] method was characterized by distortion in the processing of the legs. The MdCS [16] method produced the same error as in the case of $Noise(\sigma = 2)$; the junction points of the skeleton curve were selected incorrectly. However, the proposed EPCS algorithm achieved a satisfactory result despite the interference of these types of noise.

Outliers. In the experiments on models with outliers, $o$ was utilized to represent the number of outliers in the model. $o$ is the ratio of the number of outliers added to the models; specifically, $o = 2, 4, 6$, means $2\% \times I, 4\% \times I, 6\% \times I$ outliers are added to the point cloud, where $I$ represents the number of the points on the original model. The outliers were restricted to a space 1.2 times larger than the point cloud model space. As shown in Figure 12, the LBC [15] method cannot handle models with outliers and almost no effective results are obtained, this is because when the distance between the outliers and the original point of the model is too close, the Delaunay triangulation algorithm used by the LBC [15] method may fail. Therefore, for this experiment, only the $L_1$-medial [6], MdCS [16] and the proposed method were evaluated.

It can be seen from Figure 12, even when $o = 0$, the skeleton extracted by the $L_1$-medial [6] method exhibits topological errors, although the skeleton was successfully extracted on the feet of the octopus, which reflected better performance than for the horse model shown in Figure 10. This is because the change in the radius of the local neighborhood ball...
between the horse’s leg and body was greater than the radius of the local neighborhood ball between the leg of the octopus and the body, which eased the contradiction between the initial local neighborhood value and the growth rate to a certain extent. Regarding the MdCS [16] method, the outliers also caused more serious interference to the topology check. In particular, for the topology check, principal component analysis (PCA) is primarily used to detect the symmetry of the transport plan between points, and outliers will greatly interfere with the results of the PCA algorithm; thus, the topology error of the MdCS [16] method gradually became serious. For the $L_1$-medial [6] method, with the increase of $\sigma$, the outliers introduced interference to the increase of the local neighborhood ball radius, resulting in the gradual distortion of the extracted skeleton. Furthermore, due to the existence of the sampling process, the possibility of using outliers as sampling points progressively increased, and because there were no other points around the outliers, they could not be transmitted to other branches by the algorithm, resulting in a greater lack of skeletal branches. For the EPCS method, outliers leading to introduce challenges to the endpoint recognition algorithm, resulting in more endpoints that affect subsequent steps. However, since the radius of the endpoint is adaptively computed, the radius of the outliers is infinitely close to 0. Furthermore, even if an outlier is used as a seed point, the region growing process cannot be started due to the absence of neighboring points. Therefore, even if an outlier poses a challenge to endpoint recognition, it will not affect the extraction of the curve skeleton. However, when the outliers are located close enough to the point cloud, slight disturbance to the curve skeleton will occur, but these changes will not considerably affect the quality of the skeleton. In summary, the proposed algorithm is the least affected by outliers, and a valid curve skeleton can still be extracted when $\sigma = 6$.

Quantitative Analysis. To quantitatively analyze the quality of curve skeleton extraction results from point clouds with different types of disturbance, mean curvature skeleton (MCS) [34] extracted from the clean mesh model is considered as the ground truth [16]. The Hausdorff distance is a commonly used metric for evaluating the differences between two skeletal curves [9], [16], we thus adopt two-sided Hausdorff distances to evaluate these methods.

Table 2 and Figure 13 shows the detailed data of $h(A, B)$ and $h(B, A)$. $h(A, B)$ shows that the Hausdorff distance data of EPCS is significantly smaller than that of $L_1$-medial [6] and MdCS [16], indicating that the skeleton extracted by EPCS is more complete and reasonable than that of $L_1$-medial [6] and MdCS [16]. Moreover, Figure 11 shows that the LBC [15] performs better when $\sigma = 0$ and $\sigma = 1$ than in $L_1$-medial [6] and MdCS [16], which is consistent with that in Table 2.
The results of EPCS are all less than 0.1 in the three cases of \( f = 6, \sigma = 0 \) and \( o = 0 \), which indicates that EPCS is closer to the ground truth than the other three methods. When \( \sigma \) and \( o \) increase, \( h(A, B) \) of EPCS are always in a relatively stable interval, which verifies the anti-noise and outlier interference ability of EPCS.

\( h(B, A) \) shows the gap between the curve skeleton and ground truth extracted by several methods, ignores the absence of skeletal branches, and reveals the neutral difference of the skeletal points of several curve skeletons. It can be seen from Table 2 that the skeletal point of LBC [15] is closer to the ground truth under the premise that the skeleton can be obtained. EPCS is slightly worse than LBC [15] since the topology structure of EPCS near the junction points is generated by MVMD method, whereas both LBC [15] and MCS [34] are obtained by contraction, but EPCS still better than \( L_1 \)-medial [6] and much better than MdCS [16], furthermore, these conclusions are also proven in Figure 13. When dealing with the raw scans (with noises, outliers, and missing data), with the increase of frames, the EPCS is closer to the ground truth.

### 4.3.2 Efficiency.

The efficiency of the proposed method was compared with that of the LBC [15], \( L_1 \)-medial [6], and MdCS [16] methods in this section. The models shown in Figure 6 were employed for testing. The scale of these models varies from 3101 to 33041. Table 3 presents the execution time consumed by the three baseline algorithms and the proposed algorithm.

It can be seen that the LBC [15] method achieved relatively stable efficiency, i.e., as the number of points increased, the time consumption steadily increased. The MdCS [16] method is implemented using MATLAB; thus, the execution time was found to be unsatisfactory. As shown in Figure 6 (f), when calculating the dog model, which contained only 9009 points, the MdCS [16] method still took 213 seconds. Although the \( L_1 \)-medial [6] method achieved better performance than that LBC [15] and MdCS [16] methods, some extracted skeletal branches were missing, as shown in the second row in Figure 12.

As shown in Table 3 and Figure 14, the EPCS method achieved the best performance with respect to efficiency. Because this algorithm contains no resampling process, all points of the model participate in the computation process; thus, an increase in the point number of the model directly leads to the increase of the time consumption. On average, EPCS is 11.5 times faster than LBC [15], 5.1 times faster than \( L_1 \)-medial [6], and 311 times faster than MdCS [16].
Table 2: Quantitative comparison of curve skeleton result.

| Horse model with missing data, Figure 13 (a) | A | B | \( h(A, B) \) | \( h(B, A) \) |
|---------------------------------------------|---|---|----------------|----------------|
| \( L_1 \)- medial                         |   |   | \( f = 2 \)  | \( f = 3 \)  |
| MCS            | 1.186 | 1.088 | 1.158  | 1.161  |
| MdCS          | 0.601 | 0.622 | 0.632  | 0.589  |
| EPCS          | 0.172 | 0.207 | 0.266  | 0.056  |

Human model with noises, Figure 13 (b).

| Human model with noises, Figure 13 (b) | A | B | \( h(A, B) \) | \( h(B, A) \) |
|----------------------------------------|---|---|----------------|----------------|
| \( L_1 \)- medial                      |   |   | \( \sigma = 0 \) | \( \sigma = 2 \) |
| LBC            | 0.145 | 0.148 | 0.154  | 0.158  |
| MdCS          | 0.155 | 0.151 | 0.148  | 0.143  |
| EPCS          | 0.093 | 0.072 | 0.071  | 0.071  |

Octopus model with outliers, Figure 13 (c).

| Octopus model with outliers, Figure 13 (c) | A | B | \( h(A, B) \) | \( h(B, A) \) |
|--------------------------------------------|---|---|----------------|----------------|
| \( L_1 \)- medial                           |   |   | \( o = 0 \)  | \( o = 2 \)  |
| LBC            | 0.777 | 0.560 | 0.486  | 0.711  |
| MdCS          | 0.761 | 0.763 | 0.753  | 0.751  |
| EPCS          | 0.091 | 0.108 | 0.087  | 0.088  |

Figure 14: Efficiency comparison of LBC [15], \( L_1 \)- medial [6], MdCS [16], EPCS on models in Figure 6. MdCS [16] only shows four models which have the fewest vertices. The bar chart shows their time-consuming range.

4.4 Application

The curve skeletons of damaged cultural relics can visually display the entire pose of the relics; furthermore, they can also guide virtual and real restoration processes. However, the extraction of the curve skeletons of damaged cultural relics is a challenging problem. Even if broken cultural relics are manually restored, there may remain interference from large holes (missing data), noises, and outliers; this will lead to lower-quality models and ultimately result in the failure of many typical skeleton extraction algorithms.

The proposed EPCS method was applied on the models from Terracotta dataset, a real scanned dataset, and the method still achieved high performance. As can be seen from Figure 15, when extracting the curve skeleton, the model segmentation results could also be obtained (as shown in the fourth column). It also can be seen from Figure 15 that there were several holes in both the human figurine and horse models; however, the EPCS method still performed well. Regarding the Paoding warrior model in the last row, which is incomplete because both the arms are missing, the EPCS model still yielded a complete skeleton that was closer to the real pose of the model.
Table 3: The computation time of LBC [15], $L_1$-medial [6], MdCS [16], and EPCS, unit (s). $Rotia_1$ is the time-consuming ratio between LBC [15] and EPCS, $Rotia_2$ is the ratio between $L_1$-medial [6] and EPCS, and $Rotia_3$ is the ratio between MdCS [16] and EPCS.

| Models     | #Vertices | LBC    | $L_1$-media | MdCS  | EPCS   | $Rotia_1$ | $Rotia_2$ | $Rotia_3$ |
|------------|-----------|--------|-------------|-------|--------|-----------|-----------|-----------|
| a-ox       | 30952     | 132.630| 33.030      | 5902.718 | 10.553 | 12.57   | 3.13      | 559.34    |
| b-spider   | 22406     | 66.258 | 53.340      | 2084.692 | 5.793  | 11.44   | 9.21      | 359.86    |
| c-deer     | 11729     | 33.124 | 18.670      | 302.403 | 2.615  | 12.67   | 7.14      | 115.64    |
| d-plane    | 16553     | 38.923 | 30.900      | 973.587 | 4.261  | 9.13    | 7.25      | 228.49    |
| e-jellyfish| 3101      | 10.144 | 4.890       | 42.593 | 2.510  | 4.04    | 1.95      | 16.97     |
| f-dog      | 9009      | 29.711 | 15.490      | 213.076 | 3.466  | 8.28    | 4.47      | 61.48     |
| g-sheep    | 10436     | 29.472 | 17.580      | 271.891 | 2.989  | 9.86    | 5.88      | 90.96     |
| h-spider   | 24047     | 88.294 | 24.800      | 2370.769 | 7.046  | 12.53   | 3.52      | 336.47    |
| i-horse    | 14680     | 34.460 | 18.120      | 497.692 | 3.268  | 10.54   | 5.54      | 152.29    |
| j-wolf     | 14132     | 33.967 | 20.800      | 483.934 | 3.558  | 9.55    | 5.85      | 136.01    |
| k-hand     | 20969     | 91.344 | 76.250      | 1520.461 | 5.092  | 17.94   | 14.97     | 298.60    |
| l-cheetah  | 18159     | 46.356 | 16.420      | 1507.367 | 4.489  | 10.33   | 3.66      | 335.79    |
| m-giraffe  | 27711     | 104.881| 15.800      | 2844.746 | 10.105 | 10.38   | 1.56      | 281.52    |
| n-human being | 33041 | 182.268| 57.220      | 6258.986 | 13.081 | 13.93   | 4.37      | 478.48    |
| o-plane    | 15128     | 36.318 | 21.530      | 527.902 | 4.082  | 8.90    | 5.27      | 129.32    |
| mean       | 18137     | 63.810 | 28.323      | 1720.188 | 5.527  | 11.54   | 5.12      | 311.22    |

Figure 15: The curve skeleton and model segmentation results of EPCS on the point clouds of broken Terracotta.

4.5 Limitations

The skeletonization of point clouds is usually an ill-posed problem [6], [16], especially in the case of missing data. When there is too much missing data or serious noises, the EPCS will be distortion. Moreover, the EPCS method cannot handle the situation in which the branches of the model are stuck together, due to the characteristic of the region growing process.

5 Conclusion

In this study, the novel RCS method was designed for identifying the endpoints which are regarded as the seed points of region growing. Then the region growing was applied to complete the task of curve skeletal point computation. The complete topology of the curve skeleton was subsequently obtained using the MVMD method. Therefore, a simple and efficient curve skeleton extraction method, namely the EPCS, was proposed. The results of experiments demonstrate that the EPCS method is robust to unorganized, non-uniform sampling and un-oriented point cloud models with missing data, noises, and outliers; furthermore, the proposed EPCS also outperforms the other selected methods in terms of the efficiency. The EPCS can be applied on skeleton extraction on the models of broken cultural relics, highlighting the utilization of the proposed method. In light of the demand for real-world point cloud processing such as object...
recognition and classification, we further propose a novel deep network which uses EPCS as input to jointly complete these tasks.

Acknowledgments

The authors would like to thank all the reviewers for their valuable comments, the providers of the datasets and the contributors of Point Cloud Library. The authors would like to thank Tagliasacchi et al. (ROSA), Cao et al. (LBC), Huang et al. (L1-medial skeleton), and Qin et al. (MdCS) for making an executable software or provide source code. This work was supported in part by the National Natural Science Foundation of China (NSFC) (61802311, 61772421, 61731015, 61802244, and 61602380). Associate Prof. Yuhe Zhang is the corresponding author of this paper.

References

[1] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, and A. Telea, “3d skeletons: A state-of-the-art report,” Computer Graphics Forum, vol. 35, no. 2, p. 573–597, 2016.
[2] A. Tagliasacchi, H. Zhang, and D. Cohen-Or, “Curve skeleton extraction from incomplete point cloud,” ACM SIGGRAPH 2009 papers, p. 1–9, 2009.
[3] D. Reniers and A. Telea, “Skeleton-based hierarchical shape segmentation,” in IEEE International Conference on Shape Modeling and Applications 2007 (SMI’07). IEEE, 2007, p. 179–188.
[4] J. Tang, Y. Wang, X. Ning, and K. Lv, “Point cloud decomposition by internal and external critical points,” Computers Graphics, vol. 102, p. 18–29, 2022.
[5] S. Xie, J. Liu, and C. D. Smith, “Curve skeleton-based shape representation and classification,” in 2012 19th IEEE International Conference on Image Processing. IEEE, 2012, p. 529–532.
[6] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen, “L1-medial skeleton of point cloud,” ACM Transactions on Graphics, vol. 32, no. 4, pp. 65:1–65:8, 2013.
[7] S. Wu, H. Huang, M. Gong, M. Zwicker, and D. Cohen-Or, “Deep points consolidation,” ACM Transactions on Graphics (ToG), vol. 34, no. 6, p. 1–13, 2015.
[8] C. Lin, C. Li, Y. Liu, N. Chen, Y.-K. Choi, and W. Wang, “Point2skeleton: Learning skeletal representations from point clouds,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, p. 4277–4286.
[9] Z. Dou, C. Lin, R. Xu, L. Yang, S. Xin, T. Komura, and W. Wang, “Coverage axis: Inner point selection for 3d shape skeletonization,” arXiv preprint arXiv:2110.00965, 2021.
[10] J. Ma, S. W. Bae, and S. Choi, “3d medial axis point approximation using nearest neighbors and the normal field,” The Visual Computer, vol. 28, no. 1, p. 7–19, 2012.
[11] M. Wan, Z. Liang, Q. Ke, L. Hong, I. Bitter, and A. Kaufman, “Automatic centerline extraction for virtual colonoscopy,” IEEE transactions on medical imaging, vol. 21, no. 12, p. 1450–1460, 2002.
[12] R. Li, G. Bu, and P. Wang, “An automatic tree skeleton extracting method based on point cloud of terrestrial laser scanner,” International Journal of Optics, vol. 2017, 2017.
[13] L. Li, W. Wang, and Y. Chu, “A simple and stable centeredness measure for 3d curve skeleton extraction,” IEEE Transactions on Visualization and Computer Graphics, 2020.
[14] S. Xie, J. Liu, and C. D. Smith, “Curve skeleton-based shape representation and classification,” in 2012 19th IEEE International Conference on Image Processing. IEEE, 2012, pp. 529–532.
[15] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su, “Point cloud skeletons via laplacian based contraction,” in 2010 Shape Modeling International Conference. IEEE, 2010, p. 187–197.
[16] H. Qin, J. Han, N. Li, H. Huang, and B. Chen, “Mass-driven topology-aware curve skeleton extraction from incomplete point clouds,” IEEE transactions on visualization and computer graphics, vol. 26, no. 9, p. 2805–2817, 2019.
[17] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Van Gool, and W. Purgathofer, “A survey of urban reconstruction,” in Computer graphics forum, vol. 32, no. 6. Wiley Online Library, 2013, pp. 146–177.
[18] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva, “State of the art in surface reconstruction from point clouds,” in Eurographics 2014-State of the Art Reports, vol. 1, no. 1, 2014, p. 161–185.
[19] Y. Chang, C. Jung, and Y. Xu, “Finerpcn: High fidelity point cloud completion network using pointwise convolution,” Neurocomputing, vol. 460, p. 266–276, 2021.
[20] O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee, “Skeleton extraction by mesh contraction,” ACM transactions on graphics (TOG), vol. 27, no. 3, p. 1–10, 2008.
[21] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, “Consolidation of unorganized point clouds for surface reconstruction,” ACM transactions on graphics (TOG), vol. 28, no. 5, p. 1–7, 2009.
[22] R. L. Ogniewicz and M. Ilg, “Voronoi skeletons: theory and applications,” in CVPR, vol. 92, 1992, p. 63–69.
[23] L. Lu, B. Lévy, and W. Wang, “Centroidal voronoi tessellation of line segments and graphs,” Computer Graphics Forum, vol. 31, no. 2p4.
[24] M. Chuang and M. Kazhdan, “Fast mean-curvature flow via finite-elements tracking,” Computer Graphics Forum, vol. 30, no. 6. Wiley Online Library, 2011, p. 1750–1760.
[25] N. Vahrenkamp, E. Koch, M. Wächter, and T. Asfour, “Planning high-quality grasps using mean curvature object skeletons,” IEEE Robotics and Automation Letters, vol. 3, no. 2, p. 911–918, 2018.
[26] A. Bærentzen and E. Rotenberg, “Skeletonization via local separators,” in ACM Transactions on Graphics (TOG), vol. 40, no. 5, p. 1–18, 2021.
[27] R. Li, G. Bu, and P. Wang, “An automatic tree skeleton extracting method based on point cloud of terrestrial laser scanner,” International Journal of Optics, vol. 2017, 2017.
[28] Y. Zhang, L. Chen, F. Tan, S. Wang, and B. Yin, “An improved \( L_1 \) median model for extracting 3d human body curve-skeleton,” Multimedia Tools and Applications, vol. 80, no. 24, p. 33547–33571, 2021.
[29] X. Wang, L. Zou, X. Shen, Y. Ren, and Y. Qin, “A region-growing approach for automatic outcrop fracture extraction from a three-dimensional point cloud,” Computers geosciences, vol. 99, p. 100–106, 2017.
[30] H. Siwei and L. Baolong, “Review of bounding box algorithm based on 3d point cloud,” Monitoring and Control (ANMC) Cooperate: Xi’an Technological University (CHINA) West Virginia University (USA) Huddersfield University of UK (UK), p. 18, 2021.
[31] X. Ning, F. Li, G. Tian, and Y. Wang, “An efficient outlier removal method for scattered point cloud data,” PLOS ONE, vol. 13, no. 8, p. e0201280, Aug 2018.
[32] L. Li and W. Wang, “Improved use of lop for curve skeleton extraction,” in Computer Graphics Forum, vol. 37, no. 7. Wiley Online Library, 2018, p. 313–323.
[33] L. Zhu, W. Xing, A. Song, and Y. J. Zhang, “Visibility-driven skeleton extraction from unstructured points,” Computer Aided Geometric Design, vol. 82, p. 101929, 2020.
[34] A. Tagliasacchi, I. Alhashim, M. Olson, and H. Zhang, “Mean curvature skeletons,” in Computer Graphics Forum, vol. 31, no. 5. Wiley Online Library, 2012, p. 1735–1744.
[35] Y. Livny, F. Yan, M. Olson, B. Chen, H. Zhang, and J. El-Sana, Automatic reconstruction of tree skeletal structures from point clouds, 2010, p. 1–8.
[36] L. Fu, J. Liu, J. Zhou, M. Zhang, and Y. Lin, “Tree skeletonization for raw point cloud exploiting cylindrical shape prior,” IEEE Access, vol. 8, p. 27327–27341, 2020.
[37] S. Wu, W. Wen, B. Xiao, X. Guo, J. Du, C. Wang, and Y. Wang, “An accurate skeleton extraction approach from 3d point clouds of maize plants,” Frontiers in plant science, vol. 10, p. 248, 2019.
[38] A. Chaudhury and C. Godin, “Skeletonization of plant point cloud data using stochastic optimization framework,” Frontiers in Plant Science, vol. 11, p. 773, 2020.
[39] A. Tabb and H. Medeiros, “Fast and robust curve skeletonization for real-world elongated objects,” in 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, 2018, p. 1935–1943.
[40] A. C. Jalba, J. Kustra, and A. C. Telea, “Surface and curve skeletonization of large 3d models on the gpu,” IEEE transactions on pattern analysis and machine intelligence, vol. 35, no. 6, p. 1495–1508, 2012.
[41] H. Hu, Z. Li, X. Jin, Z. Deng, M. Chen, and Y. Shen, “Curve skeleton extraction from 3d point clouds through hybrid feature point shifting and clustering,” in Computer Graphics Forum, vol. 39, no. 6. Wiley Online Library, 2020, p. 111–132.
[42] J. Zhou, J. Liu, and M. Zhang, “Curve skeleton extraction via k-nearest-neighbors based contraction,” International Journal of Applied Mathematics and Computer Science, vol. 30, no. 1, 2020.
[43] C. Song, Z. Pang, X. Jing, and C. Xiao, “Distance field guided \( l_1 \)-median skeleton extraction,” The Visual Computer, vol. 34, no. 2, p. 243–255, 2018.
[44] J. Ma, S. W. Bae, and S. Choi, “3d medial axis point approximation using nearest neighbors and the normal field,” The Visual Computer, vol. 28, no. 1, pp. 7–19, 2012.

[45] T. Williams, M. Ganter, and D. Storti, Voxelized Skeletal Modeling Techniques Via Complemental Skeletons, Jun 2020.

[46] F. Usai, M. Livesu, E. Puppo, M. Tarini, and R. Scateni, “Extraction of the quad layout of a triangle mesh guided by its curve skeleton,” ACM Transactions on Graphics, vol. 35, no. 1, pp. 6:1–6:13, 2016.

[47] X. Ning, F. Li, G. Tian, and Y. Wang, “An efficient outlier removal method for scattered point cloud data,” PloS one, vol. 13, no. 8, p. e0201280, 2018.

[48] H. Najdataei, Y. Nikolakopoulos, V. Gulisano, and M. Papatriantafilou, “Continuous and parallel lidar point-cloud clustering,” in 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2018, p. 671–684.

[49] W. H. Jefferys, “On the method of least-squares.” The Astronomical Journal, vol. 85, p. 177, 1980.