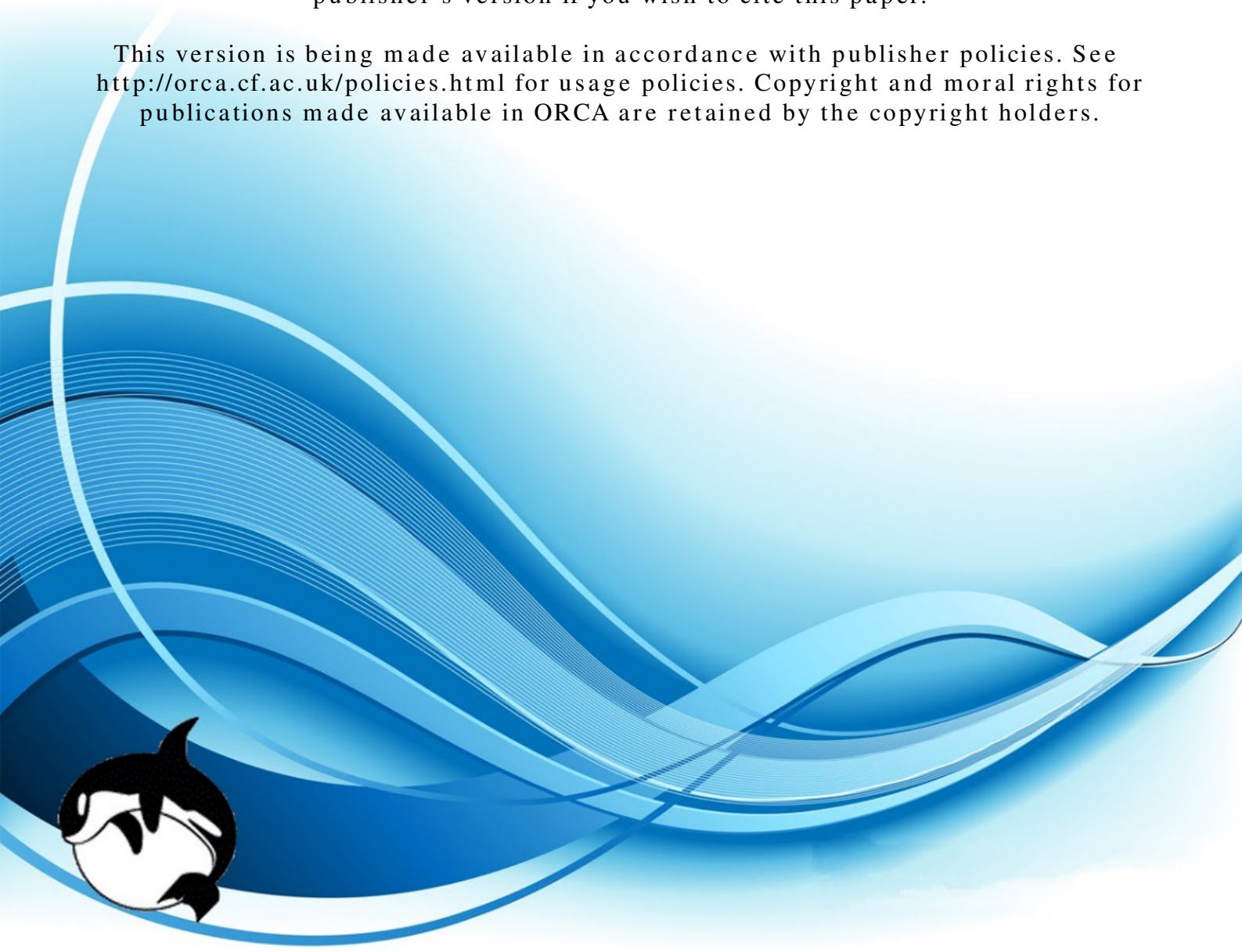


GLSkeleton: A Geometric Laplacian-based Skeletonisation Framework for Object Point Clouds

Qingmeng Wen[1], Seyed Amir Tafrishi[1], Ze Ji[1] and Yu-Kun Lai[2]

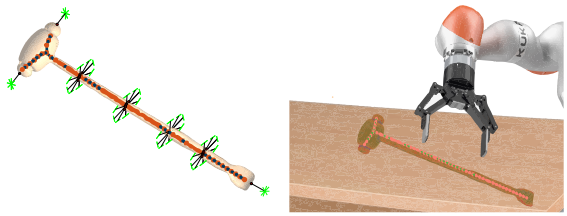
***Abstract*—The curve skeleton is known to geometric modelling and computer graphics communities as one of the shape descriptors which intuitively indicates the topological properties of the objects. In recent years, studies have also suggested the potential of applying curve skeletons to assist robotic reasoning and planning. However, the raw scanned point cloud model is typically incomplete and noisy. Besides, dealing with a large point cloud is also computationally inefficient. Focusing on the curve skeletonisation of incomplete and poorly distributed point clouds of objects, an efficient geometric Laplacian-based skeletonisation framework (GLSkeleton) is proposed in this work. We also present the computational efficiency of the introduced local reduction strategy (LPR) approach without sacrificing the main topological structure. Comprehensive experiments have been conducted to benchmark performance using an open-source dataset, and they have demonstrated a significant improvement in both contraction and overall skeletonisation computational speed.**

***Index Terms*—Computational Geometry; Computer Vision for Automation; Perception for Grasping and Manipulation**

I. Introduction

SKELETONS are frequently associated with vertebrates while this term also refers to a concept that describes the 3D shape geometry features. A study in volunteers conducted by Ayzenberg et al. [1] indicates that skeletal descriptions contribute the most when humans discriminate objects. Generally, the skeleton of a shape should provide an intuitive and effective abstraction and is expected to promote shape understanding and manipulation [2], [3]. The curve skeleton is one of the most notable descriptors of the shape of the skeleton, and its counterpart is medial axis surface [4], [5]. In comparison with the medial axis surface, the curves are more widely accepted for practical applications, attributed to their simple topology, and reduced complexity for manipulation [3], [6]. Skeletonisation of various shapes holds great potential for applications in robotics; however, its computational complexity, sensitivity to noise, and imperfect capture remain open challenges.

Curve skeletonisation algorithms have been extensively researched, but the existing work preferentially focuses on watertight surface models [7]–[13]. In the literature, Chuang et al. [11] presented a curve skeleton extraction method using a generalised potential field. Based on the work of Chuang et al., Cornea et al. [8] discussed a 3D object retrieval method with generated curve skeletons. Au et al. [7] extracted the curve skeleton through Laplacian mesh contraction, followed by a connectivity surgery refinement. Dey et al. [9] gave the definition of curve skeletons of meshes as the subset of the medial axis and designed a medial geodesic function to gain curve skeletons, whereas the medial axis is not well defined. Although most of those existing methods dealing with watertight models can generate satisfactory results, the expected perfect input models are rarely acquired. In fact, it is more common in practice that models are with boundaries and in the form of triangle soups [14]. Besides, those methods typically consider meshes as feeding models for their skeletonisation pipelines while raw and unorganised point clouds, are more accessible in comparison with meshes. However, the raw scanned point clouds cannot provide explicit topology connections like meshes and tend to be incomplete, noisy and often include unavoidable outliers.



(a) Grasping candidates (b) Grasping

Figure 1: Skeleton-assisted grasping planning

Different studies on unorganised point-cloud skeletonisation have explored various strategies for form determination. For example, Tagliasacchi et al. [2] investigated an optimal *cutting plane* for each anchor point to extract the rotational symmetry axis of cylindrical shapes. By adapting local L_1-medial information, Huang et al. [15] introduced L_1-medial skeletons, which is insensitive to noises and demonstrating considerable accuracy of each branch. However, the L_1-medial skeleton extraction method overlooks the interconnections between skeleton points. Besides, the good performance of their method prefers cylindrical shapes [16], [17]. Researchers also explored curve skeletonisation by leveraging the optimal mass transport method to enhance robustness [3]. However, their proposed algorithm is computationally inefficient and exhibits limitations in skeletonisation of sparse point clouds [17]. In recent years, the application of deep learning methods to extract curve skeletons has gained momentum. Nevertheless, existing studies mainly concentrate on generating the medial axis transform of point clouds instead of curve skeletons [17]–[19]. Besides, these learning-based methods might be constrained in applications since they are specifically designed for skeletonising point clouds with particular shapes. One might

Manuscript received: Oct, 3, 2023; Revised Jan, 31, 2024; Accepted Mar, 17, 2024.

This paper was recommended for publication by Editor Bera Aniket upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by China Scholarship Council.

[1]Qingmeng Wen, Seyed Amir Tafrishi and Ze Ji are with School of Engineering, Cardiff University, CF24 3AA, United Kingdom. {wen1, tafrishisa, jiz1}@cardiff.ac.uk

[2]Yu-Kun Lai is with School of Computer Science and Informatics, Cardiff University, CF24 4AG, United Kingdom. laiy4@cardiff.ac.uk

The open-source code will be available at https://github.com/weiqimeng1/GLSkelton.git

explore enabling mesh skeletonisation methods to work on point cloud models. In this way, surface reconstruction is conducted before the models can be operated as meshes. Although reconstructed surface theoretically improves skeletonisation accuracy, issues such as self-intersection may obstruct the reconstruction process. Thus, the existing reconstruction algorithms [20]–[22] can rarely obtain results that satisfy skeletonisation requirements in both computational efficiency and reconstruction quality. Instead of reconstructing the surface, Cao et al. extended the Laplacian-contraction-based curve skeletonisation method of meshes to directly solve point cloud models [7], [14]. In their experiment, the whole skeletonisation process typically takes less than 2 minutes with 10K points, but the computation speed is insufficient for real-time applications, especially when handling models with more points.

In recent years, researchers have been inspired by various potential applications of curve skeletons. Cornea et al. [5] provided a brief introduction to the application of curve skeletons in visualisation. Their findings highlight the broad spectrum of benefits these structures can offer across different domains, including visualisation, image processing, and animation. Wu et al. [23] argued that curve skeletons can assist 3D reconstruction of objects using visual SLAM reconstructed 3D points. Moreover, the advent of 3D skeletonization methods has expanded the scope of applications to include portable and movable devices, unlocking new possibilities for practical implementation.

On a more specific note, recent advancements in skeleton research have demonstrated significant potential in the realm of robotics, particularly in the context of grasping planning. Studies by Pokorny et al. [24] and Stork et al. [25] revealed that the topology feature can be applied to grasping objects with holes. Przybylski et al. [26]–[28] believed that finding a feasible geometric representation that simplifies analysis may result in stable grasp planning. Thus, a union of balls is considered as the medial axis and used to generate candidate grasps. Their grasping experiments have been done both in simulation and in the real world. Also, Vahrenkamp et al. [6] studied the curve-skeleton-based method for planning grasps. As illustrated in Fig. 1, the grasping candidate can be generated with curve skeletons extracted as the topology of the objects. According to experimental results by [6], the skeleton-based grasp planner is more robust and potentially outperforms the surface-based approach. Despite the great potential of curve skeletons for robot grasping, the computational cost and the uncertainty of the physical world are remaining challenges that impede the pace of application.

In this paper, we propose a point reduction strategy within the skeletonisation process using a Laplacian-based contraction method. The proposed approach's contributions are as follows:

- Introducing an in-loop local geometry-based point reduction strategy (LPR) that accelerates the Laplacian-contraction process by eliminating redundant points while preserving the main geometry features of the point cloud in Section II. The strategy also enhances the computation processes with certain robustness for cloud points characterized by non-uniform distributions and noises.
- Proposing a novel terminating checking condition for Laplacian-contraction with in-loop point cloud reduction that exhibits greater robustness to local changes in Section II-C.

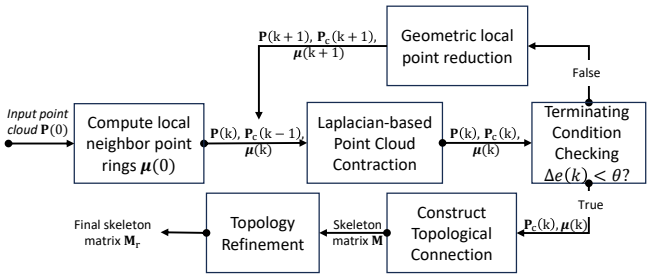


Figure 2: Block diagram of the GLSkeleton framework

- Conducting an extensive evaluation of the geometric method using a real-world OminiObject dataset comprising different objects in Section III.

Finally, we conclude our findings in Section IV.

II. GLSkeleton Framework

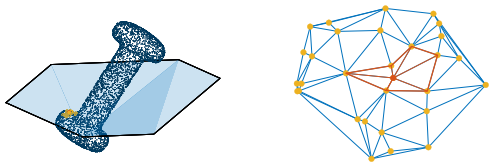
GLSkeleton presents a novel Laplacian-contraction-based skeletonisation framework stemming from the Laplacian smoothing algorithm. The framework comprises four main parts as illustrated in Fig 2. The local point ring is computed prior to the Laplacian contraction process, after which the curve skeleton is extracted through topological connection and refinement. It is noteworthy that the computational demands associated with 3D point cloud data increase significantly alongside point number. To address this, we introduce a geometric local point reduction technique (LPR) within the contraction iterations to eliminate superfluous points from local rings, as detailed in section II-B. Besides, regarding the reduced cloud number, a more robust contraction terminating condition is also introduced for the contraction loop.

A. Overview of Laplacian Skeletonisation

The Laplacian-based skeletonisation method was initially introduced for mesh skeletonisation by Au et al. [7] and was subsequently extended to point clouds by Cao et al. [14]. In order to adapt this method to point clouds, a local point ring strategy is introduced to establish connections among points without requiring explicit topology. Given a point cloud $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n]^T$, $\mathbf{p}_i = [p_{ix}, p_{iy}, p_{iz}]^T \in \mathbb{R}^3$, the local ring of a specific anchor point $\mathbf{p_i}$ is computed by identifying the k nearest neighbour points of $\mathbf{p_i}$ and then projecting them onto the tangent plane. The resulting 2D Delaunay triangulation formed by these neighbouring points, as depicted in Fig. 3 (b), results in the local ring $\boldsymbol{\mu}_i \subseteq \{1, 2, \ldots, n\}$. This point ring sequence consists of point indices corresponding to $\mathbf{P}$ and maintains a clockwise order relative to the tangent plane.

The Laplacian-based contraction method functions as an optimisation system that guides the movement of points by applying and balancing the forces of contraction and attraction. Within this system, the contraction force leads the points to move vertically toward the medial axis surface. In this process, cotangent Laplacian coordinates, known as an estimation of the surface normals, are computed. For the given point cloud $\mathbf{P} \in \mathbb{R}^{n\times 3}$, the cotangent Laplacian coordinates of the points $\boldsymbol{\delta} = [\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, ..., \boldsymbol{\delta}_n]^T$ are defined as

$$\boldsymbol{\delta} = \mathbf{L}\,\mathbf{P}, \tag{1}$$



(a) Projected plane (b) Delaunay Triangulation

Figure 3: For each anchor point (red), both neighbouring points (yellow) and the anchor point itself are projected onto a tangent plane (a). The neighbour point ring, resembling an umbrella-like shape, is hidden within the Delaunay triangulation result processed on this plane (b).

where Laplacian matrix with cotangent weights $\mathbf{L} = [\mathbf{l}_1, \mathbf{l}_2, .., \mathbf{l}_n]^T$ and $\mathbf{l}_i = [l_{i1}, l_{i2}, \ldots, l_{in}]^T$. Then, we compute L_{ij} by

$$L_{ij} = \begin{cases} \omega_{ij} = \cot \alpha_{ij} + \cot \beta_{ij}, & \text{if } j \in \boldsymbol{\mu}_i; \\ \sum_{k \in \boldsymbol{\mu}_i}^{k} -\omega_{ik}, & \text{if } i = j; \\ 0, & \text{Otherwise;} \end{cases} \tag{2}$$

where α_{ij} and β_{ij} are two opposite angles of edge (i, j) in the neighbour ring formed triangles (Fig. 3-b). Please note, $\boldsymbol{\mu}_i$ is the index set of neighbour point ring and edge (i, j) is associated with $(\mathbf{p}_i, \mathbf{p}_j)$, which denote the center point and neighbour point respectively. To prevent excessive contraction that could potentially disrupt the model's structure and achieve balance, the attraction component is also incorporated into this optimisation system. In this aspect, the deviation of points from their prior positions is taken into account and considered as the attraction force that pulls them back from too much contraction. By incorporating both contraction and attraction components in this system, the equations for Laplacian-based contraction are defined by

$$\begin{bmatrix} \mathbf{W}_L \mathbf{L} \\ \mathbf{W}_H \end{bmatrix} \mathbf{P}_c(k+1) = \begin{bmatrix} \mathbf{0} \\ \mathbf{W}_H \mathbf{P}_c(k) \end{bmatrix} \tag{3}$$

where $\mathbf{P}_c(k) \in \mathbb{R}^{n \times 3}$ and $\mathbf{P}_c(k+1) \in \mathbb{R}^{n \times 3}$ denotes input and resultant cloud points of the k-th contraction, $\mathbf{W}_H$ and $\mathbf{W}_L$ are $n \times n$ diagonal weight matrices that control the attraction and contraction force respectively.

By solving Eq. (3), the contraction process then turns to minimising the quadratic energy as follows [7]:

$$\|\mathbf{W}_L \mathbf{L} \mathbf{P}_c(k)\|^2 + \sum_i \mathbf{W}_{H,i}^2 \|\mathbf{p}_{c,i}(k+1) - \mathbf{p}_{c,i}(k)\|^2 \tag{4}$$

where the two parts of the equation take the contraction constraints and the attraction constraints, respectively. The Laplacian contraction is an iterative process. From beginning, $\mathbf{W}_H$ and $\mathbf{W}_L$ are initialised as certain values [14]. But after each iteration, their values should be updated to continue the contraction iterations. The weights can be updated after k-th iteration by the following equations,

$$\begin{bmatrix} \mathbf{W}_L(k+1) \\ \mathbf{W}_H(k+1) \end{bmatrix} = \begin{bmatrix} s_L \mathbf{W}_L(k) \\ \mathbf{W}_H(k) \left(\mathbf{S}(0)/\mathbf{S}(k)\right) \end{bmatrix}, \tag{5}$$

where s_L is a scalar, $\mathbf{S}(0) = [s_1(0), s_2(0), \ldots, s_n(0)]^T$ and $\mathbf{S}(k) = [s_1(k), s_2(k), \ldots, s_n(k)]^T$ comprises the closest distances to the neighbours of each point of current and original models respectively.

As illustrated in Fig. 2, within contraction loops, the cloud points are cautiously removed by geometric local point reduction strategy (LPR) as discussed in Section II-B, and the contraction loop will break if the contracted cloud point satisfies the contraction terminating condition, as explained in Section II-C. However, there are still no topological connections between points after Laplacian contraction. Thus, topology connections are conducted, followed by topology refinement, upon the contracted points to obtain the final skeletons.

B. Geometric Local Point Reduction Strategy

To accelerate Laplacian computation, we introduce a new geometric method called local point reduction (LPR). As shown in Algorithm 1 and Fig. 2, after each contraction loop, the resultant point cloud $\mathbf{P}_c$ is fed to a local point reduction procedure. This procedure begins by selecting and removing points based on their point-wise contraction stage function values. After the removal of the selected points, the neighbouring rings $\boldsymbol{\mu}$ are reconstructed through a ring combination process, as the neighbour rings are disrupted by the removal of ring points. In addition, the original point cloud $\mathbf{P}$ is updated within the process to maintain a one-to-one correspondence with the contracted point cloud.

In the Laplacian-based contraction for skeletonisation, the computation slowdown primarily stems from the size of the point cloud data. This occurs consistently throughout the skeletonisation process, as a significant number of points become progressively redundant during contraction operations. This redundancy arises from increased local point cloud density and gradual abstraction of the main structure as the point cloud is compressed. Thus, retaining all points might be unnecessary for subsequent procedures. As a solution, LPR is devised to eliminate points that become unnecessary after each contraction iteration.

The primary concept of LPR is to release the abnormal constraints that limit the contraction speed by removing selected points in the point ring. Considering Eqs. (4)-(5), the contraction weights $\mathbf{W}_L$ and attraction weights $\mathbf{W}_H$ restrict the contraction process within iterations. In fact, during every contraction iteration, the weights should be updated before the point cloud can be contracted. The reason is that when the last iteration is finished, the system will reach a balance, and the Laplacian weights become minimal. In Eq. (4), the contraction energy term decreases, and, in contrast, the attraction energy term increases while the cloud points are getting contracted. Thus, if the model is a convex point set, the system which reaches the balance should satisfy

$$\|\mathbf{W}_L \mathbf{L} \mathbf{P}_c(k)\|^2 = \sum_i \mathbf{W}_{H,i}^2 \|\mathbf{p}_{c,i}(k+1) - \mathbf{p}_{c,i}(k)\|^2 \tag{6}$$

Focusing on each point and substituting Eq. (2), we propose a point-wise stage function via the quadratic energy of contraction and attraction by

$$\begin{aligned} &V(\mathbf{p}_{c,i}, k+1) = w_l^2(k+1) \|\boldsymbol{\delta}_i(k+1)\|^2 \\ &- w_{h,i}(k+1)^2 \|\Delta \mathbf{p}_{c,i}(k+1)\|^2 = s_L^{k+2} w_l^2(0) \|\boldsymbol{\delta}_i(k+1)\|^2 \\ &- \left(\frac{s_i(0)}{s_i(k+1)} w_{h,0}\right)^2 \|\Delta \mathbf{p}_{c,i}(k+1)\|^2, \end{aligned} \tag{7}$$

where $\Delta \mathbf{p}_{c,i}(k+1) = \mathbf{p}_{c,i}(k+1) - \mathbf{p}_{c,i}(k)$, $w_{h,i}(k)$ and $w_l(k)$ are point cloud rate of change, point-wise weights

corresponding to $\mathbf{W}_H(k)$ and $\mathbf{W}_L(k)$ in Eq. (5), respectively. With the stage function and the given predicted point cloud, the contraction stage can be tested. If the points are right at the balance stage of the next iteration, the function will give a 0 value. But if the points go too much inward, the function will give a positive value, and vice versa. Let ψ denote the average neighbour ring size reduction rate in the k-th iteration, $\psi \in [0, 1]$, we assume

$$s_i(k+1) = s_i(k) - \psi' s_i(0), \tag{8}$$

where ψ' is the expected reduction rate of the neighbour ring size for the $k+1$-th iteration, $\psi' = \varepsilon\psi$. Then we have

$$\boldsymbol{\delta}_i(k+1) = \frac{s_i(k+1)}{s_i(0)}\boldsymbol{\delta}_i(k) = \frac{s_i(k) - \psi' s_i(0)}{s_i(0)}\boldsymbol{\delta}_i(k), \tag{9}$$

$$\begin{aligned}\Delta\mathbf{p}_{c,i}(k+1) &= [\mathbf{p}_{c,i}(0) - \mathbf{p}_{c,i}(k)] - [\mathbf{p}_{c,i}(0) \\ &- \mathbf{p}_{c,i}(k+1)] = [\mathbf{p}_{c,i}(0) - \mathbf{p}_{c,i}(k)] - \frac{s_i(0) - s_i(k+1)}{s_i(0) - s_i(k)} \\ &\cdot [\mathbf{p}_{c,i}(0) - \mathbf{p}_{c,i}(k)] = \frac{\psi' s_i(0)}{s_i(0) - s_i(k)}(\mathbf{p}_{c,i}(k) - \mathbf{p}_{c,i}(0)),\end{aligned} \tag{10}$$

By utilising the original stage function (7), the points in the point cloud that impede the contraction pace can be diagnosed, and considered to be removed. Besides, to preserve the sharp features on shapes, where the points usually appear as significant curvature differences with respect to their neighbours, the curvature difference term is also considered in our LPR. Substituting Eqs. (8)-(10) and adding the curvature protecting term $U(k)$, the function is finalised to

$$\begin{aligned}F(\mathbf{p}_{c,i}(k+1)) &= \omega_1 V_1(k+1) + \omega_2 V_2(k+1) + \omega_3 U(k) \\ V_1(k+1) &= s_L^{k+2} w_l^2(0) \|\eta\boldsymbol{\delta}_i(k)\|^2 \\ V_2(k+1) &= \left(\frac{w_{h,i}(k)}{\eta}\right)^2 \|\frac{\psi' s_i(0)}{s_i(0) - s_i(k)}(\mathbf{p}_{c,i}(k) - \mathbf{p}_{c,i}(0))\|^2 \\ U(k) &= \|\boldsymbol{\delta}_i(k) - \frac{1}{n_i}\sum_{j\in\boldsymbol{\mu}_i}\boldsymbol{\delta}_j(k)\|^2\end{aligned} \tag{11}$$

where $\eta = [s_i(k) - \psi' s_i(0)]/s_i(0)$, $\omega_1 = 1$, $\omega_2 = -1$, $\omega_3 = 3$, $\boldsymbol{\delta}_i(k)$ is the Laplacian coordinates worked as estimation of $k_{n,i} \cdot \boldsymbol{n}$ and n_i is the points number of ring neighbours respectively. Please note that the gains (ω_1, ω_2) are selected for satisfying stable convergence of contraction with convex set to reach (6), and ω_3 is determined by which the sharp details can be preserved. As illustrated in Algorithm 1, the points which meet the inequality

$$F(\mathbf{p}_{c,i}, k+1) < \phi s_i^2(k) \tag{12}$$

where ϕ is the scaling coefficient, are decided to be removed its closest neighbours after each contraction iteration.

C. Terminating Condition for Contraction Iterations

Apart from computational costs, determining when to terminate contraction iterations poses another challenge with point reduction. In the study by Cao et al. [14], a threshold based on the difference in ring sizes is used to decide when to stop the contraction process. However, if point reduction is applied, this value may not be able to indicate the contraction stage. Hence, in this research, the change in the explained score, which serves as an indicator of the global contraction rate, is adopted as the criterion to break the loop of the contraction process.

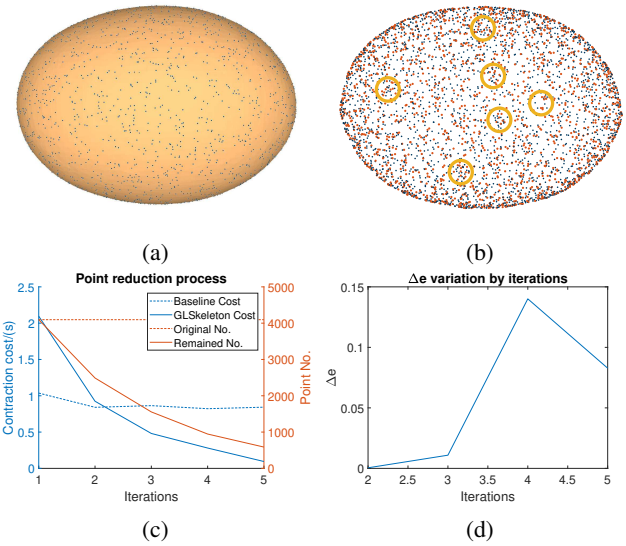


Figure 4: Illustration of LPR. The original model of an egg (a) and the model after one iteration of contraction (b), where locally improved parts are highlighted in yellow circles. The red points are removed, and the blue points are the remaining points. (c) and (d) depict the point numbers and contraction costs, variation of differences of the explained score within corresponding contraction iterations, respectively.

The PCA method is applied to describe the explained score for the contracted point cloud. Given a k point set $\{\mathbf{x}_k\}, \mathbf{x}_k \in \mathbb{R}^3$, let $\mathbf{m}$ denote the centre of the point set, the 3×3 covariance matrix of the point set is computed as follows,

$$\mathbf{CV} = \sum_k (\mathbf{x}_k - \mathbf{m}) \otimes (\mathbf{x}_k - \mathbf{m}), \tag{13}$$

where $\otimes$ is the outer product vector operator. Let $\lambda_{i,1}, \lambda_{i,2}, \lambda_{i,3}$ denote the eigenvalues of the covariance matrix $\mathbf{CV}_{\boldsymbol{\mu}_i}$, where

Algorithm 1 The computation of LPR of GLSkeleton.

procedure LPR($\mathbf{P}(k), \mathbf{P}_c(k), \boldsymbol{\mu}(k)$)
  for Each point $\mathbf{p}_{c,i}$ in $\mathbf{P}_c(k)$ and $\boldsymbol{\mu}_i$ in $\boldsymbol{\mu}(k)$ **do**
    compute stage function value $F(\mathbf{p}_{c,i})$ by Eq. (11)
    if the stage function value meet the inequality (12) **then**
      $m = \text{argmin}_{j\in1,2,\ldots,k_i} \|\mathbf{P}_{c,i} - \mathbf{P}_{c,\boldsymbol{\mu}_i(j)}\|$
      Remove $p_{\mu_i(m)}$ and $p_{c,\mu_i(m)}$ from point cloud $\mathbf{P}(k)$ and $\mathbf{P}_c(k)$ respectively
      Compute new μ_i by combining ring μ_i and μ_m with estimated surface normals (1)
      Remove ring $\boldsymbol{\mu}_m$ from $\boldsymbol{\mu}(k)$
      Update all rings in $\boldsymbol{\mu}(k)$
    end if
  end for
  $\{\mathbf{P}(k+1), \mathbf{P_c}(k+1), \boldsymbol{\mu}(k+1)\} \leftarrow \{\mathbf{P}(k), \mathbf{P}_c(k), \boldsymbol{\mu}(k)\}$
return $\mathbf{P}(k+1)$, $\mathbf{P_c}(k+1)$ and $\boldsymbol{\mu}(k+1)$
end procedure

$\lambda_{i,1} \geq \lambda_{i,2} \geq \lambda_{i,3}$. And denote the corresponding unit eigenvectors as $\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \mathbf{v}_{i,3}$. Each element λ_i is a principal component value corresponding to the directions of vector $\mathbf{v}_i$.

Consider the whole contracted model $\mathbf{P}_c = \{\mathbf{P}_{c,i}\}$ as a point set in Eq. (13) and barycenter as the centre. Similarly, we can get covariance matrix $\mathbf{CV}_{\mathbf{P}_c}$. Then the explained score $\mathbf{e}$ of $\mathbf{P}_c$ on each projected plane is defined as

$$\begin{aligned}\mathbf{e} &= \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \\ &= \frac{1}{2(\lambda_1+\lambda_2+\lambda_3)} \begin{bmatrix} \lambda_1+\lambda_2 & \lambda_1+\lambda_3 & \lambda_2+\lambda_3 \end{bmatrix}^T,\end{aligned} \tag{14}$$

where $\lambda_1, \lambda_2, \lambda_3$ are the associated eigenvalues of $\mathbf{CV}_{\mathbf{P}_c}$ and satisfy $\lambda_1 \geq \lambda_2 \geq \lambda_3$. We choose the values of the most explained two projected planes by

$$e = \frac{1}{2}(e_1 + e_2) = \frac{2\lambda_1 + \lambda_2 + \lambda_3}{2(\lambda_1+\lambda_2+\lambda_3)}, \tag{15}$$

where $e \in [0, 1]$. Since the total explained scores on those two planes keep increasing, we can get the difference of explained scores to terminate the contraction at a satisfied stage by

$$\Delta e(k) = \frac{|e(k+1) - e(k)|}{1 - (n(k) - n(k+1))/n(0)}, \tag{16}$$

where the point number differences $(n(k) - n(k+1))$ are also accounted for to eliminate point reduction effect. During the contraction iterations, if $\Delta e(k)$ is smaller than a threshold $\theta \in [0, 1]$, then the contraction loop will break.

III. Results & Discussion

In this section, we demonstrate the performance of the GLSkeleton by testing on OminiObject3D dataset [29]. OminiObject3D is a real-scanned 3D object dataset containing a wide range of object categories, of which the objects are highly diverse in shape and appearance. Apart from that, it is worth saying that as a real-scanned dataset, the resulting point cloud models usually have noises, and the points are unevenly distributed. Please note that the algorithm in this work is coded and tested in MATLAB on a PC with an Intel(R) Core(TM) i7-3770K CPU.

A. Contraction Speed Comparison

In comparison with Cao et al. [14], the in-loop local point reduction method mainly benefits the contraction speed within iterations. We consider the method of [14] as baseline and conduct experiments with both methods. Within the experiments, we set the parameters as Table I.

To make the comparison, 216 experimental point cloud models, each with 3 different resolutions are selected from the dataset without particular preference on object of interest. During the contraction iterations, the unnecessary points that impede the contraction process are successfully removed, illustrated in Fig. 4-a. The time complexity for solving the contraction system (4) is $O(n)$, where n is the point number of the model. Thus, the point reduction can linearly speed up the contraction process. The result in Fig. 4-c proves this point. Meanwhile, the proposed difference of explained score reaches a peak and then decreases with iterations as illustrated in Fig. 4-d, which means that Δe is in line with the global contraction rates variance. In other words, it is logical to use Δe as an indicator to terminate the contraction iteration.

As depicted in Fig. 5, when dealing with small-scaled data (point cloud with 1024 points), the GLSkeleton remains at the same performance level as the baseline despite the additional computation cost for point reduction. However, as the point

Table I: Parameter settings for GLSkeleton

Parameters	Value	Descriptions
ε	1.2	ψ'/ψ
ϕ	1e5	Parameter of Eq. (12)
θ	0.02	Contraction termination threshold of Δe

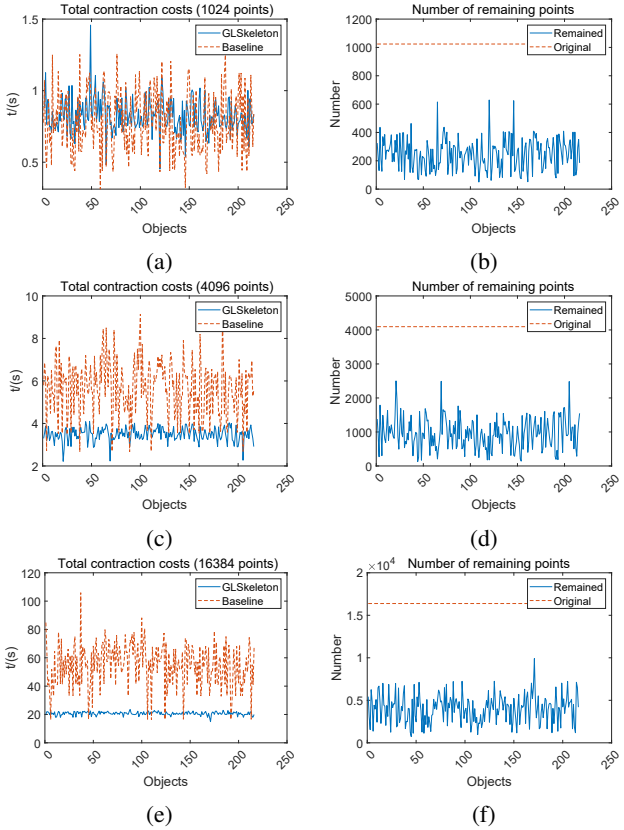
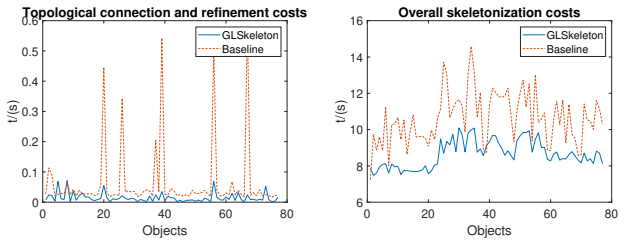


Figure 5: Contraction cost comparison between GLSkeleton (blue) and the baseline (red). From top to bottom, the left column shows the contraction costs of 216 object models with resolutions of 1024, 4096, and 16384 points, respectively. The right column depicts the corresponding remaining points after the contraction process.



(a) Curve computation cost (b) Overall computation cost

Figure 6: Curve computation cost from contracted point cloud (a) and Overall computation cost (b).

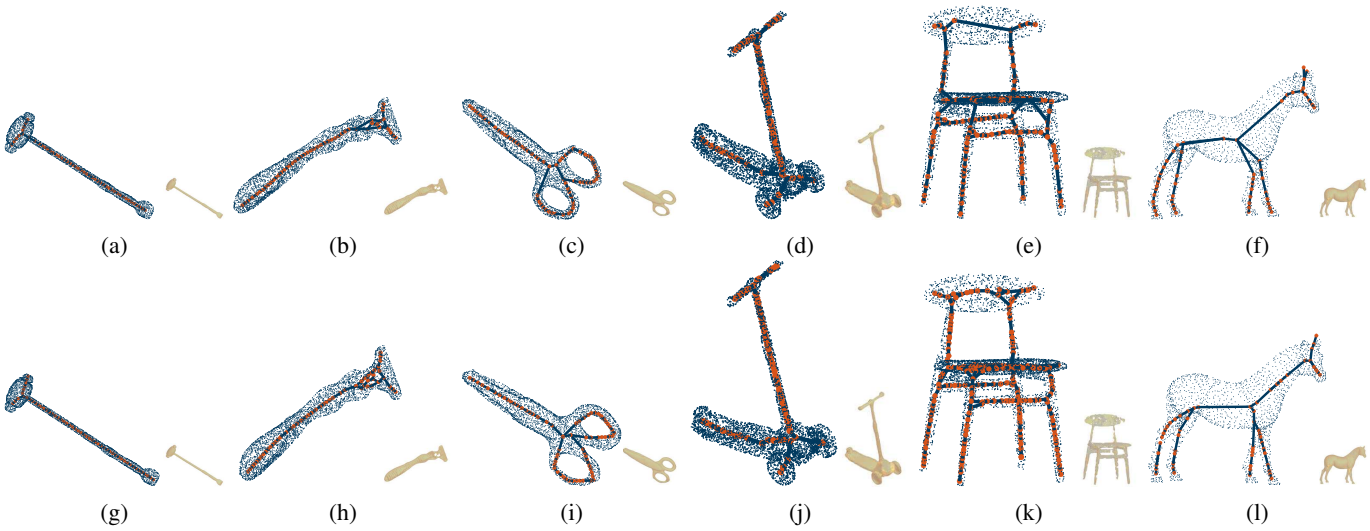


Figure 7: Skeletonisation result comparison. (a-f) and (g-l) are skeletonisation results of challenging point cloud models of the baseline (upper row) and GLSkeleton (lower row).

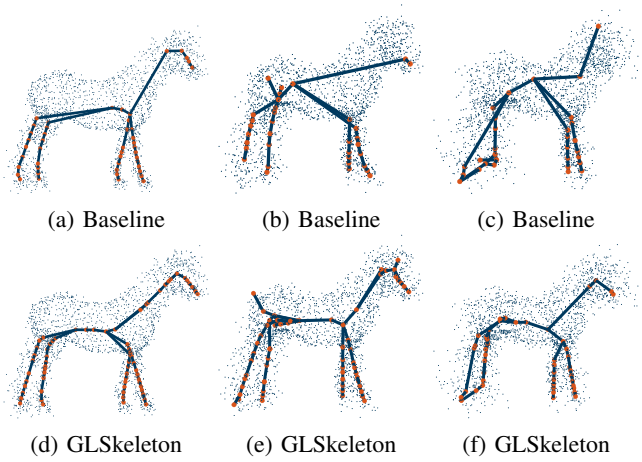


Figure 8: Skeletonisation results of noised point clouds. From the left to the right, 5%, 10% and 20% value of Gaussian spread noises are added to every point of original input point cloud.

number increases, GLSkeleton works faster than the baseline since the point number starts dominantly impacting the contraction cost. For models with 4096 points, the contraction cost is nearly halved, while in the case of models with 16384 points, the costs are further reduced. Though GLSkeleton does not show significant differences while dealing with light point cloud data (Fig. 5-a), it is still inspiring that the computational costs of different objects are less varied (the standard deviation of baseline and GLSkeleton costs are 1.2996 and 0.3205 respectively). In other words, the contraction speed is more controllable with GLSkeleton.

B. Skeletonisation Results

To further compare GLSkeleton with the baseline, complete skeletonisation experiments are conducted for selected 77 object models (4096 points) with logical patterns of 3D skeletons, while other objects are mainly composed of sphere shapes or 2D sheets. Please be aware that among the 216 objects' skeletonization patterns, some exhibit a substantial number of similarities. Despite this, there are persistent challenges where current methods struggle to establish coherent skeleton patterns (including baseline method). This difficulty ultimately influenced the decision to opt for a smaller set of object models for this part. We think those objects will give a clear comparison of the performance of the skeletonisation.

According to the results, the local cloud point reduction also remarkably benefits the curve computation process, as shown in Fig. 6-a. The main reason is that the farthest sphere sampling working for the topological connection [14] uses a kd-tree algorithm for ball range query, and the time complexity of the kd-tree range query is $O(n^{\frac{2}{3}})$. Thus, the overall computational cost of skeletonisation by GLSkeleton is reasonably faster than that by the baseline (Fig. 6-b). Please note that it is normal that the difference of computational cost in Fig. 6-b appears less significant than in Fig. 5-c. That is because GLSkeleton and the baseline method share the same neighbour ring computation process before Laplacian contraction, as depicted in Fig. 2.

According to the generated skeletons (Fig. 7), GLSkeleton mainly preserves the topological features, despite some minimal differences in comparison with the baseline (Figs. 7-a,d,g,j). Even when dealing with the models with missing data (Figs. 7-e,k), the curve structure can be plausibly extracted. However, since points are removed during contraction iterations of GLSkeleton, certain features are inevitably faded or biased, potentially resulting in non-medial skeletal branches. This phenomenon is more likely to occur when dealing with joint and loop shapes in sparse point sets, due to the enlarged size of neighbour point rings (Figs. 7-g, h, i, l). Fortunately, the effect of this issue is negligible in most cases. Illustrated in Fig. 8, experimental results of handling noisy data are shown. In the experiment, each point of the input point cloud is subjected to varying levels of Gaussian noise. Even though extreme noises are applied, GLSkeleton consistently produces acceptable results compared to no-noise conditions,

occasionally surpassing the performance of the baseline but with a lighter computation load.

Fig. 7 shows the graphical presentation of skeletonisation for some example objects' point clouds (the horse model is with 1987 points, the rests are with 4096 points) from real scanning. The resultant skeletons of GLSkeleton are able to provide clear and meaningful skeleton vertices with corresponding topological vertex connections that can be used for grasping planning [6]. The well-organised skeleton vertices can be applied for model segmentation and to compute curvatures at skeleton points, which are critical for local geometrical property analysis through applying specific grasping strategies. However, there are certain challenges to applying GLSkeleton in real-time for grasping planning. For instance, vertices of the skeleton generated by GLSkeleton are non-medial in some cases, potentially affecting segmentation of point cloud model. Another challenge could be that some strictly convex objects, such as spheres, do not have meaningful skeleton patterns.

IV. Conclusion

In this work, a geometric Laplacian-based skeletonisation framework is proposed, which can gain similar skeletonisation results with less computational cost. In this framework, the local point reduction strategy judges and removes the redundant points that impede the contraction speed, which significantly improves the overall skeletonisation speed in comparison with baselines. In addition, a more robust global contraction rate indicator, which is independent of local point distributions, is also presented to adapt to the point-reduction-in-loop contraction scenario. Finally, the computational performance is evaluated by the real-scanned dataset. Although GLSkeleton can successfully extract skeletons from the majority of unorganised point cloud models, some issues still remains. Firstly, there are remaining limitations with the Laplacian-based method [14]. For example, the accuracy of topology of the resultant skeleton might lose since a constant value of radius are chosen for farthest-point sampling, leading to unstable performance. Second, there is a gap in the evaluation of the skeletonisation results, which is crucial for addressing applications.

In the future, further work will focus on improving the robustness of the skeletonisation and publishing our dataset. We may utilize potential motion planning strategies for creating skeleton patterns from our reduced cloud point model rather than using topological connections and refinement strategy. We will also focus on study of grasp planning with the proposed skeletonisation approach, including the reference skeleton quality on grasping and geometric relation of grasping (contact point) with skeleton and object's contacted surface (local mesh).

References

[1] V. Ayzenberg and S. F. Lourenco, "Skeletal descriptions of shape provide unique perceptual information for object recognition," *Scientific Reports*, vol. 9, no. 1, p. 9359, Jun. 2019.

[2] A. Tagliasacchi, H. Zhang, and D. Cohen-Or, "Curve skeleton extraction from incomplete point cloud," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–9, Jul. 2009.

[3] H. Qin, J. Han, N. Li, H. Huang, and B. Chen, "Mass-driven topology-aware curve skeleton extraction from incomplete point clouds," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 9, pp. 2805–2817, 2020.

[4] H. Blum, *A Transformation for Extracting New Descriptors of Shape*. M.I.T. Press, 1967.

[5] N. D. Cornea, D. Silver, and P. Min, "Curve-Skeleton Properties, Applications, and Algorithms," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 3, pp. 530–548, May 2007.

[6] N. Vahrenkamp, E. Koch, M. Waechter, and T. Asfour, "Planning high-quality grasps using mean curvature object skeletons," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 911–918, 2018.

[7] O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee, "Skeleton extraction by mesh contraction," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–10, 2008.

[8] N. Cornea, M. Demirci, D. Silver, Shokoufandeh, S. Dickinson, and P. Kantor, "3d object retrieval using many-to-many matching of curve skeletons," in *International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, 2005, pp. 366–371.

[9] T. K. Dey and J. Sun, "Defining and computing curve-skeletons with medial geodesic function," in *Symposium on geometry processing*, vol. 6, 2006, pp. 143–152.

[10] I. Baran and J. Popović, "Automatic rigging and animation of 3d characters," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 72–es, 2007.

[11] J.-H. Chuang, N. Ahuja, C.-C. Lin, C.-H. Tsai, and C.-H. Chen, "A potential-based generalized cylinder representation," *Computers & Graphics*, vol. 28, no. 6, pp. 907–918, 2004.

[12] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, and A. Telea, "3d skeletons: A state-of-the-art report," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 573–597.

[13] Z. Xu, Y. Zhou, E. Kalogerakis, C. Landreth, and K. Singh, "Rignet: neural rigging for articulated characters," *ACM Trans. Graph.*, vol. 39, no. 4, aug 2020.

[14] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su, "Point Cloud Skeletons via Laplacian Based Contraction," in *2010 Shape Modeling International Conference*, Jun. 2010, pp. 187–197.

[15] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen, "L1-medial skeleton of point cloud," *ACM Trans. Graph.*, vol. 32, no. 4, jul 2013.

[16] 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.

[17] M. Ge, J. Yao, B. Yang, N. Wang, Z. Chen, and X. Guo, "Point2mm: Learning medial mesh from point clouds," *Computers & Graphics*, vol. 115, pp. 511–521, 2023.

[18] Y. Pan, B. Wang, X. Guo, H. Zeng, Y. Ma, and W. Wang, "Q-mat+: An error-controllable and feature-sensitive simplification algorithm for medial axis transform," *Computer Aided Geometric Design*, vol. 71, pp. 16–29, 2019.

[19] 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 (CVPR)*, June 2021, pp. 4277–4286.

[20] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, ser. SGP '06. Goslar, DEU: Eurographics Association, 2006, p. 61–70.

[21] C. M. Jiang, A. Sud, A. Makadia, J. Huang, M. Niessner, and T. Funkhouser, "Local implicit grid representations for 3d scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[22] S. Sellán and A. Jacobson, "Stochastic poisson surface reconstruction," *ACM Trans. Graph.*, vol. 41, no. 6, nov 2022.

[23] L. Wu, R. Falque, V. Perez-Puchalt, L. Liu, N. Pietroni, and T. Vidal-Calleja, "Skeleton-based conditionally independent gaussian process implicit surfaces for fusion in sparse to dense 3d reconstruction," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1532–1539, 2020.

[24] F. T. Pokorny, J. A. Stork, and D. Kragic, "Grasping objects with holes: A topological approach," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 1100–1107.

[25] J. A. Stork, F. T. Pokorny, and D. Kragic, "A topology-based object representation for clasping, latching and hooking," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2013, pp. 138–145.

[26] M. Przybylski, T. Asfour, and R. Dillmann, "Unions of balls for shape approximation in robot grasping," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 1592–1599.

[27] ——, "Planning grasps for robotic hands using a novel object representation based on the medial axis transform," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 1781–1788.

[28] M. Przybylski, M. Wächter, T. Asfour, and R. Dillmann, "A skeleton-based approach to grasp known objects with a humanoid robot," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 2012, pp. 376–383.

[29] T. Wu, J. Zhang, X. Fu, Y. Wang, J. Ren, L. Pan, W. Wu, L. Yang, J. Wang, C. Qian, D. Lin, and Z. Liu, "Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 803–814.