Skeleton Extraction by Mesh Contraction

Oscar Kin-Chung Au* Chiew-Lan Tai* Hung-Kuo Chu† Daniel Cohen-Or‡ Tong-Yee Lee†
*The Hong Kong Univ. of Science and Technology †National Cheng Kung University ‡Tel Aviv University

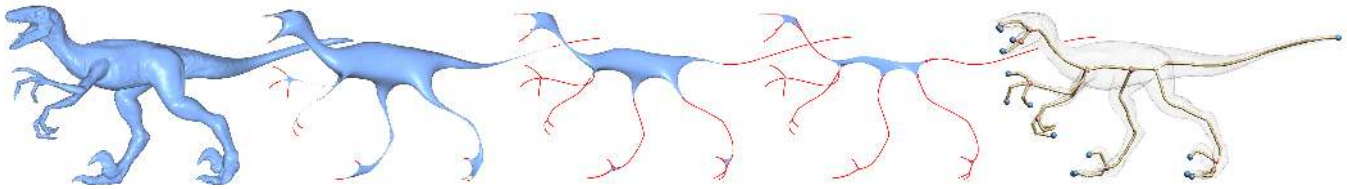

Figure 1: Our method extracts a 1D skeletal shape by performing geometry contraction using constrained Laplacian smoothing. Left to right are the original mesh and the results of the contraction after 1, 2 and 3 iterations. Faces with zero area are drawn in red. The rightmost ray-traced image shows the final skeleton after performing connectivity surgery and embedding refinement.

Abstract

Extraction of curve-skeletons is a fundamental problem with many applications in computer graphics and visualization. In this paper, we present a simple and robust skeleton extraction method based on mesh contraction. The method works directly on the mesh domain, without pre-sampling the mesh model into a volumetric representation. The method first contracts the mesh geometry into a zero-volume skeletal shape by applying implicit Laplacian smoothing with global positional constraints. The contraction does not alter the mesh connectivity and retains the key features of the original mesh. The contracted mesh is then converted into a 1D curve-skeleton through a connectivity surgery process to remove all the collapsed faces while preserving the shape of the contracted mesh and the original topology. The centeredness of the skeleton is refined by exploiting the induced skeleton-mesh mapping. In addition to producing a curve skeleton, the method generates other valuable information about the object's geometry, in particular, the skeleton-vertex correspondence and the local thickness, which are useful for various applications. We demonstrate its effectiveness in mesh segmentation and skinning animation.

Keywords: Skeleton, mesh contraction, Laplacian smoothing, segmentation, skinning

1 Introduction

Curve-skeletons are 1D structures that represent a simplified version of the geometry and topology of a 3D object. They are useful in many applications that require an analysis of the shape, such as animation, morphing, shape registration, and shape retrieval. Therefore, the extraction of curve-skeletons from 3D models is a fundamental problem in computer graphics and visualization. The problem has received a lot of attention in recent decades and yet the design of a simple and robust method for extracting curve-skeletons remains a research challenge [Cornea et al. 2007].

A curve-skeleton is essentially a geometry entity that abstracts the object's volume. Therefore, most existing skeleton extraction methods require a volumetric discrete representation of the input model. However, many models used in computer graphic applications are available only as surface representations, such as polygonal meshes. Transforming them into volumetric representations may raise discretization error in both geometry and connectivity, depending on the grid resolution used in the resampling process.

In this paper, we introduce a skeleton extraction technique that is applied in the object space, directly on the mesh representation. The surface of the given object is iteratively smoothed and contracted into an approximate zero-volume degenerate mesh that abstracts the given shape and topology well (see Figure 1). During this contraction process, the original mesh connectivity is not altered. The challenge of this geometry contraction is to control the contraction process carefully so that it leads to a collapsed shape that approximates the original geometry. We formulate the contraction as an energy minimization problem involving two terms: a contraction term based on the discrete Laplace operator that removes the geometry details along the approximate normal directions, and an attraction term that uses the mesh vertices as anchors to retain necessary geometry information in the collapsing shape. We balance these two energy terms during the iterations by carefully designing their weight functions such that the mesh is automatically contracted into a skeletal shape within only a few iterations. This geometry contraction is a global implicit smoothing process, thus it is robust and noise insensitive.

To convert the zero-volume mesh into a 1D curve skeleton, we perform a connectivity surgery process to remove all the collapsed faces from the degenerate mesh through a sequence of edge-collapse operations. The main requirement is to perform this simplification with as little disturbance as possible to the shape of the contracted mesh. A secondary requirement is to retain sufficient sample points so as to maintain a fine correspondence between the curve-skeleton and the original mesh. We devise an energy function that balances the shape and sampling requirements. The edge collapses never disconnect the mesh and we prohibit edge collapses that close tunnels, thus the extracted curve-skeleton is guaranteed to be homotopic to the original mesh.

The geometry contraction process does not ensure the centeredness of the contracted 1D shape. However, thanks to the skeleton-mesh mapping obtained through recording the edge collapses during the connectivity surgery, we can design a simple method to refine the skeleton's embedding by moving each skeletal node to the center of its corresponding mesh region.

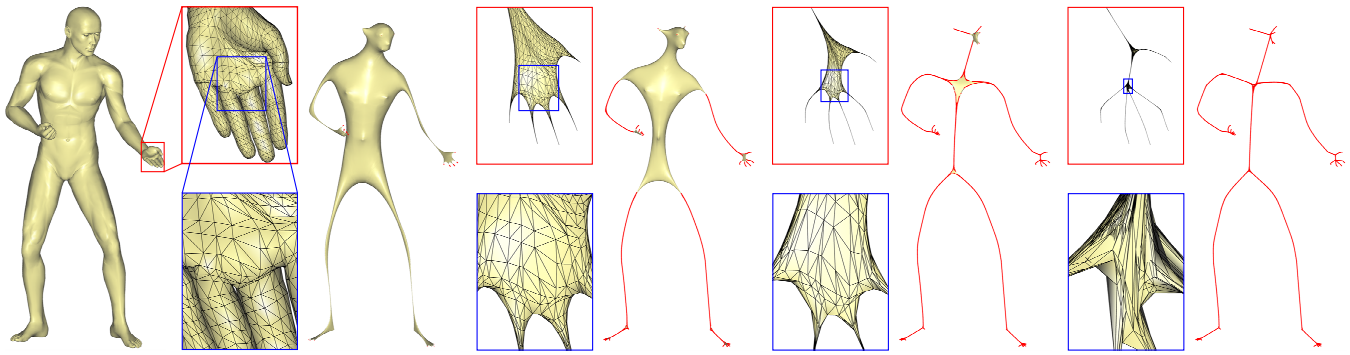

Figure 2: Contracting a male model. Results after 1, 2, 4, 6 iterations. The red frames show zoom-in views of the hand. The blue frames show an incremental zoom-in of part of the red frames.

Our skeleton extraction method generates useful geometry information, including skeleton-mesh mapping, the local thickness and volume of the object. We show that this non-trivial information is useful for segmentation and skinning.

2 Background

The literature contains extensive research on curve-skeleton extraction. In the following we review only some representative methods and refer the reader to the comprehensive survey of Cornea et al. [2007].

Curve-skeletons do not have a rigorous definition. Various applications may have different requirements on certain properties; for example, virtual navigation for medical applications has a stricter requirement on the centeredness than animation. Curve-skeletons are closely related to the medial axis. The medial axis of a 3D object is defined as the locus of the centers of all inscribed spheres of maximal radius, and it often contains surface elements. This definition implies that the medial axis is inherently sensitive to perturbation in the object's boundary. An additional non-trivial pruning process is therefore needed to handle boundary noise and to extract a 1D curve-skeleton from the medial surface [Amenta et al. 2001; Dey and Sun 2006].

Another related structure is bone-skeleton, which is used extensively in character animation [Baran and Popović 2007; Wang et al. 2007] and mesh deformation [Shi et al. 2007; Yoshizawa et al. 2007; Weber et al.]. Such bone-skeletons can be easily derived from curve-skeletons by down-sampling. We focus on curve-skeletons in this paper, but show examples of down-sampled bone-skeletons for skinning in Section 8.2.

Methods for curve-skeleton extraction can be classified into two main categories, volumetric and geometric, depending on whether an interior representation or only the surface representation is used.

Volumetric methods. Most existing curve-skeleton extraction methods make use of a volumetric discrete representation, either a regularly partitioned voxelized representation or a discretized field function defined in the 3D space. These methods share the common drawbacks of potential loss of details and numerical instability caused by inappropriate discretization resolution.

From voxelized representations, voxel-thinning methods extract curve-skeletons by iteratively removing boundary voxels while maintaining the topology [Ma et al. 2002; Palágyi and Kuba 1999]. These methods differ mainly by their way of choosing boundary voxels and the priority for removal. Recently, Wang et al. [2008] proposed to first shrink volumetric models before applying a thinning algorithm, producing smoother skeletons, but pruning is still required. Generally, thinning methods are not robust and additional considerations are needed to prevent excessive removal of surface or curve end-points.

Distance-field methods define a distance transform for each interior point of a 3D object and detect ridges of the field to get a set of candidate voxel [Hassouna and Farag 2005; Zhou and Toga 1999]. Connecting these candidate voxels gives an approximate medial surface, and hence, like other medial-axis-based methods, this process is not robust [Wade 2000; Bitter et al. 2001]. Other field methods use alternatives to distance transform including repulsive forces [Cornea et al. 2005] and radial basis functions [Ma et al. 2003]. These general field methods determine a potential value at an interior point by considering a larger set of boundary samples and therefore are less sensitive to noise. Then, a force-following algorithm is applied to connect local extremes to form a curve-skeleton. These methods are computationally intensive due to the use of a larger boundary area, and are numerically unstable caused by the computation of the first or second-order derivatives.

Geometric methods. Geometric methods work directly on polygon meshes or point sets. Voronoi diagram is a popular geometric approach. Such methods obtain an approximate medial surface by extracting the internal edges and faces of the Voronoi diagram [Amenta et al. 2001; Dey and Sun 2006; Ogniewicz and Ilg 1992] and prune the medial surface to obtain a curve-skeleton.

Reeb-graph-based methods have gained much attention in recent years. The Reeb graph is a 1D structure whose nodes are critical points of a real-value function defined on the model surface. It encodes the topology of the model [Pascucci et al. 2007]. Different methods use various real-value functions for their specific applications, such as geodesic function [Hilaga et al. 2001] or harmonic function [Aujay et al. 2007]. To represent a curve-skeleton, a Reeb graph needs to be resampled on the model surface to obtain more skeletal nodes and then be embedded into the geometry. Aujay et al. [2007] propose a harmonic Reeb graph that uses the harmonic function, found by solving the Laplace equation. Their method requires the user to specify the boundary condition explicitly. Our geometry contraction uses Laplacian constraints and treats all vertices as boundary conditions, with their weights automatically defined.

There are other methods that do not fall within the above general classes. Sharf et al. [2007] extract skeletons both from point clouds and polygonal meshes based on a deformable model evolution. The initial extracted graph is noisy, requiring filtering and merging. Chuang et al. [2000] extract a curve skeleton by applying a force-following algorithm on the convex corners of the mesh. It is based on a generalized potential field, which is computationally

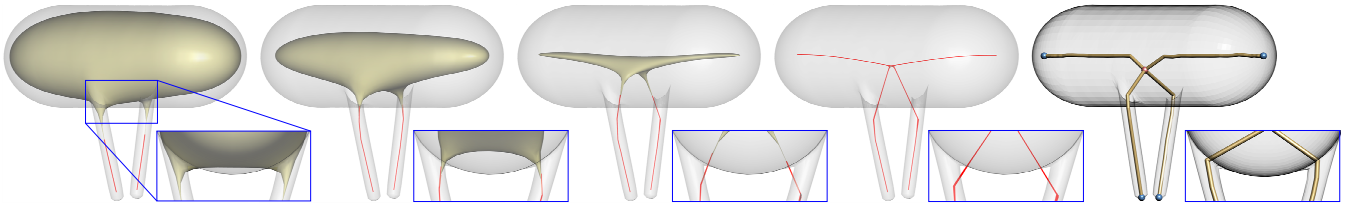

Figure 3: The geometry contraction process does not always guarantee that the contracted shape is within the object. The rightmost image is the embedded result, showing that the skeleton is correctly centered.

expensive. Katz and Tal [2003] extract a skeleton by decomposing a mesh and linking the components. Li *et al*. [2001] simplify a mesh into line segments by collapsing shorter edges and connecting the line segments. The latter two methods generate coarse curve-skeletons and do not always preserve topology.

Laplacian editing and smoothing. Each step of our iterative contraction process is a Laplacian smoothing operation constrained by all the vertices at different weights. There are previous methods that modify or approximate the mesh geometry by solving the Laplacian system, with all or a subset of vertices as the boundary constraints. In [Sorkine and Cohen-Or 2004], the original input mesh is approximated by a least-squares mesh, which is the solution of the discrete Laplace equation with uniform weighting using a carefully selected subset of vertices as the boundary constraints (called anchors).

More recently, Nealen *et al*. [2006] present a mesh optimization technique that generalizes the concept of the constrained Laplacian system for geometry smoothing and parameterization smoothing. Their method solves a Laplacian system with different weighting schemes to control the parameterization smoothing, and different right-hand-side terms of the linear system to control the geometry smoothing. In their general system, all vertices are constrained to achieve global smoothness and volume preservation. Our geometry contraction process solves a sequence of constrained Laplace equations, with weaker positional constraints than theirs, allowing the geometry to be contracted into an approximate zero-volume mesh.

3 Overview

Given a mesh $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, with vertices $\mathbf{V}$ and edges $\mathbf{E}$, where $\mathbf{V} = [\mathbf{v}_1^T, \mathbf{v}_2^T, \ldots, \mathbf{v}_n^T]^T$ are the vertex positions. We address the problem of extracting from the mesh a curve-skeleton $\mathbf{S} = (\mathbf{U}, \mathbf{B})$ with skeleton nodes $\mathbf{U}$ and edges $\mathbf{B}$, where $\mathbf{U} = [\mathbf{u}_1^T, \mathbf{u}_2^T, \ldots, \mathbf{u}_m^T]^T$ are the node positions. Our approach is based on a *geometry* contraction process that iteratively smoothes and collapses the mesh geometry in a constrained manner. This process is applied in the object-space, directly on the mesh representation, without any voxelization process. With carefully weighted contraction and attraction forces, the contraction process produces a thin skeleton shape with junctions and branches corresponding to the logical components of the object. The contraction process is presented in Section 4. To convert the contracted mesh into a 1D skeleton, we apply a *connectivity* surgery described in Section 5. Finally, to refine the skeleton's geometric embedding, we describe in Section 6 a post-process that moves the skeleton nodes to the center of their respective local regions.

Our method has the following advantages:

1. The geometry contraction process does not alter the connectivity of the original mesh, and the connectivity surgery maintains its connectedness and retains all tunnels in the mesh as loops in the resulting 1D skeleton structure. Hence, the final curve-skeleton is guaranteed to be homotopic to the original object.

2. The geometry contraction process is based on an iterative implicit smoothing operation. Thus, the method inherently deals with noise, making it insensitive to noise.

3. The method works directly on the original geometry rather than on a resampled volumetric representation, making it efficient, rotation invariant, and pose insensitive.

4. The skeletonization process yields a skeleton-mesh mapping and local thickness, which serve as important information for various applications that require shape analysis.

4 Geometry Contraction

The geometry contraction process removes details and noise from the mesh surface by applying a Laplacian smoothing that moves the vertices along their approximate curvature normal directions. Note that an unconstrained normal flow of the vertices would progressively smooth out all the details of the model and converge into a single point. To constrain the normal flow, we use an implicit updating scheme controlled by anchor points serving as positional constraints. The anchor points provide the attraction forces causing the contracting mesh to converge to a thin shape of the original object.

The vertex positions $\mathbf{V}'$ are smoothly contracted along their normal directions by solving the discrete Laplace equation: $\mathbf{L}\mathbf{V}' = 0$, where $\mathbf{L}$ is the $n \times n$ curvature-flow Laplace operator with elements

$$\mathbf{L}_{ij} = \begin{cases} \omega_{ij} = cot\ \alpha_{ij} + cot\ \beta_{ij} & \text{if } (i,j) \in \mathbf{E} \\ \sum_{(i,k)\in\mathbf{E}}^{k} -\omega_{ik} & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

and α_{ij} and β_{ij} are the opposite angles corresponding to the edge (i,j) [Desbrun et al. 1999]. With the cotangent weighting, the Laplacian coordinates $\delta = \mathbf{L}\mathbf{V} = [\delta_1^T, \delta_2^T, \ldots, \delta_n^T]^T$ approximate the (inward) curvature-flow normals, i.e., $\delta_i = -4A_i\kappa_i\mathbf{n}_i$, where A_i, κ_i and $\mathbf{n}_i$ are the local one-ring area, the approximate local mean curvature, and the approximate outward normal of vertex i, respectively. Thus, solving $\mathbf{L}\mathbf{V}' = 0$ means removing the normal components and contracting the mesh geometry. We refer to the rows in the Laplacian system as the *contraction constraints* because they provide the forces to contract the mesh.

Since the matrix $\mathbf{L}$ is singular, extra constraints are required to solve for a unique solution of $\mathbf{V}'$. To avoid degenerate solutions and ensure that the contracted mesh abstracts the original shape well, we constrain all the vertices to their current positions as soft constraints with different weights. We call them the *attraction constraints* since they attract the vertices to the geometry. The weights of the attraction constraints and the contraction constraints for different vertices are carefully set during the iterations, such that the mesh contracts to a shape that abstracts the original mesh well, rather than to a

Figure 4: (Left) The 1D structure obtained by performing the connectivity surgery on the contracted mesh in Figure 1. (Middle) The induced skeleton-mesh mapping. (Right) The resulting curve-skeleton after embedding refinement

single point. That is, we solve the following system for the vertex positions:

$$\begin{bmatrix} \mathbf{W}_L\mathbf{L} \\ \mathbf{W}_H \end{bmatrix} \mathbf{V}' = \begin{bmatrix} 0 \\ \mathbf{W}_H\mathbf{V} \end{bmatrix}, \tag{2}$$

where $\mathbf{W}_L$ and $\mathbf{W}_H$ are the diagonal weighting matrices that balance the contraction and attraction constraints, respectively. The i-th diagonal element of $\mathbf{W}_L(\mathbf{W}_H)$ is denoted $\mathbf{W}_{L,i}$ ($\mathbf{W}_{H,i}$). Note that the system (2) is over-determined. Thus, we solve it in the least-squares sense, which is equivalent to minimizing the following quadratic energy:

$$\left\| \mathbf{W}_L\mathbf{L}\mathbf{V}' \right\|^2 + \sum_i \mathbf{W}_{H,i}^2 ||\mathbf{v}_i' - \mathbf{v}_i||^2, \tag{3}$$

where the first term corresponds to the contraction constraints and the second term corresponds to the attraction constraints.

Solving system (2) once does not collapse the entire model into a 1D shape. It requires several iterations with proper weights for the process to converge to a thin shape. After the first contraction step, certain high-frequency details are filtered out and the mesh is noticeably contracted. However, using the same weights $\mathbf{W}_L$ and $\mathbf{W}_H$ in subsequent iterations would not further contract the mesh much, because the remaining details are retained by the current attraction constraints. Therefore, to increase the collapsing speed, we increase the contraction weight $\mathbf{W}_{L,i}$ for every vertex i after each iteration. In addition, to avoid over contraction, we update the attraction weight $\mathbf{W}_{H,i}$ for each vertex according to its collapsed degree determined by its local one-ring area. Specifically, we want the vertices with smaller contracted one-ring area to be attracted more strongly to their current positions and thus contract less in the next iteration. We find that such weight setting can retain the key features and appropriate branching structure in the contracted mesh. The iterative contraction process is as follow, where the superscripts t denote the iteration number:

1. Solve $\begin{bmatrix} \mathbf{W}_L^t\mathbf{L}^t \\ \mathbf{W}_H^t \end{bmatrix} \mathbf{V}^{t+1} = \begin{bmatrix} 0 \\ \mathbf{W}_H^t\mathbf{V}^t \end{bmatrix}$ for $\mathbf{V}^{t+1}$,
2. Update $\mathbf{W}_L^{t+1} = s_L\mathbf{W}_L^t$ and $\mathbf{W}_{H,i}^{t+1} = \mathbf{W}_{H,i}^0\sqrt{A_i^0/A_i^t}$, where A_i^t and A_i^0 are the current and the original one-ring areas, respectively,
3. Compute the new Laplace operator $\mathbf{L}^{t+1}$ with the current vertex positions $\mathbf{V}^{t+1}$ using equation (1).

The initial ratio of the weights $\mathbf{W}_L^0$ and $\mathbf{W}_H^0$ controls the smoothness and the degree of contraction of the first iteration result, thus it determines the amount of details retained in subsequent and final contracted meshes. Since the scale of the Laplacian coordinate is proportional to a vertex's one-ring edge lengths (under the same local one-ring shape), the contraction forces from the Laplace equations are smaller for denser models. Hence, to handle models of different sizes and resolutions, we use the following default initial setting: $\mathbf{W}_H^0 = 1.0$ and $\mathbf{W}_L^0 = 10^{-3}\sqrt{A}$, where A is the average face area of the model. Note that setting a uniform initial ratio of these weights for all vertices works well even for meshes with irregular sampling since the subsequent updating of weights is according to the local contraction degree which is resolution-independent. We found that this default setting produces good quality skeletons, with the key features retained and small surface details filtered away. All results shown in this paper are obtained with this default setting.

In our experiments, we use $s_L = 2.0$, which is found to contract all models efficiently, usually in less than 10 iterations. The iterative updating stops when the ratio of the current and the original volumes of the model is smaller than the threshold ε_{vol} (we use $\varepsilon_{vol} = 1e-6$). Figure 1 shows the sequence of contraction results for the raptor model. Each contraction iteration is essentially an analysis process, updating the attraction force of each vertex smoothly based on its local contraction ratio. The forces are defined such that the contracted regions at the thinner branches act as strong anchors retaining the key features of the object. The thinner regions of the model always collapse first, while the thicker regions take more iterations to collapse. The use of the curvature-flow Laplace operator also ensures that meshes with poor triangle quality are contracted well, with the triangle shapes maintained as much as possible during the iterative contraction (see Figure 2).

The iterative updating process increases the weights of the attraction forces as the vertices become more contracted, thus the system matrix becomes more diagonally dominant as the iteration progresses, making the updating process stable. During the construction of the new linear system to be solved in the next iteration, we also check for degenerate faces and avoid any possible numerical errors, such as infinity values or divide-by-zero error. The geometry contraction is a global constrained smoothing process, with high-frequency details and noise removed in each iteration (but with important geometry details retained by the strong attraction forces), resulting in a smaller volume. The iterations converge when the volume is close to zero.

5 Connectivity Surgery

We denote the contracted mesh as a 2D graph with vertex positions $\tilde{\mathbf{V}} = [\tilde{\mathbf{v}}_1^T, \tilde{\mathbf{v}}_2^T, \ldots, \tilde{\mathbf{v}}_n^T]^T$. The contracted mesh has an approximate zero volume with a shape that is visually a 1D skeleton. However, its connectivity is still that of the original mesh. To convert the contracted mesh into a 1D graph, we apply a connectivity surgery operation. The surgery applies a series of edge-collapses to remove collapsed faces from the degenerated mesh, until all faces have been removed. The main requirement here is to retain the shape of the degenerated mesh during this surgery process, while keeping sufficient skeletal nodes to maintain a fine correspondence between the skeleton and the original surface. We devise a cost function consisting of a shape term and a sampling term. The surgery operation is then an iterative greedy algorithm that collapses the edge having

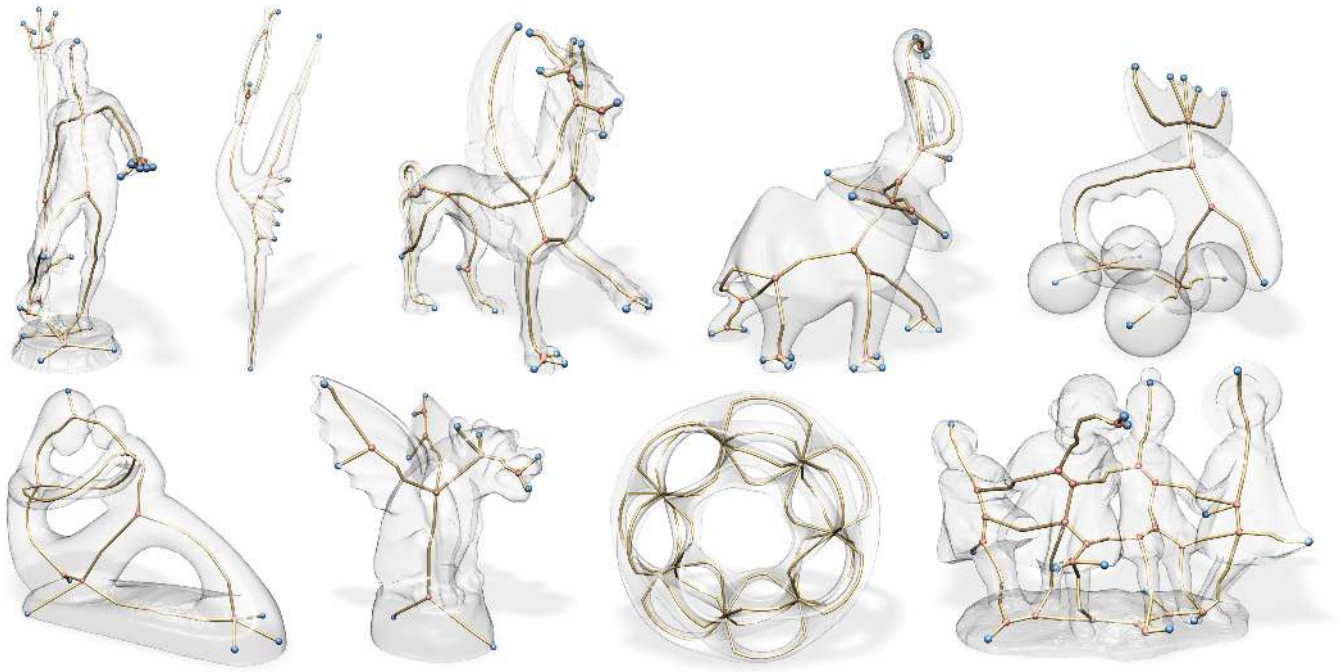

Figure 5: Top row: Neptune, dancer, feline, elephant, and elk. Bottom row: fertility, gargoyle, heptoroid and dancing children. Our method extracts good quality curve-skeletons for complex models, including high-genus models and flat regions (see the elk model).

the minimum cost in each iteration.

For simplicity, we apply the half-edge collapse. The half-edge collapse $(i \to j)$ merges vertex i to vertex j (i.e., with the resulting vertex takes the position $\tilde{\mathbf{v}}_j$) and removes all the faces that are incident to the collapsed edge. This operation never disconnects a connected component. To preserve the mesh's topology, we prevent the collapse of a tunnel by prohibiting any edge collapse $(i \to j)$ if k is a common adjacent vertex of vertices i and j but (i, j, k) is not a face in the current simplifying mesh. This simple restriction ensures that the resulting 1D skeleton has the same number of loops as the number of tunnels in the original mesh.

Shape Cost. The shape cost term is conceptually similar to the popular QEM simplification method [Garland and Heckbert 1997] which preserves mesh geometry well. QEM estimates distortion caused by an edge collapse by computing an error metric at each vertex, measuring the sum of squared distances between the vertex and its associated faces. Since the faces of our contracted mesh have zero area, this face-based error metric is clearly not applicable here. We thus apply a QEM-like mechanism over the edges. Specifically, we define the matrix $\mathbf{K}_{ij}$ for each edge (i, j) in the contracted mesh such that the value $\mathbf{p}^T(\mathbf{K}_{ij}^T\mathbf{K}_{ij})\mathbf{p}$ is the squared distance between the point $\mathbf{p}$ (in homogeneous representation) to the line defined by the edge (i, j):

$$\mathbf{K}_{ij} = \begin{bmatrix} 0 & -a_z & a_y & -b_x \\ a_z & 0 & -a_x & -b_y \\ -a_y & a_x & 0 & -b_z \end{bmatrix}, \tag{4}$$

here $\mathbf{a}$ is the normalized edge vector of the edge (i, j) and $\mathbf{b} = \mathbf{a} \times \tilde{\mathbf{v}}_i$.

The initial error metric of vertex i is the sum of all the squared distances to its adjacent edges, that is,

$$F_i(\mathbf{p}) \quad = \quad \mathbf{p}^T \sum_{(i,j)\in\mathbf{E}} (\mathbf{K}_{ij}^T\mathbf{K}_{ij})\,\mathbf{p} \;=\; \mathbf{p}^T\mathbf{Q}_i\,\mathbf{p} \tag{5}$$

To keep the shape of the contracted mesh/graph as undisturbed as possible during the simplification, we select the next edge collapse $(i \to j)$ based on the following shape cost:

$$\mathscr{F}_a(i, j) = F_i(\tilde{\mathbf{v}}_j) + F_j(\tilde{\mathbf{v}}_j). \tag{6}$$

That is, the next edge-collapse $(i \to j)$ is the one with the minimum sum of squared distances from the collapsed position $\tilde{\mathbf{v}}_j$ to the adjacent edges of vertices i and j, as well as to all the adjacent edges of the vertices that are previously collapsed to vertex i or j. After an edge collapse, we update the error matrix of vertex j as $\mathbf{Q}_j \leftarrow \mathbf{Q}_i + \mathbf{Q}_j$ such that the edges that were previously associated to vertex i are now associated to vertex j. We store the error matrix of each vertex as a 4×4 matrix and, like QEM, each cost updating step involves only a matrix addition.

Sampling Cost. The above shape cost retains the shape of the original contracted mesh well. However, it leads to over-simplification in straight regions, resulting in long skeleton edges and the loss of fine correspondence between the skeleton and the surface. Therefore, we add a sampling-aware cost term that penalizes edge collapses that generate long edges. This sampling cost measures the total distance the adjacent edges of the source vertex i travel during the edge collapse $(i \to j)$:

$$\mathscr{F}_b(i, j) = ||\tilde{\mathbf{v}}_i - \tilde{\mathbf{v}}_j|| \sum_{(i,k)\in\tilde{\mathbf{E}}} ||\tilde{\mathbf{v}}_i - \tilde{\mathbf{v}}_k||, \tag{7}$$

where $\tilde{\mathbf{E}}$ is the current simplified edge connectivity.

The Total Cost. The total cost function is a weighted sum of the shape cost and sampling cost:

$$\mathscr{F}(i, j) = w_a\mathscr{F}_a(i, j) + w_b\mathscr{F}_b(i, j). \tag{8}$$

We use $w_a = 1.0$ and $w_b = 0.1$ for all our examples. Note that during the iterations, the relative shape cost increases since the shape

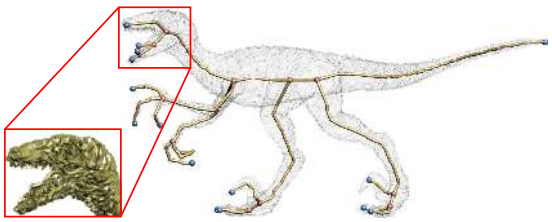

Figure 6: Our method is noise insensitive. The curve-skeleton extracted from a noisy dense raptor model.

error matrices are cumulative. Therefore, the shape cost increasingly dominates the edge collapsing process, retaining the shape of the contracted mesh. Figure 4 (left) shows an example of the connectivity surgery result.

We record all the edge collapses during the connectivity surgery process. This gives rise to a skeleton-mesh mapping that indicates, for each skeleton node k, the set of vertices Π_k on the original mesh that are contracted and collapsed to that skeleton node. Each set of vertices forms a cylinder-like shape (for non-junction nodes) or a sphere-like shape (for junction nodes). Figure 4 (middle) shows an example of the induced skeleton-mesh mapping.

6 Embedding Refinement

The iterative geometry contraction process may produce a skeletal shape that is off center or go outside the mesh, especially where adjacent object components have large differences in thickness and curvature. The main cause of this behavior is that a thicker region requires stronger contraction constraints to be contracted into a 1D shape, but these strong constraints also pull vertices at the nearby thinner regions towards the center of the thicker component (see Figure 3), causing the contracted shape to go outside the thinner components. We exploit the skeleton-mesh mapping Π induced by the skeletonization process to design a simple embedding refinement method.

The idea is to move each skeleton node k to the approximate center of its corresponding local mesh region Π_k. Each boundary of a mesh region comprises a loop of vertices that are contracted to roughly the same location (which is often off center), hence their weighted average displacement represents the shifting of the skeletal node from the center. For each boundary j, with vertex index set $\mathscr{S}_j$, we compute the weighted average displacement of the boundary vertices $\mathbf{d}_j$ during the geometry contraction:

$$\mathbf{d}_j \quad = \quad \frac{\sum_{i\in\mathscr{S}_j} l_{j,i}(\tilde{\mathbf{v}}_i - \mathbf{v}_i)}{\sum_{i\in\mathscr{S}_j} l_{j,i}}, \tag{9}$$

where $l_{j,i}$ is the total length of the two adjacent edges of a vertex i in the boundary loop j. Each regular non-junction node has two boundaries, with average displacements during contraction denoted $\mathbf{d}_1$ and $\mathbf{d}_2$. Since such a local region has a cylinder-like shape, we simply shift the skeletal node as follows: $\mathbf{u} = \mathbf{u} - (\mathbf{d}_1 + \mathbf{d}_2)/2$. For a junction node, there are more than two boundaries, therefore we shift the node by the sum of the average displacements of the boundaries, weighted by the boundary loop lengths. Finally, for a terminal node $\mathbf{u}$ (with only one boundary), we compute the average displacement $\mathbf{d}$ of all vertices in the local region, and set the new node position as $\mathbf{u} - \mathbf{d}$.

The connectivity surgery process removes all the faces in the contracted mesh, leaving a 1D connected graph as the skeleton. However, this skeleton may have more complex branching structure than the anatomical branching structure of the model. For example, the raptor model in Figure 4 (left) has two junction nodes branching to the forelegs instead of one. To simplify the branching structure and also further refine the skeleton embedding, we merge a junction node with an adjacent node if the merged junction node has a better centeredness than the original junction node. The centeredness of a junction node k is measured as the standard deviation of the distances between k's position and the vertices in Π_k, denoted σ_k. We merge the junction node k with a neighbor if $\sigma'_k < 0.9\sigma_k$, where σ'_k is the centeredness of the merged junction node. In cases where there is more than one adjacent node fulfilling this condition, we choose the neighbor with the smallest σ'_k. This process is repeated until the condition for merging with a neighbor is not met. Figure 4 (right) shows the embedding result for the raptor model.

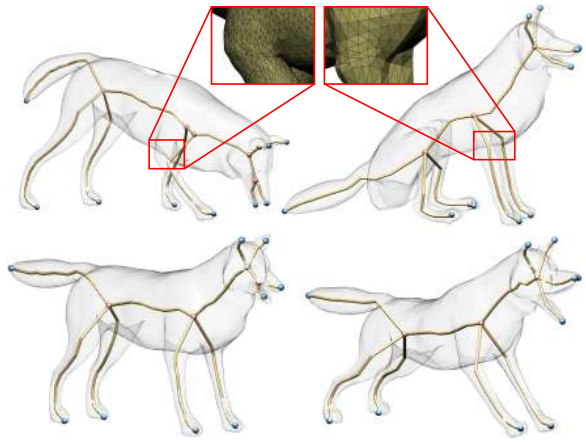

Figure 7: Our method is both pose- and sampling-insensitive. The left models have 50K triangles while the right models have 8K triangles. The extracted curve-skeletons for the different poses and resolutions are similar in terms of junctions and branches.

The embedding refinement process relocates each skeletal node to an approximate center-of-mass of its local mesh region. It successfully centers the skeleton nodes for most organic-shaped objects we tested. However, for certain man-made objects where some local regions have the center-of-mass outside the object (e.g., one with a C-shape cross-section), our embedding refinement still fails to ensure that the skeleton is inside the object.

7 Results and Discussion

In this section, we show more results of our skeleton extraction method and discuss some of its properties, demonstrated with examples. We then compare our results with previous methods, and discuss some implementation details and limitations of our method.

Figure 5 shows some ray-traced images of models and the curve-skeletons extracted using our method. Note that all the skeletons represent well the geometry and topology of the original shapes. Unlike field-based methods, our method does not require junctions detection and the chaining of skeleton nodes. The extracted skeleton is guaranteed to be homotopic to the original object. The heptoroid, fertility and dancing children are examples of high-genus models. The elk model demonstrates that our method also works well for models with thin flat regions.

The extracted skeletons have sparser nodes at the core parts of a model. This property is due to the fact that many triangle faces are contracted to the same neighboring region, causing the connectivity surgery process to require more edge collapses there to remove all the faces. In some cases, the sparseness of nodes leads to skeletons

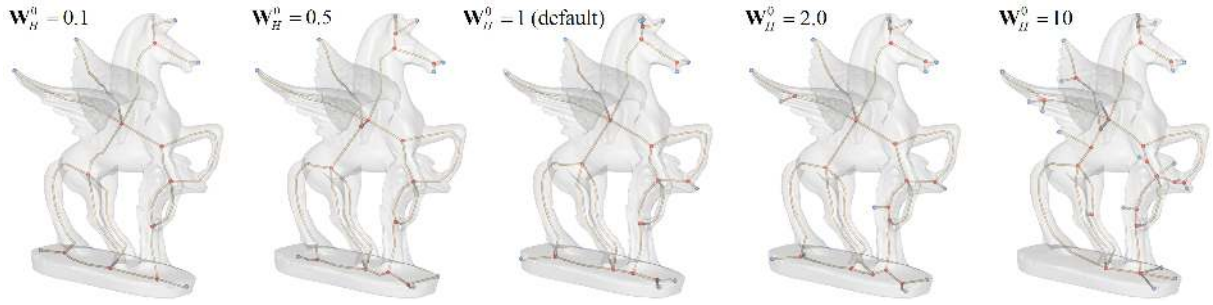

Figure 8: Our method is fairly insensitive to a small variance in the default initial ratio of the contraction and attraction forces. Here we fix W_L^0 and vary W_H^0. Nevertheless, much smaller W_H^0 values (e.g., 0.1) lead to missing branches, while much larger values (e.g., 10) lead to superfluous branches.

with a zigzag effect (e.g., dancing children in Figure 5). A solution might be to design an algorithm that removes faces using operators that modify the connectivity and appropriately introduce new vertices and edges. However, we did not find this to be necessary since these relatively large core parts have approximate spherical shapes, and are sufficiently abstracted with a few skeleton nodes for many applications. For example, the fewer skeletal nodes there does not affect the deformation space or the segmentation quality (see Section 8).

The geometry contraction is based on implicit mesh smoothing, with the contracted mesh as the solution of a least-squares system. Therefore, the noise and fine surface details are smoothed out evenly over the surface, making our method insensitive to noise and thus not requiring any post-filtering. This property is demonstrated in Figure 6 where a good quality skeleton is extracted from a highly noisy raptor model.

Our method is rotation invariant, since the geometry contraction depends only on the normal field of the mesh, which is invariant under global rotations. In addition, since the normal field is locally invariant to local rotations, the method extracts similar skeletons from different poses of the same object (see Figure 7). If the poses are of the same mesh model with identical connectivity, we can extract skeletons with identical branching structure for all poses by applying a unified connectivity surgery process (see Section 8). Finally, since the initial weighting ratio of the contraction constraints are sampling-aware, the extracted skeletons are largely independent of the size and resolution of the input model (Figure 7).

We found that a small variance in the default initial ratio of the contraction and attraction forces does affect the contracted mesh slightly, but it has little effect on the resulting skeletal branching structure (see Figure 8). Nevertheless, a much higher $\mathbf{W}_H^0$ (10 times higher than the default setting) may lead to a skeleton containing new small branches, while a much smaller $\mathbf{W}_H^0$ causes over-smoothing and produces a small contracted mesh that misses some branches.

Some skeleton extraction algorithms generate a hierarchy of skeletons with increased complexity, allowing the user to select a resolution with fewer unwanted branches. Since our method always produces a nice branching structure, we did not define such a hierarchy of skeletons. However, we note that it is possible to build a hierarchy by ordering the branches according to their approximate volume in a bottom-up manner. Specifically, a hierarchy can be constructed by iteratively removing the terminal branch with the smallest volume. In section 8.1, we use the hierarchical order provided by the approximate volume to design a simple segmentation algorithm.

Comparisons. We compare our results with the results of one algorithm from each of the five main classes of skeleton extraction methods discussed in Section 2. In the volumetric category, we adopt the implementation of Cornea et al. [2007] for potential-field, distance-field and thinning methods and use the same parameter values described in their paper. These implementations consist of the core part of the algorithms with minimum additional steps sufficient only to obtain curve-skeletons (rather than unconnected voxels or a structure with surface elements). For the geometric category, we implement a simple Reeb-graph method based on a harmonic function defined by user-specified feature points, with the geodesic distances from a source feature as the boundary values [Aujay et al. 2007]. The Reeb graph is embedded into the geometry using the centroid of each connected component in the level sets of the harmonic function. We also compare with the method of Dey and Sun [2006], which also works on the surface mesh directly and extracts a subset of an approximate medial surface as the skeleton according to geodesic distance on the original surface.

Figure 9 shows the comparison results. The skeletons extracted with the distance-field algorithm contains a large number of superfluous branches and may be disconnected. The thinning algorithm produces skeletons that are not smooth and contain small extra branches due to the propagation of irregularities from the surface to the curve-skeleton. The potential-field algorithm yields cleaner and smoother skeletons, however, a large number of branches may appear in thin and flat regions (bunny's ears) due to the existence of many critical points. Moreover, the force-following process may result in disconnected skeletons due to the poor local voxelization resolution. The results of the harmonic-based Reeb-graph algorithm are highly dependent on the user-specified feature points as well as the level-set sampling and embedding steps. The Reeb graph successfully captures the model's topology, however the nodes are critical points of a function that does not abstract the geometry well and the centeredness of the resulting skeleton is poor in general. Dey and Sun's method does not always preserve the topology since their method may extract disconnected edges from the medial surface and requires an erosing step to filter and connect the edges to form the final skeleton. In contrast, our method generates curve-skeletons that are clean, connected, topology-preserving and geometry-aware. The entire process of our method works on the original domain: the geometry contraction alters only the mesh vertex positions, the connectivity surgery alters the connectivity, and the embedding refinement alters the skeletal node positions. No node connecting and branch pruning are required and the entire process maintains a well connected and topologically preserving graph (mesh or skeleton).

Implementation Details. In term of memory usage and processing time, our skeleton extraction framework is dominated by the geom-

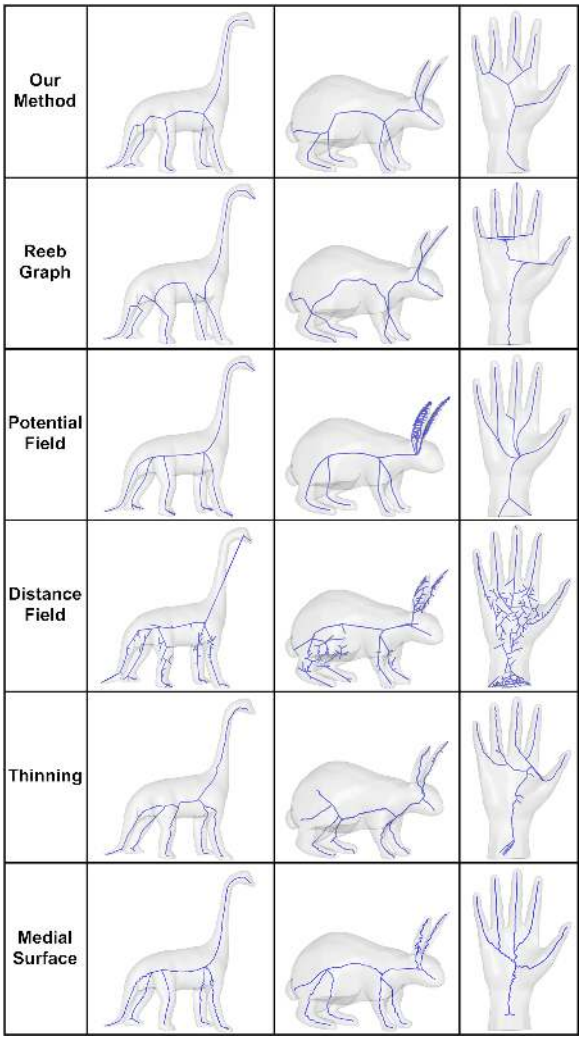

Figure 9: Comparisons with the results of various classes of curve-skeleton extraction algorithms.

etry contraction process. Each iteration requires solving a sparse linear system, which takes $O(n^3)$ time with a naïve linear solver. For efficiency, we implement a multigrid solver to solve for the new contracted vertex positions, making each iteration taking only $O(n)$ times. This greatly increases the overall performance of our system when handling large models. The connectivity surgery and the embedding refinement only takes $O(n \log n)$ and $O(mn)$ time in the worst case, respectively (recall that n is the number of vertices and m is the number of skeletal nodes). Table 1 shows the running time of some skeleton extraction examples. All data are recorded on an Intel Core 2 Dual E6600 machine with 2GB memory, using a single thread implementation.

Limitations. Our curve-skeleton extraction framework only works for closed mesh models with manifold connectivity since our geometry contraction requires a well-defined Laplace operator for every vertex. Although our method is largely insensitive to the model resolution, it cannot generate fine skeletons for very coarse models. Based on our experience testing a large set of models, we found that our method can generate good quality skeletons for common models with more than 5000 vertices.

Figure 10: (Top) A visualization of the local thickness at each skeleton node. (Bottom) Segmentation results.

Model	#Vertices	t_1	t_2	t_3	Total
Wolf	4344	1.1	0.4	0.01	2
Raptor	25000	6.3	2.7	0.03	10
Neptune	112220	116	15	0.2	138
Asian Dragon	250000	197	31	0.6	240

Table 1: Columns t_1, t_2, and t_3 show the running time (in seconds) of geometry contraction, connectivity surgery, and embedding refinement, respectively.

8 Applications

In this section, we demonstrate the effectiveness of our skeleton extraction framework for mesh segmentation and skinning animation.

8.1 Mesh Segmentation

We exploit the induced skeleton-mesh mapping and the local thickness of each skeleton node to design a simple segmentation algorithm. Each branch of the skeleton, i.e., a sequence of regular nodes between two junction nodes or between one junction node and one terminal node, corresponds to a logical component of the object. This branching structure serves as a useful guide for segmenting the mesh. We first order the branches according to their approximate volume. The volume of a branch is measured as $\sum_{i\in\Gamma} r_i^3$, where Γ is the index set of the skeleton nodes within the branch, and r_i is the approximate thickness of the mesh region Π_i corresponding to node i, defined as the average distance between the node i and the vertices in the region Π_i.

Starting from the thickest branch, we iteratively assign a cut to each branch to segment the mesh, with each cut resulting in exactly one additional segment (see accompanying video). In an automatic setting, the cutting stops when every branch has been assigned one cut. To allow user control, we also let the user specify a desired number of segments, which may be fewer or larger than the number of branches, as the terminating condition. After every branch is cut once, subsequent cuts are assigned globally.

To identify a cut within a branch, we first choose a node as the cutting node. Specifically, for each regular node s (within the branch)

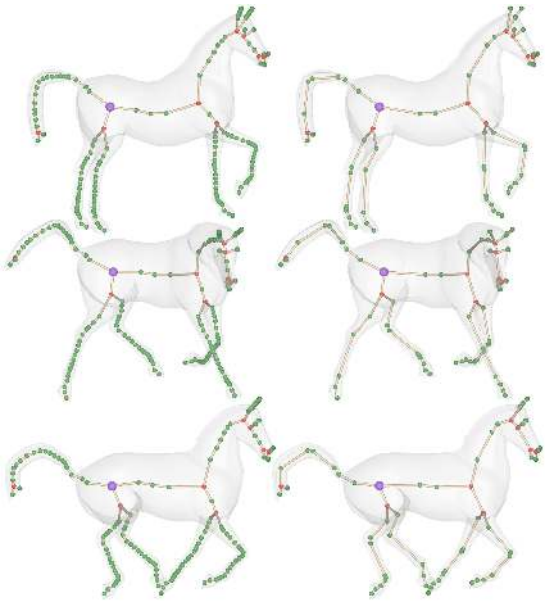

Figure 11: Compatible curve-skeletons (left column) and bone-skeletons (right column) extracted from three poses of a horse model. The root node is drawn in purple.

with adjacent nodes t_1 and t_2, we compute the 1D Laplacian of the local thickness $\delta_s = 2r_s - r_{t_1} - r_{t_2}$, and select the node with the smallest δ as the cutting node. A smaller value of δ_s (possibly negative) means that the local region of the node is more concave and should be cut first. The cutting node defines a search region in which we determine the final cutting boundary by a minimal-cut algorithm. We set the search region as the mesh regions of the cutting node and its two adjacent nodes. The capacity function for the minimal-cut algorithm is defined as in [Katz et al. 2005], which includes a concavity term and a boundary-length term to balance the quality of the local shape fitting and the smoothness of the segment boundary, respectively. Since the minimal-cut tends to go through edges with a small capacity, it finds a boundary that passes through concave short edges. Figure 10 visualizes the local thickness information and the segmentation results of the Asian dragon and Armadillo models.

8.2 Skinning Animation

Our skeleton extraction method is applied directly on the mesh domain, thus it enables the extraction of compatible skeletons from a given set of example poses of the same model. We first contract the geometry of each example mesh independently. Then, we apply the connectivity surgery to all the meshes simultaneously. Specifically, we use a unified cost function that sums up the edge collapsing costs across all contracted meshes to choose a minimum-cost edge to collapse and collapse the same edge in all the contracted meshes. Consequently, the resulting curve-skeletons of all meshes have the same final set of nodes and edges. This compatible skeleton structure adequately abstracts the geometry features of all meshes. For skinning purpose, we then down-sample the curve-skeletons into compatible bone-skeletons based on the bending angles in the different poses. We calculate, for each node with two neighbors, the bending angle between its adjacent bones and, if the maximum bending angle across all examples is less than a threshold (we use 20°), we remove that node (and connect its two adjacent nodes directly) from all curve-skeletons. This process produces bone-skeletons that have the same set of joints and bones for all meshes (see Figure 11). Since we only remove nodes with two neighbors, the resulting bone-skeletons preserve the topology of the original curve-skeletons. The set of bone-skeletons adequately represent the deformation space defined by the example poses.

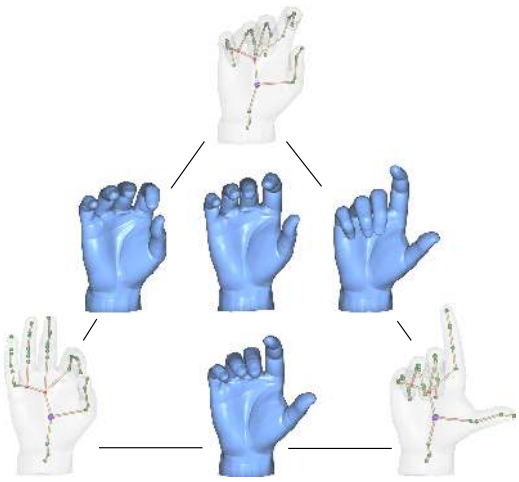

Figure 12: Example-based skinning animation. The compatible hierarchical bone-skeletons extracted from three hand models and the reconstructed poses.

For articulated animation, we need a hierarchical bone-skeleton defined by the root node, which usually represents the largest component of the object. Since the contraction paths of the vertices fill the object volume, we use them to identify the root node. We compute the approximate local volume occupied by each vertex as the product of the path length and the local area of that vertex, and compute the approximate volume of the node's region as the sum of the local approximate volumes of all its vertices. To demonstrate that the automatically extracted compatible bone-skeletons adequately represent the deformation space, we input the mesh-skeleton pairs to the rotation regression model of Wang et al. [2007]. Figure 12 shows the compatible bone-skeletons extracted from three hand models and the new poses obtained by blending the compatible bone-skeletons and reconstructing the meshes using the trained regression model.

9 Conclusion

In the paper, we present a novel geometric-based framework for extracting curve-skeletons from mesh surfaces. We iteratively remove the surface geometry without altering the connectivity to obtain a thin skeletal shape and then perform a surgery to remove the redundant connectivity to get a 1D structure. Both processes use only simple mesh processing algorithms, namely, constrained Laplacian geometry smoothing and mesh simplification. The extraction is applied directly on the mesh domain and does not require voxelization. We adopt an implicit geometry contraction process with all the vertices as boundary constraints, ensuring that the constructed curve-skeleton is smooth and noise insensitive. The connectivity surgery is performed with edge collapses which guarantee the correct connectivity and topology of the skeleton. We demonstrate the effectiveness of our skeleton extraction framework for surface segmentation and skinning animation. In the future, we will explore its use in other applications, such as interactive mesh editing and volumetric signal processing. We would also like to extend our framework to process more general meshes or other surface representations, including non-manifold meshes, surfaces with boundaries, and point set data.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments, Yu-Shuen Wang for giving a talk at HKUST that inspired this work, Youyi Zheng for his help in implementing the segmentation application, and Carl Jantzen for providing the video voice over. This work was supported in part by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project No: 620107), the National Science Council, Taiwan (NSC-96-2628-E-006-200-MY3), the Landmark Program of the NCKU Top University Project (Contract B0008), the Israeli Ministry of Science, and the Israel Science Foundation,

References

AMENTA, N., CHOI, S., AND KOLLURI, R. K. 2001. The power crust. In *Symposium on Solid Modeling and Applications*, 249–266.

AUJAY, G., HÉTROY, F., LAZARUS, F., AND DEPRAZ, C. 2007. Harmonic skeleton for realistic character animation. In *Symposium on Computer Animation*, 151–160.

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3D characters. *ACM Trans. Graph. 26*, 3, Article 72.

BITTER, I., KAUFMAN, A. E., AND SATO, M. 2001. Penalized-distance volumetric skeleton algorithm. *IEEE Transactions on Visualization and Computer Graphics 7*, 3, 195–206.

CHUANG, J.-H., TSAI, C.-H., AND KO, M.-C. 2000. Skeletonization of three-dimensional object using generalized potential field. *IEEE Trans. Pattern Anal. Mach. Intell. 22*, 11, 1241–1251.

CORNEA, N. D., SILVER, D., YUAN, X., AND BALASUBRAMANIAN, R. 2005. Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer 21*, 11, 945–955.

CORNEA, N. D., MIN, P., AND SILVER, D. 2007. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics 13*, 3, 530–548.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of ACM SIGGRAPH 99*, 317–324.

DEY, T. K., AND SUN, J. 2006. Defining and computing curve-skeletons with medial geodesic function. In *Symposium on Geometry Processing*, 143–152.

GARLAND, M., AND HECKBERT, P. S. 1997. Surface simplification using quadric error metrics. In *Proceedings of ACM SIGGRAPH 97*, 209–216.

HASSOUNA, M. S., AND FARAG, A. A. 2005. Robust centerline extraction framework using level sets. In *Proceedings on Computer Vision and Pattern Recognition - Volume 1*, 458–465.

HILAGA, M., SHINAGAWA, Y., KOHMURA, T., AND KUNII, T. L. 2001. Topology matching for fully automatic similarity estimation of 3D shapes. In *ACM Trans. Graph.*, 203–212.

KATZ, S., AND TAL, A. 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *ACM Trans. Graph.*, 954–961.

KATZ, S., LEIFMAN, G., AND TAL, A. 2005. Mesh segmentation using feature point and core extraction. *The Visual Computer 21*, 8–10, 649–658.

LI, X., WOON, T.-W., TAN, T.-S., AND HUANG, Z. 2001. Decomposing polygon meshes for interactive applications. In *Symposium on Interactive 3D graphics*, 35–42.

MA, C.-M., WAN, S.-Y., AND LEE, J.-D. 2002. Three-dimensional topology preserving reduction on the 4-subfields. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 12, 1594–1605.

MA, W.-C., WU, F.-C., AND OUHYOUNG, M. 2003. Skeleton extraction of 3D objects with radial basis functions. In *Proceedings of the Shape Modeling International*, 207–215.

NEALEN, A., IGARASHI, T., SORKINE, O., AND ALEXA, M. 2006. Laplacian mesh optimization. In *Proceedings of ACM GRAPHITE*, 381–389.

OGNIEWICZ, R., AND ILG, M. 1992. Voronoi skeletons: Theory and applications. In *Proceedings on Computer Vision and Pattern Recognition*, 63–69.

PALÁGYI, K., AND KUBA, A. 1999. A parallel 3D 12-subiteration thinning algorithm. *Graph. Models Image Process. 61*, 4, 199–221.

PASCUCCI, V., SCORZELLI, G., BREMER, P.-T., AND MASCARENHAS, A. 2007. Robust on-line computation of Reeb graphs: simplicity and speed. *ACM Trans. Graph. 26*, 3, Article 58.

SHARF, A., LEWINER, T., SHAMIR, A., AND KOBBELT, L. 2007. On-the-fly curve-skeleton computation for 3D shapes. *Computer Graphics Forum 26*, 3, 323–328.

SHI, X., ZHOU, K., TONG, Y., DESBRUN, M., BAO, H., AND GUO, B. 2007. Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics. *ACM Trans. Graph. 26*, 3, Article 81.

SORKINE, O., AND COHEN-OR, D. 2004. Least-squares meshes. In *Proceedings of Shape Modeling International*, 191–199.

WADE, L. 2000. *Automated generation of control skeletons for use in animation*. PhD thesis. Adviser-Richard E. Parent.

WANG, Y.-S., AND LEE, T.-Y. 2008. Curve skeleton extraction using iterative least squares optimization. *IEEE Transactions on Visualization and Computer Graphics*.

WANG, R. Y., PULLI, K., AND POPOVIĆ, J. 2007. Real-time enveloping with rotational regression. *ACM Trans. Graph. 26*, 3, Article 73.

WEBER, O., SORKINE, O., LIPMAN, Y., AND GOTSMAN, C. Context-aware skeletal shape deformation. *Computer Graphics Forum 26*, 3, 265–274.

YOSHIZAWA, S., BELYAEV, A., AND SEIDEL, H.-P. 2007. Skeleton-based variational mesh deformations. *Computer Graphics Forum 26*, 3, 255–264.

ZHOU, Y., AND TOGA, A. W. 1999. Efficient skeletonization of volumetric objects. *IEEE Transactions on Visualization and Computer Graphics 5*, 3, 196–209.