CGI2017 manuscript No.
(will be inserted by the editor)

Medial-Axis-Driven Shape Deformation with Volume Preservation

Lei Lan · Junfeng Yao · Ping Huang · Xiaohu Guo

Abstract The medial axis is a natural skeleton for shapes. However, it is rarely used in the existing skeleton-based shape deformation techniques. In this paper, we propose a novel medial-axis-driven skin surface deformation algorithm with volume preservation property. Specifically, an as-rigid-as-possible deformation scheme is used to deform the medial axis so that its local transform is as close as possible to a rigid transform. We maintain surface features of the deformed shape based on an implicit skinning method. Our experiments show that the proposed algorithm effectively preserves the volume of deformed shape, and addresses the bending and twisting problems associated with traditional skeleton-based shape deformation techniques.

Keywords Medial Axis · Shape Deformation · Implicit Skinning · Volume Preservation

1 Introduction

It could be observed from our natural world that the pose of humans and most animals are dependent on the pose of their internal skeleton. The geometric structure of a skeleton is simpler than its associated surface, and the motion of skeleton is rigid inherently. So, skeleton-based shape deformation methods were proposed intuitively in early works, and has been very popular in many applications. The idea of skeleton, first proposed by Blum [6], is called *medial axis*. It is defined as the set of points with at least two closest points on the shape boundary. Thus it contains the surface features and local thickness of shape. However, the existing skeleton-based shape deformation methods usually apply a stick skeleton [24, 18] or a curve skeleton [44], instead of medial axis. Because medial axis computation is sensitive to noise, it is very difficult for the early works to obtain a high quality medial axis, which is structurally simple (without undesirable spikes), accurately approximating the surface, and compact enough for computing deformation. Fortunately, with the recent advancement of medial axis simplification, such as Q-MAT [22], a high quality medial axis can be obtained by pruning unstable branches and simplification from an initially poor quality medial axis. Thus, it becomes practical now to use “real” medial axis to drive shape deformation.

The medial axis of a 3D shape is a combination of non-manifold triangle meshes with dangling edges. Thus, it seems difficult to directly integrate medial axis into the pipeline of existing skeleton-based shape deformation methods. Yoshizawa et al. [41] proposed a variational mesh deformation approach, by using medial axis for preserving geometric details and thickness of shapes. In their method, the medial axis is deformed by *Skeletal Subspace Deformations* (SSD) [4], then the deformed shape is reconstructed with the original Laplacian coordinates from the deformed medial axis. However, their method needs to build a one-to-one correspondence between the surface vertices and medial axis triangles, thus it is not general enough to handle medial axis with dangling edges (e.g., fingers of a hand).

In this paper, we propose a truly medial-axis-driven shape deformation algorithm. Different from Yoshizawa et al.’s approach [41], medial axis is directly deformed by the user, and an implicit skinning technique is proposed to drive the surface deformation, and preserve the volume of deformed shape. To achieve this goal, an

Lei Lan, Junfeng Yao and Ping Huang
Xiamen University

Xiaohu Guo
The University of Texas at Dallas

As-Rigid-As-Possible (ARAP) deformation scheme is adopted to deform medial axis so that the local transform of medial primitive is as close as possible to a rigid transform. The deformed medial axis drives each vertex of the surface to a temporary position by using their parametric coordinates, which is defined by the local enveloping primitives of medial axis. We extend the implicit skinning method [37] to rebuild surface features from the deformed medial axis. The local scalar field is simply defined based on each enveloping primitive of medial axis, which allows preserving surface features through iso-surface projection and tangential relaxation. Volume preservation can be achieved easily by adjusting the radius of spheres on medial axis, since the medial axis is deformed in an ARAP manner. The results prove our algorithm can be used to manipulate different 3D shapes and produce visually plausible deformations with volume preservation.

2 Related Work

2.1 Medial Axis Computation

Extracting the medial axis from given shapes is called *Medial Axis Transform* (MAT). MAT is typically computed by the Voronoi diagram of a set of sampled points on the shape boundary [1]. However, the computed medial axis has many undesirable spikes, making them unsuitable for any practical application. Du et al. [10] proposed a diffusion-based extraction method which combines the grassfire flow simulation and diffusion propagation. To obtain a structurally simple and compact medial axis, several methods have been proposed to simplify medial axis by identifying and pruning the spikes. To determine the points or edges to be pruned, most existing methods formulate a local or global threshold based on certain pruning criteria.

Angle-based filtering method [3, 2, 12, 9, 34] adopts the angle as global threshold, which is formed by a vertex of medial axis with its two closest points on the shape boundary. The vertex is removed from medial axis directly, if its angle is less than a user-specified threshold. Similarly, *λ-based filtering method* [8, 7] specifies a threshold λ as the smallest circumradius of closest points at the simplified medial axis. The point is removed if its circumradius of closest points is smaller than threshold λ. A local pruning criterion is applied by *Scale Axis Transformation* (SAT) [26]. It adopts a factor $s > 1$ to enlarge all medial spheres, then removes the medial spheres that are contained in other medial spheres. The final medial axis is obtained by scaling back the surviving medial spheres by the factor $1/s$. Although the method is highly effective, the quality of simplification deeply depends on the factor s. Besides the pruning criteria defined on the vertices of medial axis, Faraj et al. [11] proposed *Progressive MAT* (PMAT) method to perform MAT simplification by collapsing edges of medial axis. The pruning criterion is defined as a cost of edge-collapse, which is related to the edge length and the difference of the medial radii at the endpoints. Sun et al. [36] proposed the union of volume primitives as volume representation. The volume primitives are linear interpolation of the medial spheres. The medial axis simplification is guided by the volume approximation error. Li et al. [22] proposed an efficient and effective MAT simplification method, called *Q-MAT*. In Q-MAT, a quadratic error metric [13] is adopted to measure approximation errors in MAT simplification, and a stability ratio is proposed to distinguish the spikes of medial axis. Recently, Yan et al. [40] proposed a global measure criterion based on the *Erosion Thickness* (ET) which performs very well in differentiating boundary noises from shape features.

2.2 Shape Deformation

In our deformation algorithm, the deformation of a given 3D model is driven by its medial axis. Although, the medial axis is a natural skeleton for shapes, most existing skeleton-driven deformation methods take the form of a "stick skeleton", which could be considered as a simplified form of medial axis. Traditional skeleton-driven deformation methods assume that a skeleton is composed of rigid bones with linear [24] or non-linear [18, 19] blending weights. At run-time, the mesh vertices are rigidly transformed by its associated bones. The methods, such as *Linear Blend Skinning*(LBS) and *Dual Quaternion Skinning* (DQS) [18], have been proved to be practical for many applications due to their efficiency and simplicity. However, the quality of deformation may be degraded by the well known artifacts, such as candy-wrapper artifacts and volume-loss artifacts for LBS and bulging artifact for DQS. To prevent these artifacts, multi-linear skinning methods [39, 25, 17] introduce extra scalar weights for each bone. These extra weights add additional degrees of freedom to joints by blending separately in the subspace of bones. Helper bones with single weight [27, 28] can be estimated from given examples, and be added to diminish the angle between bones when the joints suffer from twisting and bending. Although the methods reduce the artifacts, extra weight functions and bones introduce more computation overheads. Le and Hodgins [20] proposed to precompute the optimized center of rotation for each point from the rest pose and skinning weight. During animation, these centers of rotation are used to interpolate the rigid transformation for each vertex, which can reduce the artifacts significantly with less computations. For volume preservation, Zhou et al. [42] presented volumetric graph Laplacian to encode the volumetric details of input mesh and formulated the volumetric details as a quadric energy function. The volumetric graph can be built without a solid meshing of surface's interior. Huang et al. [16] introduced the nonlinear volume constraint into subspace deformation. Zhou et al. [43] proposed an explicit mathematical model of spine-driven bending to address preserving local volume.

Our medial mesh deformation method is related to an important category of methods which try to maintain geometric relationship between mesh primitives. Sorkine and Alexa [33] solve non-linear optimization to keep local transform *As-Rigid-As-Possible* (ARAP). Sumner et al. [35] proposed an embedded deformation method, which samples some vertices from surface and organize them as a graph structure. The features of surface are encoded in the graph by applying each transformation of node on graph to deform its nearby space. A non-linear optimization problem is solved to ensure all transformations of nodes are as-close-as-possible to affine transformations. These methods can produce a high quality of deformation for surfaces.

2.3 Implicit-Function-Driven Deformation

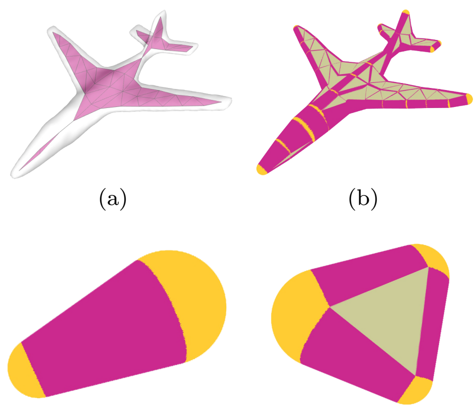
In our deformation algorithm, the surface features are maintained by the implicit function defined on medial axis. The idea of implicit-function-driven deformation have been proposed, such as Metaballs [5,4,31], polygon-based implicit primitives [32], ellipsoidal implicit primitives [21], convolution surfaces [29], and for point set surfaces [15]. Recently, Vaillant et al. [37] proposed implicit skinning method to mimic realistic deformations, such as skin contact effects and muscular bulges. An extended method [38] is proposed for interactive character skinning.

Bloomenthal [4] used medial axis with a convolution field to address the well known artifacts in the skeleton-driven shape deformation, but the medial axis drives deformation indirectly. In our algorithm, we use deformation of medial axis to drive the deformation of its 3D shape, and use the implicit function constructed from medial axis to maintain the surface details.

3 Implicit Surface Based on Medial Axis

Medial axis of a 3D shape S is a combination of non-manifold triangle meshes with dangling edges. It could be represented as a mesh M_s, called *medial mesh*, as shown in Figure 1(a). Following Sun et al. [36], a vertex of medial mesh M_s is a medial sphere, denoted m. m is embedded in 4D space by $m = \{\mathrm{c}, r\}$, where c is the center of medial sphere and r is the associated radius. Let $e_{ij} = \{m_i, m_j\}$ denote an edge of M_s, which connects two medial spheres m_i and m_j. Similarly, let $f_{ijk} = \{m_i, m_j, m_k\}$ denote a triangle face of M_s.

The volume primitive associated with edges and faces of the medial mesh is called *enveloping primitives*, as shown in Figure 1(b). For an edge e_{ij}, its enveloping primitive is swept by the family of spheres defined by linear interpolation of the medial spheres m_i and m_j: $\{m|m = \alpha m_i + (1-\alpha)m_j,\ \alpha \in [0,1]\}$. It comprises two spherical caps joined by a truncated cone, and will be called a *medial cone*, as shown in Figure 1(c). For the triangle face f_{ijk}, its primitive is obtained by linearly interpolating the three medial spheres m_i, m_j and m_k: $\{m|m = \beta_i m_i + \beta_j m_j + (1-\beta_i-\beta_j)m_k, \beta_i \in [0,1], \beta_j \in [0, 1-\beta_i]\}$. This primitive is called a *medial slab*, bounded by three spherical caps, three conical patches, and two triangles, as shown in Figure 1(d). For a 3D shape S, its medial mesh M_s is an inner skeleton with the enveloping primitives approximating S effectively.



(a) (b) (c) (d)

Fig. 1: (a) Medial mesh of Plane shape. (b) Enveloping primitives of its medial mesh. (c) Medial cone of an edge. (d) Medial slab of a triangle.

3.1 Implicit Surface of Enveloping Primitives

Our approach is inspired by the idea of implicit skinning [37], in which a scalar field is constructed with its 0.5-level-set approximating the surface. We denote the enveloping primitives of the medial mesh as C. Since the boundary surface ∂S of shape S could be approximated by the boundary surface ∂C of C, we can build the implicit surface of ∂C to approximate ∂S. In this way, when the medial mesh M_s is deformed, we can update the implicit surface of ∂C to drive the deformation of ∂S.

For a given point p, we construct the implicit function $f(\mathrm{p})$ based on the distance from p to M_s as well as the radius defined on M_s. Similar to Vaillant et al.'s approach [37], $f(\mathrm{p})$ is generated by combining a set of local fields using Ricci's max operator [30]:

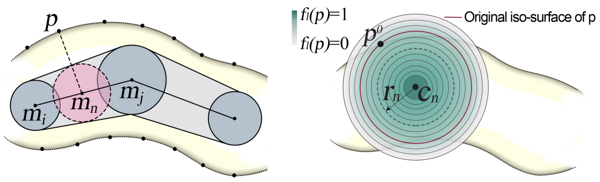
$$f(\mathrm{p}) = \max_l \{f_l(\mathrm{p})\}, \tag{1}$$

where $f_l(\mathrm{p})$ denotes a local scalar field and is constructed individually by a medial cone or a medial slab using the following distance-driven scalar function $d_l(\mathrm{p})$.

For a primitive C_l (medial cone or medial slab), $d_l(\mathrm{p})$ can be defined by finding a medial sphere $m_n = \{\mathrm{c}_n, r_n\}$ on C_l, such that the scalar function E_m is minimized: $d_l(\mathrm{p}) = \min E_m(m_n)$, where:

$$E_m(m_n) = \|\mathrm{p} - \mathrm{c}_n\|^2 - r_n^2. \tag{2}$$

We call the sphere m_n minimizing $E_m(m_n)$ as the *footprint sphere* of point p on primitive C_l, as shown in Figure 2(a).



(a) (b)

Fig. 2: (a) The footprint sphere m_n of a point p on the medial cone defined by m_i and m_j. (b) Definition of local scalar function $f_l(\mathrm{p})$ for any surface point p, with its footprint sphere $\{\mathrm{c}_n, r_n\}$. Note that the surface point may not exactly lie on the 0.5-level-set of f_l.

Without loss of generality, let us consider C_l being the medial cone of e_{ij}. In this case $\mathrm{c}_n = \alpha \mathrm{c}_i + (1-\alpha)\mathrm{c}_j$, and $r_n = \alpha r_i + (1-\alpha) r_j$. By replacing them into Eq. (2), the scalar function $d_l(\mathrm{p})$ could be seem as a quadratic minimization problem with $\alpha \in [0,1]$ being the only variable to be decided. We could reformulate Eq. (2) as follows:

$$\begin{aligned} E_m(\alpha) &= \|\mathrm{p} - (\alpha \mathrm{c}_i + (1-\alpha)\mathrm{c}_j)\|^2 - (\alpha r_i + (1-\alpha) r_j)^2 \\ &= (A_i + A_j - 2A_{ij})\alpha^2 - 2(A_j - A_{ij})\alpha + A_{ij}, \end{aligned} \tag{3}$$

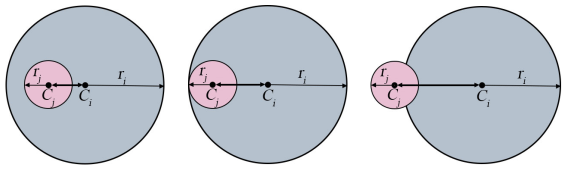
where:

$$\begin{aligned} A_i &= (\mathrm{p} - \mathrm{c}_i)^T(\mathrm{p} - \mathrm{c}_i) - r_i^2, \\ A_j &= (\mathrm{p} - \mathrm{c}_j)^T(\mathrm{p} - \mathrm{c}_j) - r_j^2, \\ A_{ij} &= (\mathrm{p} - \mathrm{c}_i)^T(\mathrm{p} - \mathrm{c}_j) - r_i r_j. \end{aligned} \tag{4}$$

The second order derivative of $E_m(\alpha)$ is:

$$\begin{aligned} H(E_m) &= 2(A_i + A_j - 2A_{ij}) \\ &= 2[(\mathrm{c}_i - \mathrm{c}_j)^T(\mathrm{c}_i - \mathrm{c}_j) - (r_i - r_j)^2] \end{aligned} \tag{5}$$

Note that $H(E_m) \leq 0$ if one sphere is inside another for the two spheres m_i and m_j, as shown in Figure 3. However, this will break the geometric morphology of a medial cone. Thus, $H(E_m) > 0$ could be proved for all valid medial cones. To acquire minimal $E_m(\alpha)$, α could be solved by $\frac{\mathrm{d}E_m(\alpha)}{\mathrm{d}\alpha} = 0$. If $\alpha < 0$ or $\alpha > 1$, we set $\alpha = 0$ or $\alpha = 1$, respectively.



(a) (b) (c)

Fig. 3: (a) $H(E_m) < 0$; (b) $H(E_m) = 0$; (c) $H(E_m) > 0$.

The footprint sphere of point p on the medial cone can be determined by $m_n = \alpha m_i + (1-\alpha) m_j$. The same method can be extended directly to compute the footprint sphere of p on a medial slab. Please refer to Appendix A for the proof of uniqueness of footprint sphere for the medial slab case.

Obiviously, $d_l(\mathrm{p})$ defined above is a globally-supported scalar function in the range of $[-r_n^2, +\infty]$. To allow for the compositions of the $f_l(\mathrm{p})$ according to Eq. (1), we use the following mapping function $t_r(\cdot)$ to convert $d_l(\mathrm{p})$ to a compactly-supported scalar function [37]:

$$t_r(\cdot) = \frac{-3}{16}(\cdot)^5 + \frac{5}{8}(\cdot)^3 - \frac{15}{16}(\cdot) + \frac{1}{2}, \tag{6}$$

And, the local scalar functions of $f_l(\mathrm{p})$ are computed as follows:

$$f_l(\mathrm{p}) = \begin{cases} 1, & if \quad \frac{d_l(\mathrm{p})}{r_n} \leq -1, \\ 0, & if \quad \frac{d_l(\mathrm{p})}{r_n} > 1, \\ t_r(\frac{d_l(\mathrm{p})}{r_n}), & otherwise, \end{cases} \tag{7}$$

where r_n is the radius of the footprint sphere of p. In this way, the local scalar functions of $f_l(\text{p})$ are mapped to the range of $[0, 1]$. The boundary surface ∂C_l is mapped to 0.5-level-set. When p is outside the surface ∂C_l, we have $f_l(\text{p}) \in [0, 0.5)$; when p is inside the surface ∂C_l, we have $f_l(\text{p}) \in (0.5, 1]$. Figure 2(b) gives an illustration of this local scalar function $f_l(\text{p})$. Since the boundary surface ∂C is just an approximation of the boundary surface ∂S, for any given point p on ∂S, it may not exactly lie on the 0.5-level-set of f_l. In Section 4.3 we present a projection operator to maintain the surface points on their original level-set throughout the surface deformation.

3.2 Parametric Coordinate

For any surface point p on ∂S, we would like to maintain its "relative position" w.r.t. the footprint sphere m_n and its corresponding medial primitive C_l. We call such "relative position" as the *parametric coordinate* of p in C_l. Whenever the medial mesh M_s is deformed by users, each surface point p can be directly deformed to the position according to its parametric coordinate, before applying further iso-surface projections (Section 4.3) and tangential relaxations (Section 4.4).

We define the parametric coordinate of p w.r.t. C_l using the polar coordinate system. Specifically, if C_l is a medial cone, the parametric coordinate is defined as $\delta^c = \{\alpha, \rho, \varphi, \theta\}$; if C_l is a medial slab, the parametric coordinate is defined as $\delta^s = \{\beta_i, \beta_j, \rho, \varphi, \theta\}$. α (or β_i and β_j) is the linear interpolation parameter of m_n on medial cone (or medial slab), where m_n is the footprint sphere of p on C_l. ρ is the distance from p to the spherical surface of m_n, and $(\rho + r_n, \varphi, \theta)$ is the polar coordinate of p w.r.t. its footprint sphere m_n. The polar coordinate system of m_n is aligned with the local coordinate system defined on the medial primitive C_l, except for its origin being at c_n. Specifically, in the local coordinate system of C_l, we can transform parametric coordinates to Cartesian coordinates $[x, y, z]^T$ as follows:

$$\begin{aligned} x &= c_{n,x} + (\rho + r_n)\sin\varphi\cos\theta, \\ y &= c_{n,y} + (\rho + r_n)\sin\varphi\sin\theta, \\ z &= c_{n,z} + (\rho + r_n)\cos\theta. \end{aligned} \tag{8}$$

4 Deformation Algorithm

4.1 Overview

In the previous section, a time-varying global scalar field $f(\text{p})$ is defined by combining the local scalar fields $f_l(\text{p})$, which is defined based on the footprint spheres on enveloping primitives of medial mesh M_s. For every vertex p of the boundary surface ∂S, we initially compute the global field values $f(\text{p})$ and its parametric coordinates w.r.t. the medial mesh M_s. When the user performs as-rigid-as-possible (ARAP) deformation to the medial mesh, p is first transformed to a temporary position by using its parametric coordinates. Then, the surface features are rebuilt by projecting vertices to their original iso-surfaces along the current gradient direction of $f(\text{p})$. Tangential relaxation is further applied to evenly distribute vertices on the deformed surface, in order to capture the deformed shape and avoid self-intersections between neighboring triangles. Since the medial mesh is deformed in an ARAP manner, we provide an approach to preserve the global volume of the shape by simply adjusting the radii of the medial mesh. The pipeline of our deformation algorithm is illustrated in Figure 4.

4.2 Medial Mesh Deformation

Firstly, each step begins from the deformation of medial mesh M_s. In our algorithm, users are allowed to manipulate medial mesh by selecting medial spheres and manipulate them to desired positions. We choose the ARAP scheme to guide the deformation of medial mesh, and the energy term is formulated based on the rigid shape matching [23] as:

$$E_d(\{\text{R}_j, \text{t}_j\}, \{\tilde{\text{c}}_i\}) = \sum_i \sum_{j \in \mathscr{N}(i)}^{n} \|\text{R}_j \text{c}_{ij}^0 + \text{t}_j - \tilde{\text{c}}_i\|^2, \tag{9}$$

where $\mathscr{N}(i)$ denotes the set of indices of medial primitives $\{C_j | j \in \mathscr{N}(i)\}$ which are connected with the medial sphere m_i. R_j and t_j are the rotation matrix and translation vector for medial primitive C_j. $\tilde{\text{c}}_i$ is the deformed position of the medial sphere m_i. In the rest pose of the medial mesh, each medial sphere m_i has a corresponding center position c_{ij}^0 in its connected medial primitive C_j. If each C_j is individually transformed by rotation R_j and translation t_j, these c_{ij}^0 will be transformed along with each C_j and will not agree with each other on their positions in general.

When the user manipulates some medial spheres to their desired positions, we need to minimize E_d and solve for $\{\text{R}_j, \text{t}_j\}$ and $\{\tilde{\text{c}}_i\}$ in an iterative manner: (1) fix the positions $\{\tilde{\text{c}}_i\}$ of medial spheres, and solve for rigid transforms $\{\text{R}_j, \text{t}_j\}$ of primitives; (2) fix $\{\text{R}_j, \text{t}_j\}$, and solve for $\{\tilde{\text{c}}_i\}$. The details of our ARAP deformation computation are given in Appendix B.

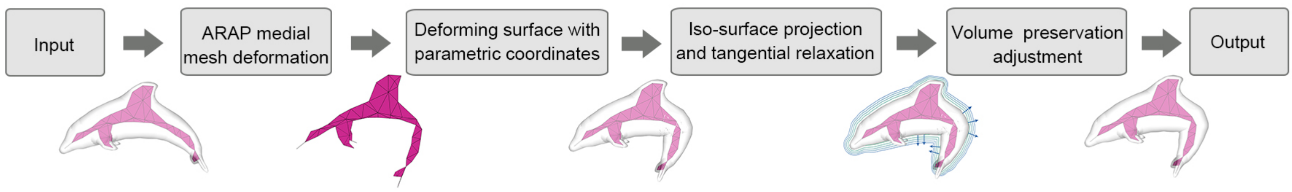


Fig. 4: The overall pipeline of our deformation algorithm.

4.3 Iso-surface Projection

After the medial mesh M_s is deformed, we first deform the boundary surface ∂S according to the parametric coordinates of the vertices. However, due to the existence of bending deformation, such initial deformation of ∂S may not agree with the deformed global scalar field $f(\mathrm{p})$ as defined in Eq. (1). We still need to further project them to the surface corresponding to their original $f(\mathrm{p})$ value.

Note that $f(\mathrm{p})$ is composed from local scalar functions $f_l(\mathrm{p})$, which is further dependent on the footprint sphere of p. Thus our projection is formulated as an iteration of the following two steps: (1) based on the current position of p^k, find its footprint sphere $m_n^k = \{\mathrm{c}_n^k, r_n^k\}$; (2) project p^k along the gradient direction $\frac{\mathrm{p}^k - \mathrm{c}_n^k}{\|\mathrm{p}^k - \mathrm{c}_n^k\|}$:

$$\mathrm{p}^{k+1} = \mathrm{c}_n^k + \lambda \frac{\mathrm{p}^k - \mathrm{c}_n^k}{\|\mathrm{p}^k - \mathrm{c}_n^k\|}, \tag{10}$$

where $\lambda = |(r_n^k)^2 + r_n^k \frac{\|\mathrm{p}^0 - \mathrm{c}_n^0\|^2 - (r_n^0)^2}{r_n^0}|^{\frac{1}{2}}$ is the marching length in one projection. Here the superscript k is the iteration number, and the superscript 0 denotes the rest state. Note that the above step (2) will move p exactly onto its original level-set defined by $t_r(\frac{\|\mathrm{p}^0 - \mathrm{c}_n^0\|^2 - (r_n^0)^2}{r_n^0})$ in Eq. (6). Thus the iteration typically converges in very few iterations.

4.4 Tangential Relaxation

For some large bending operations, such as the bending of elbow, the surface vertices may become too sparse for the outer elbow region, or too dense for the inner elbow region. Thus it is important to relax the stretching or squeezing of the surface mesh. The tangential relaxation steps [37] are conducted as follows:

$$\mathrm{p}^{k+1} = (1 - \mu)\mathrm{p}^k + \mu \sum_j \Phi_j \mathrm{q}_j^k, \tag{11}$$

where the superscript k is the iteration number, $\mu = 0.2$ is a constant, q_j^k is the position of its 1-ring neighbor projected onto its tangent plane, and Φ_j is its associated mean value coordinate.

We use the following quantity ε to control the tangential relaxation steps:

$$\varepsilon = \frac{\sum_i \|\mathrm{p}_i^k - \mathrm{p}_i^{k+1}\|^2}{n_v}, \tag{12}$$

where n_v is the total number of surface vertices. Tangential relaxation is repeated until $\varepsilon \leqslant 1.0 \times 10^{-6}$ or the number of iterations exceeds 20.

After iso-surface projection introduced above, the tangential relaxation is performed. Since the tangential relaxation moves each vertex on its tangent plane, after tangential relaxation, iso-surface projection is executed again to guarantee the surface vertices stay on their original $f(\mathrm{p})$ value. Figure 5 shows the illustration of iso-surface projection and tangential relaxation.

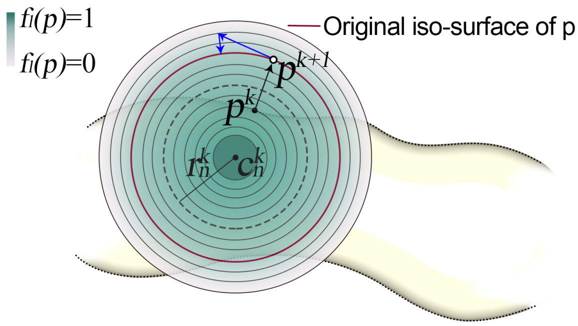


Fig. 5: Iso-surface projection moves the vertice p along black arrow. Tangential relaxation moves the vertice p along blue arrow.

4.5 Global Volume Preservation

Since the deformation is directly driven by medial mesh, it is easy to preserve the volume of deformed 3D shapes by adjusting the radii of the medial mesh. Firstly, we can compute the volume of the 3D shape S at rest state by:

$$V^0 = \frac{1}{6} \sum_{\{i,j,k\} \in \mathscr{T}} \mathrm{p}_i \cdot (\mathrm{p}_j \times \mathrm{p}_k), \tag{13}$$

where $\mathscr{T}$ is the set of triangles on the surface ∂S, and p_i, p_j, and p_k are the vertices of triangle $\{i, j, k\}$. During the surface deformation, we update the radii of all medial spheres on M_s uniformly, in order to preserve the volume of S. We denote such uniform radius change as $\triangle r$, and estimate the new volume as follows:

$$V = \frac{1}{6} \sum_{\{i,j,k\} \in \mathscr{T}} (\mathrm{p}_i + \triangle r \mathrm{n}_i) \cdot [(\mathrm{p}_j + \triangle r \mathrm{n}_j) \times (\mathrm{p}_k + \triangle r \mathrm{n}_k)], \tag{14}$$

where n_i, n_j, n_k are the surface normal at p_i, p_j, p_k, respectively. We formulate the following volume preserving energy to be minimized:

$$E_v(\triangle r) = (V - V^0)^2, \tag{15}$$

Newton iterations are used to solve $\triangle r$. It should be noted that some of the medial spheres on the medial mesh are already small enough, so reducing their radii by $\triangle r$ may result in negative radii. Thus for a medial sphere m_i, if $\triangle r < -\frac{2}{3} r_i$, we simply skip the radius update for m_i.

After adjusting the radii for medial spheres, the global scalar function $f(\mathrm{p})$ is updated, so iso-surface projection needs to be applied again.

5 Results

We have implemented our algorithm as an interactive editing system. Our algorithm is written in Microsoft Visual C++ 2012 and run on an Intel(R) Xeon E5645 CPU at 2.40GHz. Medial axis is extracted and simplified using Q-MAT [22]. The interactive system allows users to select some medial spheres as "fixed", and manipulate some other medial spheres by controlling their positions. When the user picks a medial sphere and drags it, only ARAP medial mesh deformation and surface deformation with parametric coordinates are computed on-the-fly. Iso-surface projection, tangential relaxation, and volume preservation are executed once the user releases the control and stops dragging the medial sphere. All the experiments are animated at 45 fps.

To evaluate the quality of volume preservation, we use the following error metric:

$$e_v = \frac{|V^d - V^0|}{V^0} \times 100\%,$$

where V^d is the volume of deformed shape and V^0 is its original volume.

We show deformation results of Rapter and three different models in Figures 6 and Figures 11. The results are summarized in Table 1. It demonstrates very good volume preservation for stretching and rotational deformation. The computation time highly depends on the number of surface vertices and primitives of medial axis. Longer computation times are needed for the steps of iso-surface projection and tangential relaxation, such as the deformations of Dophin, Chair and Rapter.

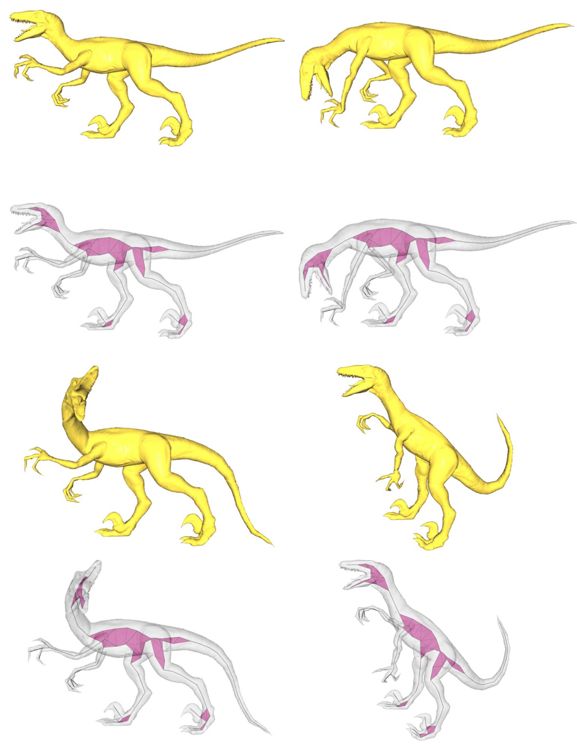


Fig. 6: Deformations of Rapter models.

Figures 7 shows highlighted views of Rapter deforsation by opening its mouth. It can be seen that the features of its teeth are well preserved throughout such large rotational deformation.

Figure 8 demonstrates that our algorithm could addresses the twisting and bending problems, which are notorious in the traditional skeleton-based shape deformation techniques. The the candy-wrapper artifacts caused by twisting (see Figure 8(a)) and volume-loss artifacts caused by bending (see Figure 8(b)) may appear in the deformation before tangential relaxation, and disappear after we relax the surfaces along the tangential directions of their iso-surfaces (see Figure 8(c) and 8(d)).

	#Ver	#Tri	#Pri	V^0	V^d	e_v	DC (ms)	IP (ms)	TR (ms)	VP (ms)
Raptor	20876	41592	158	0.019449	0.019656	1.0615%	64	637	721	38
Hand	6191	12378	36	0.053282	0.053258	0.4517%	31	39	76	13
Dophin	15100	30196	65	0.215781	0.217734	0.9052%	45	302	387	35
Chair	10500	21008	60	0.126142	0.12725	0.8749%	39	123	241	29

Table 1: Experimental results. From left to right, the first to seventh columns are names of models, the number of vertices, the number of triangles, the number of primitives. The last four columns are the average computation time for ARAP medial mesh deformation and deforming surfaces with parametric coordinates (DC), iso-surface projection (IP), tangential relaxation (TR) and volume preservation (VP).

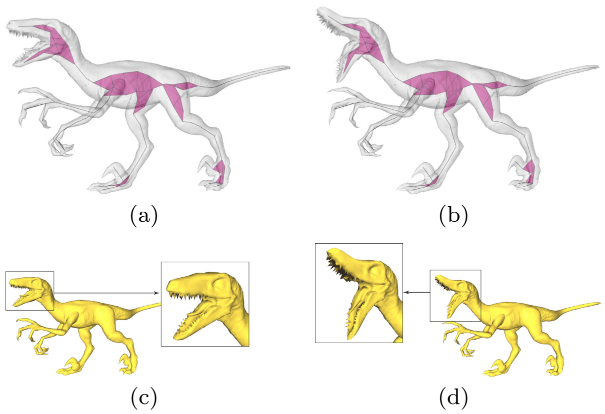


Fig. 7: Our method preserves surface features throughout the deformation, as illustrated on the teeth of Raptor.

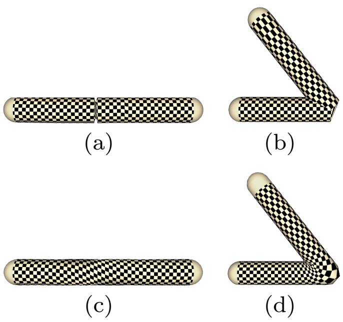


Fig. 8: (a) and (b): Twisting and bending the model before tangential relaxation. (c) and (d): After tangential relaxation.

To highlight the effect of volume preservation in our algorithm, we repeat a deformation process with and without volume preservation. In the experiment, a plane with 6448 vertices is modified to become a "flying bird" through interactive deformations. Figure 9 shows comparison of volume between two deformations during the entire process. The vertical axis represents the error of volume preservation and the horizontal axis represents the deformation steps from $t0$ to $t7$. Figure 10 illustrates the comparison of thickness after the deformation at $t2$.

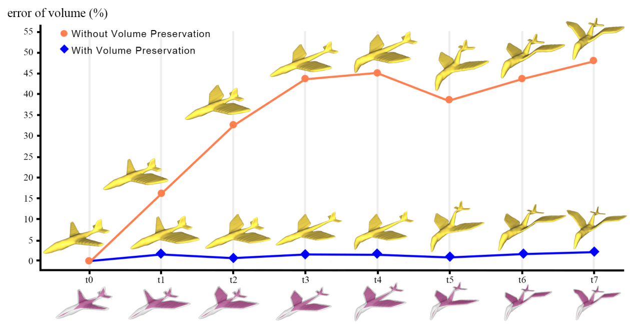


Fig. 9: Experiment on volume preservation. The orange circles represent the volume of deformed model without volume preservation, the blue diamonds represent the volume of deformed model with volume preservation.

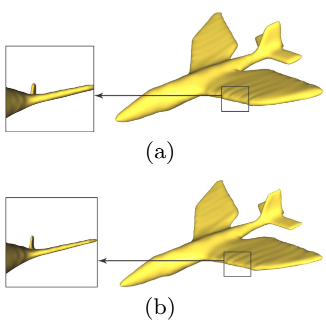


Fig. 10: Comparison of thickness at $t2$: (a) Without volume preservation ($e_v = 32.49\%$). (b) With volume preservation ($e_v = 0.7713\%$)

(a) Hand (b) Dolphin (c) Chair

Fig. 11: Deformations of Hand, Dolphin and Chair models.

6 Conclusion and Future Work

In this paper, we present a shape deformation algorithm driven by medial-axis, which is essentially a skeleton structure for representing 3D shapes, but provides more information about the surface, such as thickness and features, as compared to the traditional stick-skeleton or curve-skeleton. We combine ARAP deformation with radius adjustment on the medial mesh to guarantee global volume preservation during the shape deformation process. The iso-surface projection with tangential relaxation can not only preserve surface features, but also address the candy-wrapper and volume-loss artifacts in twisting and bending associated with traditional deformation methods.

Our current implementation of the deformation algorithm is not fully optimized in performance. In the future, we would like to further explore potential optimization approaches, e.g., GPU-based implicit skinning with tangential relaxation, in order to achieve real-time performance. Note our current ARAP deformation energy on the medial mesh does not penalize bending deformation. We would like to consider adding the bending energy to the medial mesh deformation and try to accommodate different kinds of kinematic constraints, e.g., rotational constraints for neighboring medial primitives, rigidity constraints for some medial primitives, etc.

Acknowledgement

The authors would like to thank the anonymous reviewers for their valuable suggestions. This work was partially supported by the Project of College Excellent Professional Leaders Overseas Visitor of Fujian Provincial Education Department in 2015, and the open funding project of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (Grant No. BUAA-VR-16KF-22).

References

1. Amenta, N., Bern, M.: Surface reconstruction by Voronoi filtering. Discrete & Computational Geometry **22**(4), 481–504 (1999)
2. Amenta, N., Choi, S., Kolluri, R.K.: The power crust. In: Proceedings. ACM Symposium on Solid Modeling and Applications, pp. 249–266 (2001)
3. Attali, D., Montanvert, A.: Modeling noise for a better simplification of skeletons. In: Proceedings. International Conference on Image Processing, pp. 13–16 vol.3 (1996)
4. Bloomenthal, J.: Medial-based vertex deformation. In: Proceedings. ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 147–151 (2002)
5. Bloomenthal, J., Lim, C.: Skeletal methods of shape manipulation. In: Proceedings. Shape Modeling International, p. 44 (1999)
6. Blum, H.: A transformation for extracting new descriptors of shape. Models for the Perception of Speech & Visual Form **19**, 362–380 (1967)
7. Chaussard, J., Couprie, M., Talbot, H.: A discrete λ-medial axis. In: Discrete Geometry for Computer Imagery (2009)
8. Chazal, F., Lieutier, A.: The "λ-medial axis". Graphical Models **4**(4), 304–331 (2005)

9. Dey, Tamal, K., Zhao, Wulue: Approximate medial axis as a Voronoi subcomplex. Computer-Aided Design **36**(2), 195–202 (2004)
10. Du, H., Qin, H.: Medial axis extraction and shape manipulation of solid objects using parabolic pdes. In: Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications, SM '04, pp. 25–35. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2004)
11. Faraj, N., Thiery, J.M., Boubekeur, T.: Progressive medial axis filtration. In: SIGGRAPH Asia 2013 Technical Briefs, pp. 1–4 (2013)
12. Foskey, M., Lin, M.C., Manocha, D.: Efficient computation of a simplified medial axis. Journal of Computing & Information Science in Engineering **3**(4), 96–107 (2003)
13. Garland, M.: Surface simplification using quadric error metrics. In: Conference on Computer Graphics & Interactive Techniques, pp. 209–216 (1997)
14. Giulini, D.: The rich structure of minkowski space. In: Minkowski Spacetime: A Hundred Years Later, vol. 165, pp. 83–132 (2009)
15. Guo, X., Hua, J., Qin, H.: Scalar-function-driven editing on point set surfaces. IEEE Computer Graphics & Applications **24**(4), 43–52 (2004)
16. Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.Y., Teng, S.H., Bao, H., Guo, B., Shum, H.Y.: Subspace gradient domain mesh deformation. ACM Trans. Graph. **25**(3), 1126–1134 (2006)
17. Jacobson, A., Sorkine, O.: Stretchable and twistable bones for skeletal shape deformation. ACM Transactions on Graphics **30**(6), 61–64 (2011)
18. Kavan, L., Collins, S., Žára, J., O'Sullivan, C.: Geometric skinning with approximate dual quaternion blending. ACM Trans. Graph. **27**(4), 105:1–105:23 (2008)
19. Kavan, L., Žára, J.: Spherical blend skinning: a real-time deformation of articulated models. In: Proceedings. Symposium on Interactive 3D Graphics, pp. 9 – 16 (2005)
20. Le, B.H., Hodgins, J.K.: Real-time skeletal skinning with optimized centers of rotation. ACM Transactions on Graphics **35**(4), 1–10 (2016)
21. Leclercq, A., Akkouche, S., Galin, E.: Mixing triangle meshes and implicit surfaces in character animation. In: Proceedings. Eurographic Workshop on Computer Animation and Simulation, pp. 37–47 (2001)
22. Li, P., Wang, B., Sun, F., Guo, X., Zhang, C., Wang, W.: Q-MAT: Computing medial axis transform by quadratic error minimization. ACM Transactions on Graphics **35**(1) (2015)
23. Ller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless deformations based on shape matching. ACM Transactions on Graphics **24**(3), 471–478 (2005)
24. Magnenat-Thalmann, N., Laperrière, R., Thalmann, D.: Joint-dependent local deformations for hand animation and object grasping. In: Proceedings on Graphics Interface '88, pp. 26–33. Canadian Information Processing Society, Toronto, Ont., Canada, Canada (1988)
25. Merry, B., Marais, P., Gain, J.: Animation space: A truly linear framework for character animation. ACM Transactions on Graphics **25**(4), 1400–1423 (2006)
26. Miklos, B., Giesen, J., Pauly, M.: Discrete scale axis representations for 3d geometry. ACM Transactions on Graphics **29**(4), 157–166 (2010)
27. Mohr, A., Gleicher, M.: Building efficient, accurate character skins from examples. ACM Transactions on Graphics **21**(3), 562–568 (2003)
28. Mukai, T.: Building helper bone rigs from examples. In: Proceedings. Symposium on Interactive 3D Graphics and Games, pp. 77–84 (2015)
29. van Overveld, C.W.A.M., van den Broek, B.C.: Using the implicit surface paradigm for smooth animation of triangle meshes. In: Proceedings. International Conference on Computer Graphics, p. 214 (1999)
30. Ricci, A.: A constructive geometry for computer graphics. Computer-Aided Design (2), 157–160 (1973)
31. Shen, J., Thalmann, D.: Interactive shape design using metaballs and splines. Proceedings. Implicit Surfaces (2002)
32. Singh, K., Parent, R.: Implicit function based deformations of polyhedral objects. Proceedings. Eurographics Workshop on Implicit Surfaces (1995)
33. Sorkine, O., Alexa, M.: As-rigid-as-possible surface modeling. Proceedings. Eurographics Symposium on Geometry Processing pp. 109–116 (2007)
34. Sud, A., Foskey, M., Manocha, D.: Homotopy-preserving medial axis simplification. International Journal of Computational Geometry & Applications **17**(5), 39–50 (2005)
35. Sumner, R.W., Schmid, J., Pauly, M.: Embedded deformation for shape manipulation. ACM Transactions on Graphics **26**(3), 80 (2007)
36. Sun, F., Choi, Y., Yu, Y., Wang, W.: Medial meshes for volume approximation. CoRR **abs/1308.3917** (2013). URL http://arxiv.org/abs/1308.3917
37. Vaillant, R., Barthe, L., Guennebaud, G., Cani, M.P., Rohmer, D., Wyvill, B., Gourmel, O., Paulin, M.: Implicit skinning: Real-time skin deformation with contact modeling. ACM Transactions on Graphics **32**(4), 96–96 (2013)
38. Vaillant, R., Guennebaud, G., Barthe, L., Wyvill, B., Cani, M.P.: Robust iso-surface tracking for interactive character skinning. ACM Transactions on Graphics **33**(6), 1–11 (2014)
39. Wang, X.C., Phillips, C.: Multi-weight enveloping: Least-squares approximation techniques for skin animation. In: Proceedings. ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 129–138 (2002)
40. Yan, Y., Sykes, K., Chambers, E., Letscher, D., Ju, T.: Erosion thickness on medial axes of 3d shapes. Acm Transactions on Graphics **35**(4), 1–12 (2016)
41. Yoshizawa, S., Belyaev, A., Seidel, H.P.: Skeleton-based variational mesh deformations. Computer Graphics Forum **26**(3), 255–264 (2007)
42. Zhou, K., Huang, J., Snyder, J., Liu, X., Bao, H., Guo, B., Shum, H.Y.: Large mesh deformation using the volumetric graph laplacian. Acm Transactions on Graphics **24**(3), 496–503 (2005)
43. Zhuo, W., Rossignac, J.: Fleshing: Spine-driven bending with local volume preservation. In: Computer Graphics Forum, p. 295–304 (2013)
44. Ztireli, A.C., Baran, I., Popa, T., Dalstein, B., Sumner, R.W., Gross, M.: Differential blending for expressive sketch-based posing. In: Proceedings. ACM Siggraph/Eurographics Symposium on Computer Animation, pp. 155–164 (2013)

A Footprint Sphere on Medial Slab

For the medial slab C_l of a triangle face f_{ijk}, the *footprint sphere* of point p on the slab is the sphere $m_n = \{\mathrm{c}_n, r_n\}$ with minimal scalar function $E_m(m_n)$ defined as:

$$E_m(m_n) = \|\mathrm{p} - \mathrm{c}_n\|^2 - r_n^2. \quad (16)$$

Suppose $\{\mathrm{c}_i, r_i\}$, $\{\mathrm{c}_j, r_j\}$, and $\{\mathrm{c}_k, r_k\}$ are the three spheres defining this medial slab, then we have $\mathrm{c}_n = \beta_i \mathrm{c}_i + \beta_j \mathrm{c}_j + (1-\beta_i-\beta_j)\mathrm{c}_k$, and $r_n = \beta_i r_i + \beta_j r_j + (1-\beta_i-\beta_j) r_k$. The scalar function $E_m(m_n)$ can be written as:

$$\begin{aligned} E_m(\beta_i, \beta_j) &= \|\mathrm{p} - (\beta_i \mathrm{c}_i + \beta_j \mathrm{c}_j + (1-\beta_i-\beta_j)\mathrm{c}_k)\|^2 - (\beta_i r_i \\ &\quad + \beta_j r_j + (1-\beta_i-\beta_j) r_k)^2 \\ &= A_i \beta_i^2 + A_j \beta_j^2 + A_k (1-\beta_i-\beta_j)^2 + 2B_{ij}\beta_i\beta_j \\ &\quad + 2B_{ik}\beta_i(1-\beta_i-\beta_j) + 2B_{jk}\beta_j(1-\beta_i-\beta_j), \end{aligned} \quad (17)$$

where

$$\begin{aligned} A_i &= (\mathrm{p}-\mathrm{c}_i)^T(\mathrm{p}-\mathrm{c}_i) - r_i^2, \\ A_j &= (\mathrm{p}-\mathrm{c}_j)^T(\mathrm{p}-\mathrm{c}_j) - r_j^2, \\ A_k &= (\mathrm{p}-\mathrm{c}_k)^T(\mathrm{p}-\mathrm{c}_k) - r_k^2, \\ B_{ij} &= (\mathrm{p}-\mathrm{c}_i)^T(\mathrm{p}-\mathrm{c}_j) - r_i r_j, \\ B_{ik} &= (\mathrm{p}-\mathrm{c}_i)^T(\mathrm{p}-\mathrm{c}_k) - r_i r_k, \\ B_{jk} &= (\mathrm{p}-\mathrm{c}_j)^T(\mathrm{p}-\mathrm{c}_k) - r_j r_k. \end{aligned} \quad (18)$$

The Hessian matrix of $E_m(\beta_i, \beta_j)$ is:

$$\begin{aligned} \mathrm{H}(E_m) &= \begin{pmatrix} \frac{\partial^2 E_m}{\partial \beta_i^2} & \frac{\partial^2 E_m}{\partial \beta_i \partial \beta_j} \\ \frac{\partial^2 E_m}{\partial \beta_j \partial \beta_i} & \frac{\partial^2 E_m}{\partial \beta_j^2} \end{pmatrix} \\ &= 2\begin{pmatrix} A_i + A_k - 2B_{ik} & A_k + B_{ij} - B_{ik} - B_{jk} \\ A_k + B_{ij} - B_{ik} - B_{jk} & A_j + A_k - 2B_{jk} \end{pmatrix}. \end{aligned} \quad (19)$$

Let us denote:

$$\begin{aligned} H_{11} &= A_i + A_k - 2B_{ik}, \\ H_{12} &= A_k + B_{ij} - B_{ik} - B_{jk}, \\ H_{22} &= A_j + A_k - 2B_{jk}. \end{aligned} \quad (20)$$

Since two Euclidean vectors v and w satisfy the law of cosines:

$$(\mathrm{v}-\mathrm{w})^T(\mathrm{v}-\mathrm{w}) = \mathrm{v}^T\mathrm{v} + \mathrm{w}^T\mathrm{w} - 2\mathrm{v}^T\mathrm{w}, \quad (21)$$

we can rewrite H_{11}, H_{12}, and H_{22} as:

$$\begin{aligned} H_{11} &= (\mathrm{c}_i-\mathrm{c}_k)^T(\mathrm{c}_i-\mathrm{c}_k) - (r_i-r_k)^2, \\ H_{12} &= (\mathrm{c}_i-\mathrm{c}_k)^T(\mathrm{c}_j-\mathrm{c}_k) - (r_i-r_k)(r_j-r_k), \\ H_{22} &= (\mathrm{c}_j-\mathrm{c}_k)^T(\mathrm{c}_j-\mathrm{c}_k) - (r_j-r_k)^2. \end{aligned} \quad (22)$$

If we denote the 4-dimensional vectors v_i and v_j in Minkowski space as:

$$\begin{aligned} \mathrm{v}_i &= [(\mathrm{c}_i-\mathrm{c}_k)^T, (r_i-r_k)]^T, \\ \mathrm{v}_j &= [(\mathrm{c}_j-\mathrm{c}_k)^T, (r_j-r_k)]^T, \end{aligned} \quad (23)$$

then H_{11}, H_{12}, and H_{22} can be written using Minkowski inner product $g(\cdot,\cdot)$ as:

$$\begin{aligned} H_{11} &= g(\mathrm{v}_i, \mathrm{v}_i), \\ H_{12} &= g(\mathrm{v}_i, \mathrm{v}_j), \\ H_{22} &= g(\mathrm{v}_j, \mathrm{v}_j). \end{aligned} \quad (24)$$

As shown in Figure 3 of the paper, as long as the two spheres m_i and m_k are not arranged as "one inside another", then we can guarantee $H_{11} > 0$. Similarly $H_{22} > 0$ also holds for general valid configurations of m_j and m_k.

Since both $g(\mathrm{v}_i, \mathrm{v}_i) > 0$ and $g(\mathrm{v}_j, \mathrm{v}_j) > 0$, i.e., v_i and v_j are *spacelike* vectors in Minkowski space, they satisfy the usual Cauchy-Schwarz inequality (see Formula 3 of [14]):

$$g(\mathrm{v}_i, \mathrm{v}_i) g(\mathrm{v}_j, \mathrm{v}_j) \geq g(\mathrm{v}_i, \mathrm{v}_j)^2, \quad (25)$$

with equality holds when v_i and v_j are co-linear. For a general medial slab, v_i and v_j will not be co-linear, thus we have $g(\mathrm{v}_i, \mathrm{v}_i) g(\mathrm{v}_j, \mathrm{v}_j) - g(\mathrm{v}_i, \mathrm{v}_j)^2 > 0$.

Thus the determinant of Hessian $\mathrm{H}(E_m)$ is positive:

$$|\mathrm{H}(E_m)| = 2(H_{11}H_{22} - H_{12}^2) > 0, \quad (26)$$

The scalar function E_m will have a unique global minimum, and thus the footprint sphere can be solved from minimizing Eq. (17) with $[\frac{\partial E_m}{\partial \beta_i}, \frac{\partial E_m}{\partial \beta_j}] = [0, 0]$.

B Minimizing ARAP Deformation Energy

For our medial mesh M_s, the ARAP deformation energy is defined as:

$$E_d(\{\mathrm{R}_j, \mathrm{t}_j\}, \{\tilde{\mathrm{c}}_i\}) = \sum_i \sum_{j \in \mathscr{N}(i)}^{n} \|\mathrm{R}_j \mathrm{c}_{ij}^0 + \mathrm{t}_j - \tilde{\mathrm{c}}_i\|^2, \quad (27)$$

where $\mathscr{N}(i)$ denotes the set of indices of medial primitives $\{C_j | j \in \mathscr{N}(i)\}$ which are connected with the medial sphere m_i. R_j and t_j are the rotation matrix and translation vector for medial primitive C_j. $\tilde{\mathrm{c}}_i$ is the deformed position of the medial sphere m_i, and c_{ij}^0 is its corresponding center position in C_j at the rest pose.

Each medial primitive has its local coordinate system with origin on the center of the primitive. So the translation vector can be simply: $\mathrm{t}_j = \frac{1}{3}(\tilde{\mathrm{c}}_i + \tilde{\mathrm{c}}_j + \tilde{\mathrm{c}}_k)$ for triangle f_{ijk}, and $\mathrm{t}_j = \frac{1}{2}(\tilde{\mathrm{c}}_i + \tilde{\mathrm{c}}_j)$ for edge e_{ij} on the medial mesh. To minimize E_d in Eq. (27), the rotation matrix R_j of all primitives and the medial sphere central positions $\tilde{\mathrm{c}}_i$ need to be solved in turn iteratively.

In each iteration, if we first fix the value of $\tilde{\mathrm{c}}_i$, we can minimize E_d and solve the rotation matrices R_j as follows. Let us denote $\tilde{\mathrm{c}}_{ij} = \tilde{\mathrm{c}}_i - \mathrm{t}_j$, for each primitive $j \in \mathscr{N}(i)$. Then:

$$\begin{aligned} E_d &= \sum_i \sum_{j \in \mathscr{N}(i)}^{n} \|\mathrm{R}_j \mathrm{c}_{ij}^0 + \mathrm{t}_j - \tilde{\mathrm{c}}_i\|^2 \\ &= \sum_j \sum_{i \in \mathscr{V}(j)} \|\mathrm{R}_j \mathrm{c}_{ij}^0 - \tilde{\mathrm{c}}_{ij}\|^2 \\ &= \sum_j \sum_{i \in \mathscr{V}(j)} ({\mathrm{c}_{ij}^0}^T \mathrm{c}_{ij}^0 - 2{\mathrm{c}_{ij}^0}^T \mathrm{R}_j^T \tilde{\mathrm{c}}_{ij} + \tilde{\mathrm{c}}_{ij}^T \tilde{\mathrm{c}}_{ij}), \end{aligned} \quad (28)$$

where $\mathscr{V}(j)$ is the set of vertices for primitive j. Since ${\mathrm{c}_{ij}^0}^T \mathrm{c}_{ij}^0$ and $\tilde{\mathrm{c}}_{ij}^T \tilde{\mathrm{c}}_{ij}$ are fixed for now, minimizing E_d is equivalent to maximizing the following F_d:

$$\begin{aligned} F_d &= \sum_j \sum_{i\in\mathscr{V}(j)} ({\mathrm{c}_{ij}^0}^T \mathrm{R}_j^T \tilde{\mathrm{c}}_{ij}) \\ &= trace(\sum_j \sum_{i\in\mathscr{V}(j)} ({\mathrm{c}_{ij}^0}^T \mathrm{R}_j^T \tilde{\mathrm{c}}_{ij})) \\ &= \sum_j \sum_{i\in\mathscr{V}(j)} trace({\mathrm{c}_{ij}^0}^T \mathrm{R}_j^T \tilde{\mathrm{c}}_{ij}) \\ &= \sum_j \sum_{i\in\mathscr{V}(j)} trace(\mathrm{R}_j^T \tilde{\mathrm{c}}_{ij} {\mathrm{c}_{ij}^0}^T) \\ &= \sum_j trace(\mathrm{R}_j^T \sum_{i\in\mathscr{V}(j)} \tilde{\mathrm{c}}_{ij} {\mathrm{c}_{ij}^0}^T) \\ &= \sum_j trace(\mathrm{R}_j^T \mathrm{A}_j) \\ &= \sum_j F_d^j, \end{aligned} \tag{29}$$

where the matrix $\mathrm{A}_j = \sum_{i\in\mathscr{V}(j)} \tilde{\mathrm{c}}_{ij} {\mathrm{c}_{ij}^0}^T$, and $F_d^j = trace(\mathrm{R}_j^T \mathrm{A}_j)$. Since each R_j is independent of each other, we can maximize each F_d^j individually. We decompose A_j using Singular Value Decomposition (SVD): $\mathrm{A}_j = \mathrm{U}_j \mathrm{D}_j \mathrm{V}_j^T$. Then $F_d^j = trace(\mathrm{R}_j^T \mathrm{U}_j \mathrm{D}_j \mathrm{V}_j^T) = trace(\mathrm{V}_j^T \mathrm{R}_j^T \mathrm{U}_j \mathrm{D}_j)$. Since D_j is a diagonal matrix, the trace achieves maximum when $\mathrm{V}_j^T \mathrm{R}_j^T \mathrm{U}_j$ is an identity matrix. So R_j can be solved as:

$$\mathrm{R}_j = \mathrm{U}_j \mathrm{V}_j^T. \tag{30}$$

After getting the rotation matrices for primitives, we can assume R_j to be fixed, and minimize E_d by solving for medial sphere center positions $\tilde{\mathrm{c}}_i$. They can be simply solved as:

$$\tilde{\mathrm{c}}_i = \frac{1}{|\mathscr{N}(i)|} \sum_{j\in\mathscr{N}(i)} (\mathrm{R}_j \mathrm{c}_{ij}^0 + \mathrm{t}_j), \tag{31}$$

where $|\mathscr{N}(i)|$ is the number of primitives that are connected to medial sphere i.

It should be noted that in each iteration, we compute the optimal R_j and $\tilde{\mathrm{c}}_i$ in turn, and each of these steps will decrease the energy E_d until converged.