Skeleton Based As-Rigid-As-Possible Volume Modeling

Shaoting Zhang, Andrew Nealen and Dimitris Metaxas

Computer Science, Rutgers University

Abstract

Shape deformation and editing are important for animation and game design. Based on as-rigid-as-possible (ARAP) surface modeling, an efficient approach is proposed to approximately preserve the volume of an object with large-scale deformations. The classical ARAP surface modeling uses two-stage iterations to recover rotations and preserve edge lengths. However, there is no volume preserving constraint, which may cause undesired artifacts. We show that the volume can be roughly kept by leveraging the skeleton information. First a skeleton is selected, and points are evenly generated on the skeleton. Then each point is correlated with several vertices on the surface of the object. The connectivity between the skeleton and the surface is defined as skeleton edges*, which can be easily added into the linear system of the ARAP method as additional rows without breaking the manifoldness or sacrificing speed. Since this linear system is able to preserve the lengths of both the surface and skeleton edges, the area of cross sections and the volume between cross sections can be approximately preserved. In our experiments, we show that the rotations are natural and volumes are roughly kept. The system achieves real time performance for surface meshes with 5k vertices.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—geometric algorithms, languages, and systems

1. Introduction

Shape deformation is widely used in many applications such as animation, game design and virtual reality. During the past decade, people have put a lot of effort in as-rigid-as-possible (ARAP) shape modeling to obtain natural deformations. ARAP deformation means that the shape should be locally preserved, and the deformations are locally similar. Previous works try to constrain transformation matrices to achieve the ARAP effect. In 2D, there have been many excellent methods. Alexa et al. [ACOL00] successfully applied ARAP deformation to shape interpolation. Sorkine et al. [SCOL*04] showed that the similarity transformation in 2D can be completely characterized with a linear expression. Igarashi et al. [IMF05] presented a two-step closed form algorithm to deform 2D shapes. The first step allows translation, rotation and uniform scaling. The second step adjusts the scaling locally. Schaefer et al. [SMW06] devised a moving least squares framework for 2D space warping, where each element of the space grid deforms ARAP, and the warping is controlled by positional constraints on several grid points. These methods have shown promising performance on 2D ARAP deformations.

3D ARAP deformation is more challenging because of the nonlinearity of similarity transformations. Botsch et al. [BK04] formulated a linearized system to represent ARAP deformation. It allows an efficient optimization, but causes artifacts such as local details and general shape distortions for large deformations. Zorin et al. [ZSS97] proposed a multi-resolution technique to deform low-frequency components of the surface first, followed by adding back high-frequency details as local displacements. However, this solution leads to local self-intersections when the deformation introduces bending. Sorkine et al. [SCOL*04] proposed Laplacian Surface Editing (LSE), which uses a first-order approximation of similarity transformations. It works well when only moderate rotations are involved in the deformation. Nonlinear ARAP approaches were also proposed, such as the volumetric graph Laplacian [ZHS*05], Laplacian constraints [HSL*06], PriMo [BPGK06] and ARAP surface modeling [SA07], all of which produce compelling results. Particularly, [SA07] uses an iterative scheme to minimize its carefully designed energy formulation, which is easy to implement and closely related to the widely used LSE. The rotations are natural and edge lengths are preserved, even for

large scale deformations. However, there is no volume preserving constraint in this system, which may cause undesired artifacts.

Based on ARAP surface modeling, we propose an efficient and easy-to-implement approach to approximately preserve the volume by leveraging skeleton information. The energy formulation of [SA07] is adopted to obtain rotations and to preserve edge lengths. In addition, a skeleton is generated and points on the skeleton are correlated with vertices on the surface. The connectivity of each cross section is defined as *skeleton edges* and is added into the traditional linear ARAP system as additional rows. Using the edge length preserving property, the cross section area and volume can be roughly preserved. The contributions of our method are the following: 1) We propose an approach to integrate the skeleton information with ARAP surface modeling to approximately preserve the volume without breaking its manifoldness or sacrificing the speed; 2) In terms of implementations, our approach can be easily extended from modeling frameworks relying on LSE or ARAP surface modeling.

2. Algorithms

Linear LSE: Laplacian coordinates represent each point as the weighted difference between such point and its neighborhoods. Given original coordinates ($V = [v_1, ..., v_n]^T$), the connectivity, and m control points, the coordinates of the reconstructed object ($V' = [v'_1, ..., v'_n]^T$) can be obtained by minimizing the quadratic energy function:

$$\|LV' - \delta\|_2^2 + \Sigma_{i=1}^{m} \|v'_{c_i} - v_{c_i}\|_2^2 \tag{1}$$

where L represents the discrete Laplace-Beltrami operator using uniform or cotangent weights [PP93], [NISA06], $\delta = LV$ (computed in the beginning), and v_c denotes the control point. The first term penalizes the shape difference after reconstruction, and the second term penalizes the change of positions of control points. With m control points, (1) can be minimized as a $(n+m) \times n$ overdetermined linear system:

$$\begin{bmatrix} L \\ I_c \end{bmatrix} V' = \begin{bmatrix} \delta \\ V_c \end{bmatrix} \tag{2}$$

where I_c is the index matrix of V_c, which maps each V'_c to V_c. The reconstructed shape looks generally natural when rotations are small.

Rotation and edge constraints: [SA07] introduced a nonlinear approach to find natural rotations, which consists of 2 steps. In the first step, an initial guess is calculated by linear LSE (or defined as the previous frame). In the second step, a rotation matrix R_i is computed for each vertex by minimizing:

$$\Sigma_{j \in \mathbb{N}(i)} \|(v'_i - v'_j) - R_i(v_i - v_j)\|_2^2 \tag{3}$$

where $\mathbb{N}(i)$ represents the set of neighbors of the ith vertex and R_i is the rotation matrix for the ith vertex and depends on its neighbors. Minimizing (3) amounts to minimizing the change of edge lengths. Denoting the edge $e_{ij} = v_i - v_j$, one can write the covariance matrix S_i as:

$$S_i = \Sigma_{j \in \mathbb{N}(i)} (w_{ij} e_{ij} e'^T_{ij}) \tag{4}$$

where w_{ij} is the cotangent weight of e_{ij}, and e'_{ij} is the edge after reconstruction. The rotation matrix R_i of the ith vertex is derived from the singular value decomposition (SVD) of the covariance matrix $S_i = U_i \Sigma_i V_i^T$, and $R_i = V_i U_i^T$ [SA07], [Sor09]. A new linear system is then obtained by plugging R_i into the righthand side of (2) based on the derivative of (3) that ensures convergence [SA07]. The two procedures can be alternately performed to recover natural rotations when there is no large stretching. However, there is no volume preserving constraint in this method.

Skeleton and volume constraints: Volume preservation is important in many real-world applications. By using the volumetric mesh, the volume magnitude can be preserved to some degree. However, the volumetric mesh conflicts with traditional ARAP and Laplacian methods, which are designed for 2D manifolds. Furthermore, it increases the computation time. In this work, the skeleton information is incorporated to approximately keep the volume without breaking the manifoldness or significantly increasing the computation complexity.

To initialize the model, two steps are needed. In the first step, a skeleton is manually defined, and points are evenly generated from the skeleton (black spheres in Fig. 1). Since our focus is on the deformation, robust and automatic skeleton extractions are left for future investigation. In the second step, each skeleton point is correlated with vertices on the surface (grey points in Fig. 1). In this step, rays perpendicular to the skeleton segment are emitted from each skeleton point and intersected with the surface. The surface vertices closest to these intersections are connected to that skeleton point. The connectivity between skeleton points and surface vertices is defined as *skeleton edge* (dashed lines in Fig. 1). Skeleton edges corresponding to the same skeleton point define a cross section (grey discs in Fig. 1). Although the object's topology changes after adding skeleton edges, the surface and each cross section are still 2D manifolds, which can be easily added to the ARAP framework. Note that there are no skeleton edges between skeleton points, since this would break the manifoldness.

By minimizing (3), the optimization strives to converge to a state where the edge length error is small if the modeling constraints do not impose large stretching on the surface. Because of the length preservation of both skeleton and surface edges, the areas of triangles consisting of these edges are kept when skeleton edges are relatively dense. Thus areas of cross sections consisting of these triangles are approximately the same. Combining the areas with surface edges connecting two cross sections, the volume in between can be roughly kept unchanged. Based on the same idea, the vol-

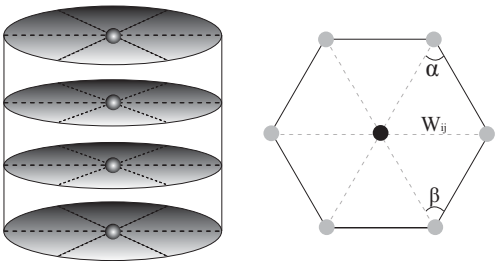

Figure 1: *Left: the skeleton points (black spheres) are inside the cylinder. Each point is connected to surface vertices by skeleton edges. Right: a cross section view, corresponding to a grey disc on the left. The skeleton edges are represented by dashed lines. Grey points are on the cylinder's surface. W_{ij} is the cotangent weight of the skeleton edge below it, calculated as $W_{ij} = \frac{1}{2}(\cot\alpha + \cot\beta)$ [NISA06].*

ume of the main object is approximately preserved during deformations, to the extent allowed by the constraints (3). To add these skeleton edges as new constraints, we append additional rows to (2):

$$\begin{bmatrix} L & | & 0 \\ I_c & | & 0 \\ & L_s & \end{bmatrix} \begin{bmatrix} V' \\ V'_s \end{bmatrix} = \begin{bmatrix} \delta \\ V_c \\ \delta_s \end{bmatrix} \quad (5)$$

where L_s, V'_s and δ_s are respectively the Laplace-Beltrami operator, cartesian coordinates and Laplacian coordinates of skeleton points. Assuming there are l skeleton points and m control points, then (5) is a $(n+m+l)\times(n+l)$ overdetermined linear system. Since L_s depends on the connectivity and relative positions between skeleton points and surface vertices, it multiplies both V' and V'_s to compute δ_s, while L only depends on V' when computing δ. In other words, V'_s is coupled to V', but not vice versa. This setting ensures that the first $(n+m)$ rows are the same as (2), except additional l zero columns. Thus the shape and control point constraints are the same. The newly added l rows naturally incorporate the volume preserving constraint without breaking the original system. As (2) corresponds to (1), solving (5) amounts to minimizing the following energy function:

$$\|LV' - \delta\|_2^2 + \|L_s V'_{all} - \delta_s\|_2^2 + \Sigma_{i=1}^{m} \|v'_{c_i} - v_{c_i}\|_2^2 \quad (6)$$

where V'_{all} contains all surface vertices and skeleton points. Thus adding these new constraints corresponds to appending a new term to the energy function.

The whole optimization framework is extended from that of [SA07]. An initial guess is obtained by solving (5). Then rotation matrices are computed by using SVD, and are plugged into the righthand side of (5) based on the derivative of (3). The two procedures are alternately performed until convergence. This approach computes a natural rotation for each vertex and approximately preserves the edge length and the volume. In addition, the computation complexity of (5) is similar to that of (2) since (5) only appends l rows and columns. Because the skeleton is generally 1D, l is much smaller than the number of surface vertices. Thus the new system does not significantly decrease the efficiency.

3. Experiments and discussions

The C++ implementation was run on a Intel Core2 Quad 2.40GHz CPU with 8G RAM. Fig. 2 compares the linear LSE, ARAP surface modeling and our method. The cactus model (620 vertices, 1,236 polygons) is a challenging test case due to its long protruding features. With enough iterations, ARAP surface modeling can recover rotations for these features. However, when rotations are large, the bent volume is shrunk. Skeleton constraints can alleviate this problem. The regions marked by black boxes show the volume differences using two methods. Fig. 3 compares ARAP surface modeling with our method on a horse model (2,482 vertices, 4,960 polygons). Using our method, the volume of the main body is mostly preserved. More results are available online.

Tab. 1 shows the relative root mean square errors of edge lengths and volume magnitudes. As expected, the skeleton based method performs much better in terms of volume preservation. It also shows the calculation time for each iteration of different methods. Our method only increases processing time by about 3% compared to ARAP surface modeling. Generally, it achieves real time performance for objects with 5k vertices and 3 iterations.

Limitations also exist. Self-intersection may happen, so the cross sections may cross, which makes the model less stable. Thus mechanism to prevent self-intersection is necessary. Robust skeleton generation for complex models is also challenging. Currently only one skeleton is considered. Mesh contraction [ATC*08] can be employed to generate skeletons with branches.

To summarize, we proposed an approach to approximately preserve the volume without breaking the manifoldness of traditional ARAP or increasing the computational complexity. Our method is easy-to-implement and may be useful to systems relying on ARAP techniques.

References

[ACOL00] ALEXA M., COHEN-OR D., LEVIN D.: As-rigid-as-possible shape interpolation. In *SIGGRAPH '00* (2000), pp. 157–164.

[ATC*08] AU O. K.-C., TAI C.-L., CHU H.-K., COHEN-OR D., LEE T.-Y.: Skeleton extraction by mesh contraction. *ACM Trans. Graph. 27*, 3 (2008), 1–10.

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. *ACM Trans. Graph. 23*, 3 (2004), 630–634.

[BPGK06] BOTSCH M., PAULY M., GROSS M., KOBBELT L.: Primo: coupled prisms for intuitive surface modeling. In *SGP '06* (2006), pp. 11–20.

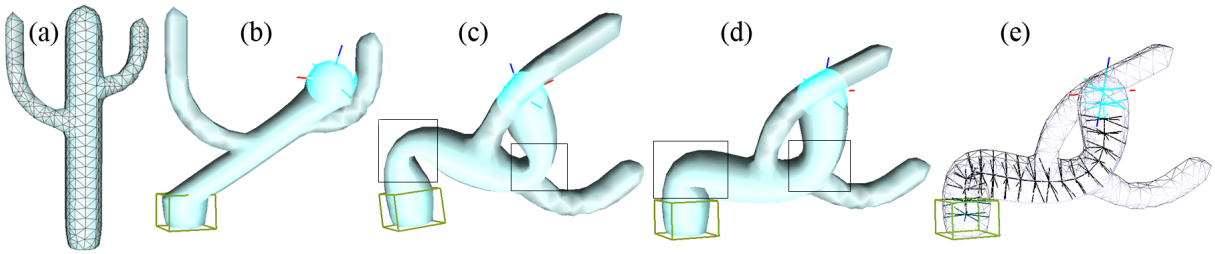

Figure 2: *Dancing cactus. (a) original model, (b) linear LSE, (c) ARAP surface modeling (100 iterations), (d) using skeleton information with ARAP method (100 iterations), (e) skeleton edges and wire frames of (d). The deformations are achieved by anchoring the bottom and translating the top. The black boxes show volume differences between traditional ARAP surface modeling and skeleton based method. Note that only translations are involved.*

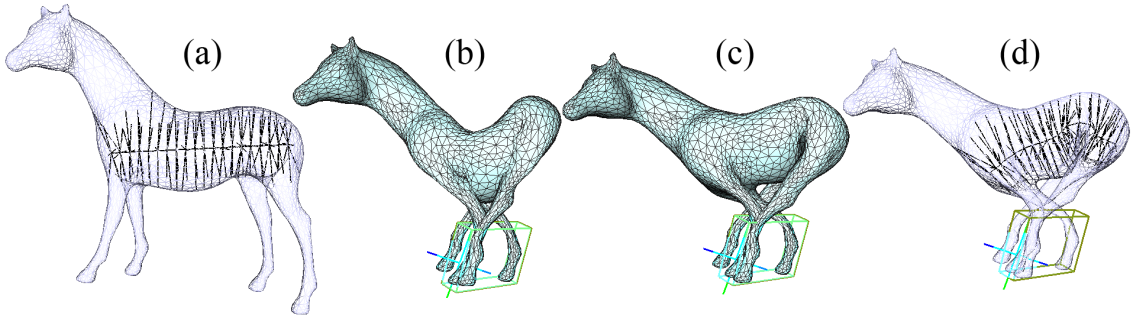

Figure 3: *Running horse. (a) original model with skeleton edges displayed, (b) ARAP surface modeling (3 iterations), (c) using skeleton information with ARAP method (3 iterations), (d) skeleton edges and wire frames of (c). The deformations are achieved by anchoring the front legs and translating the rear legs.*

Model	RRMS-E	RE-V	Times
Fig. 2 (b)	0.126	0.453	0.017
Fig. 2 (c)	0.074	0.131	0.024
Fig. 2 (d)	**0.075**	**0.056**	**0.025**
Fig. 3 (b)	0.068	0.356	0.117
Fig. 3 (c)	**0.040**	**0.125**	**0.121**

Table 1: *Errors and processing times of Fig. 2 and 3. RRMS-E stands for relative root mean square errors of edge lengths. RE-V means the relative error of volume magnitudes: abs(original volume - current volume)/(original volume). "Times" is the calculation time (seconds) for each iteration.*

[HSL*06] 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 (2006), 1126–1134.

[IMF05] IGARASHI T., MOSCOVICH T., F. H. J.: As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (2005), 1134–1141.

[NISA06] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: Laplacian mesh optimization. In *GRAPHITE '06* (2006), pp. 381–389.

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experiment. Math* (1993).

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *SGP '07* (2007), pp. 109–116.

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *SGP '04* (2004), pp. 175–184.

[SMW06] SCHAEFER S., MCPHAIL T., WARREN J.: Image deformation using moving least squares. *ACM Trans. Graph. 25*, 3 (2006), 533–540.

[Sor09] SORKINE O.: Least-squares rigid motion using svd. *Technical notes* (2009).

[ZHS*05] ZHOU K., HUANG J., SNYDER J., LIU X., BAO H., GUO B., SHUM H.-Y.: Large mesh deformation using the volumetric graph laplacian. *ACM Trans. Graph. 24*, 3 (2005), 496–503.

[ZSS97] ZORIN D., SCHRÖDER P., SWELDENS W.: Interactive multiresolution mesh editing. In *SIGGRAPH '97* (1997), pp. 259–268.