Journal of Korea Multimedia Society Vol. 22, No. 4, April 2019(pp. 491-501)
https://doi.org/10.9717/kmms.2019.22.4.491

Real-time Shape Manipulation using Deformable Curve-Skeleton

Eisung Sohn[†]

ABSTRACT

Variational methods, which cast deformation as an energy-minimization problem, are known to provide a good trade-off between practicality and speed. However, the time required to deform a fully detailed shape means that these methods are largely unsuitable for real-time applications. We simplify a 2D shape into a curve skeleton, which can be deformed much more rapidly than the original shape. The curve skeleton also provides a simplified control for the user, utilizing a small number of control handles. Our system deforms the curve skeleton using an energy-minimization method and then applies the resulting deformation to the original shape using linear blend skinning. This approach effectively reduces the size of the variational optimization problem while producing deformations of a similar quality to those obtained from full-scale nonlinear variational methods.

Key words: Shape Manipulation, Shape Deformation, Deformable Curve-skeleton, Interactive Animation Interface

1. INTRODUCTION

Two-dimensional (2D) shape deformation has been used for various computer graphics applications, including image editing, games, and animation. While there are many physically accurate methods that require lengthy computation, a plausible result is usually sufficient for most artistic or interactive applications in which real-time performance and intuitive control are paramount.

Considerations of complexity and practicality have motivated many recent studies. Variational methods [1, 2, 3], in which deformation is performed by solving a global quadratic variational minimization problem, have been successful. The key idea is to minimize the distortion energy of the local geometry so that the global shape remains as rigid as possible. By implementing control handles as constraints, it is easy to produce realistic-looking deformations. However, because the runtime of the optimization depends heavily on the size of the meshes, this approach is unsuitable for high-resolution objects in real time.

Linear blend skinning (LBS) is a fast deformation method widely supported by modern graphics hardware. It is a straightforward process in which each vertex is transformed by a linearly weighted combination of affine transformations of the control handles. The recent introduction of bounded biharmonic weights [4] has drawn attention back to the versatility of LBS deformation. Because it automatically generates weights that produce a smooth deformation from control handles, this new incarnation of LBS can smoothly deform objects of arbitrary topology. However, there is a control issue: The user must explicitly specify the transformations of all the control handles. To address this problem, Jacobson et al. introduced a pseu-

※ Corresponding Author : Eisung Sohn, Address: (21983) Room 425, Libertas Hall A, Yonsei University, 85 Songdogwahak-ro, Yeonsu-gu, Incheon, Korea, TEL : +82-32-749-3054, E-mail : esohn@yonsei.ac.kr
Receipt date : Dec. 28, 2018, Revision date : Mar. 8, 2019
Approval date : Mar. 26, 2019
[†] University College, Yonsei University
※ This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (2019R1G1A1010839).

do-edge, which automatically infers the rotation of a control handle. They have also shown how to determine the skinning transformations from a sparse set of controls [5]. Unfortunately, the results of this linear deformation are inferior to those produced by nonlinear variational methods.

We introduce an intermediate deformable structure, called a deformable curve skeleton (DCS), with the aim of achieving high-quality deformations in real time together with control that is as intuitive as possible. A curve skeleton is a valid simplification of a shape and also a structure that is easy to deform. A non-linear variational optimization can be speedily applied to a curve skeleton using the fast LBS method to obtain smooth and seamless non-linear deformations at a low cost.

Our approach performs deformations in two steps: First, the curve skeleton is deformed under the given constraints using a non-linear variational method, and then the deformation is applied to the original shape using LBS. Compared to our previous approach which determines the deformation in a single step, the two step approach produces significantly better results[25].

Our examples demonstrate that deformations produced in this way are very similar to the results of full-scale nonlinear variational methods, but reducing the problem size means that our pose-time optimization is an order of magnitude faster than non-linear variational methods and comparable to simple LBS methods. Also, controlling a DCS is easier than handling a full shape, as the curve skeleton is a more abstracted structure that, in effect, infers missing degrees of freedom and incorporates inverse kinematics. We show that our method can run on a mobile device, on which computational efficiency and concise interaction are crucial.

2. PREVIOUS WORK

Various deformation methods have been presented over several decades. We focus on two main categories: methods that deform shapes using a weighted blend of control handles and those that deform the shapes using their structure.

The best-known technique in the first category is the use of bones of a skeleton as control handles. Skeleton-based deformations [6, 7, 8, 9] are good for articulated objects with rigid limbs, such as humans and animals. To achieve a smooth deformation, every part of shape has to be carefully assigned to corresponding bones in the skeleton, with appropriate weights.

Cage-based deformations [10] use an enclosing structure to control deformation. The user moves the vertices of the cage to induce a deformation of the original shape. Some methods use points as deformation handles [11, 12]. Points are quick and easy to manipulate but do not necessarily provide intuitive control of complex deformations.

A unified method, combining all three handle types points, bones and cages have been presented [4], within an automatic weighting scheme which allows users to choose the most appropriate handles of a shape. However, the tedious job of specifying the functions of all the control handles remains. This issue has been addressed by an automatic inference method [5], in which a user only has to specify a subset of the degrees of freedom, and the rest is automatically inferred by using rigidity energies.

The second category includes physically-based simulations [13] and the finite element method (FEM) [14]. Methods of this sort provide a more plausible result but require a lot of computation, and are therefore unsuitable for most interactive applications. However, variational methods involve the minimization of a shape deformation energy and are capable of producing physically plausible deformations at interactive speed. The efficiency of some methods can be improved by the linearization of its nonlinear constraints using various approximations [1, 15]. However, the linear approximation is bound to produce a questionable deformation.

Some methods solve nonlinear optimization problems directly [2, 3, 16, 17, 18]. One approach [2] aims to preserve two local shape properties: the Laplacian coordinates of the boundary curve as well as the local area of the shape-interior, which are together represented as a non-quadratic energy function. The lack of robustness and stability in this approach has been addressed by an improved algorithm which uses a simple triangular mesh [3]. However, a limitation inherent in these variational methods is that they rely on optimization at pose time. The tractability of this optimization is highly dependent on the mesh size, and so the computation time grows exponentially with mesh size.

Various methods of extracting curve-skeletons from 2D and 3D objects have been presented [19]. We have found a Laplacian-based contraction technique [20, 21] to be effective in generating a unique, smooth curve-skeleton from a 3D object. We extrude a 2D polygonal shape into a 3D object to make it eligible for Laplacian-based contraction.

3. OVERVIEW

The input shape can be either a bitmap or vector graphics. A bitmap must be converted into a polygonal mesh. We extract a closed polygon from the boundary of the shape defined by the transparent pixels, and then we triangulate it using Shewchuk's method [22].

High-quality curve-skeletons should exhibit many properties: such as homotopy, invariant under isometric transformations, thinness, centeredness, and smoothness. However, they are not essential in the curve-skeleton used by our method, which can even work satisfactorily with a curve-skeleton drawn directly by the user. Nevertheless, poorly constructed curve-skeleton may lead to poor deformation. So we want to construct curve-skeletons that are as concise and smooth as possible.

We found that Laplacian-based mesh contraction [20] gives satisfactory results for our purpose. This technique contracts a 3D mesh geometry into a zero-volume skeletal shape by applying implicit Laplacian smoothing with global positional constraints. It operates on 3D meshes, and therefore, we need to extrude the flat 2D mesh along the z-axis and, as shown in Fig. 1.

The resulting 3D mesh has appropriate curvature-flow normals and is for Laplacian smoothing.

A curve-skeleton is represented as the edges of a 2D graph. We re-sample these edges at regular intervals so that they all have a similar length. We choose this resolution heuristically: the quality of deformation is not critically dependent on this resolution, as shown in Fig. 6.

The user locates control handles on the curve-skeleton to deform it. A control handle provides weighted positional and rotational constraints. The user can modify weights freely to control the curve-skeleton. When the user moves the control handles, the deformation algorithm minimizes the

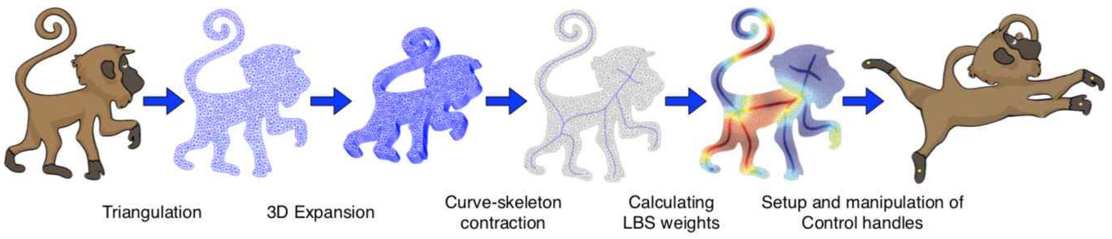
Triangulation
3D Expansion
Curve-skeleton contraction
Calculating LBS weights
Setup and manipulation of Control handles

Fig. 1. Overview of the deformation process. We first triangulate the original shape, and then expand it to a 3D mesh, from which we obtain a curve-skeleton by Laplacian contraction. Then we calculate the LBS weights to bind the curve-skeleton to the original shape. Finally we deform the shape by setting up and manipulating control handles.

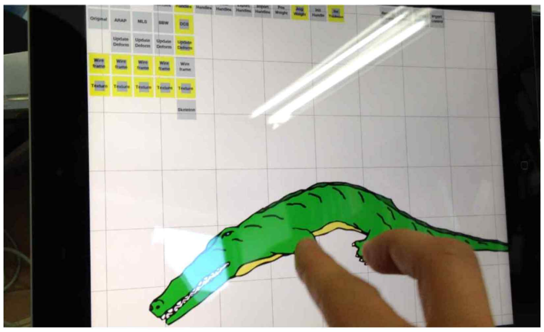

Fig. 2. An interactive animation running on an Apple iPad. The combination of speed and high quality make our method suitable for mobile devices.

distortion of the curve-skeleton caused by the changed constraints.

Each time the pose is updated, the original shape is deformed by a linear blend skinning method to follow the deformed curve-skeleton. The user can designate the LBS weights manually, but our method includes an implementation of bounded biharmonic weights [4] to generate smooth weights automatically before the binding stage, by minimizing the Laplacian energy subject to bound constraints.

4. DEFORMABLE CURVE-SKELETON

We aim to minimize the overall distortion over the curve-skeleton caused by given constraints. Several different kinds of distortion contribute to the elastic energy stored in a deformed material: they include stretching, bending, shearing, and twisting. We can simplify the elastic energy model by considering only the dominant distortions of curves, which are bending, stretching and contracting. Those distortions are related to the curvature and length of a curve. Our algorithm finds the deformation which best preserves two local properties of a curve-skeleton: the edge length and the Laplacian coordinates of the curves.

Variations in the length of a curve correspond to the amount of stretching and contraction, and variations in its Laplacian coordinates correlate with the amount of bending. In the following sections, we describe these two local properties.

4.1 Edge lengths

A given curve-skeleton is represented as a graph (V, E) in R^2, where V is a set of n vertices and E is a set of m edges.

We control changes in the lengths of edges by minimizing the following quadratic energy function:

$$\sum_{(i,j)\in\mathbf{E}} \|(\mathbf{v}_i - \mathbf{v}_j) - e(\mathbf{v}_i, \mathbf{v}_j)\|^2, \tag{1}$$

where $e(\mathbf{v}_i, \mathbf{v}_j) = \frac{\mathbf{e}^0_{i,j}}{\mathbf{e}_{i,j}}(\mathbf{v}_i - \mathbf{v}_j)$, the original length of edge (i, j) is $\mathbf{e}^0_{i,j}$, and $\mathbf{e}_{i,j}$ is its current length. We can rewrite this equation in matrix form:

$$\|\mathbf{HV} - e(\mathbf{V})\|^2, \tag{2}$$

where H is an $m \times n$ matrix:

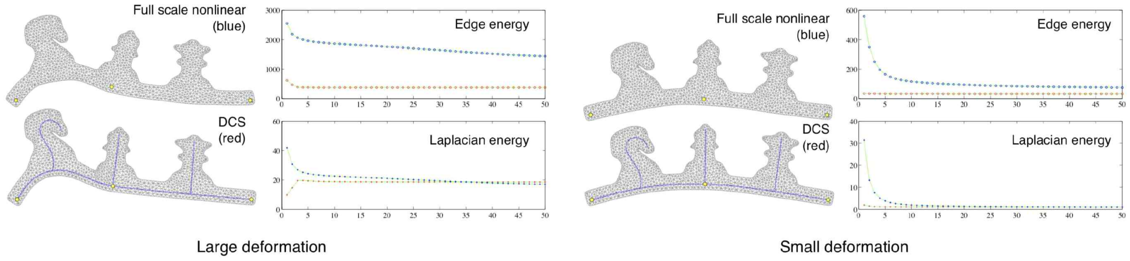

Fig. 3. Convergence of deformation energy following sudden large and small displacements of a control handle. The full-scale nonlinear method attempts to optimize all the triangles in the mesh, while the DCS method operates on the curve-skeleton. The DCS method converges faster in both cases. The stiffness weight α=3.

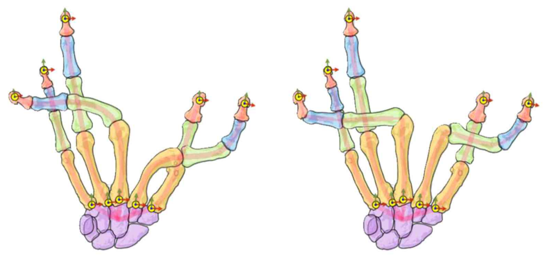
(a) Static Laplacian weights
(b) Variational Laplacian weights

Fig. 4. Static Laplician weights produce a continuous deformation of a curve-skeleton; but variational Laplacian weights allow it to retain the appearance of a jointed assembly of rigid elements.

$$\mathbf{H} = \begin{cases} 1 & \text{for} \quad i \mid (i,j) \in \mathbf{E} \\ -1 & \text{for} \quad j \mid (i,j) \in \mathbf{E} \\ 0 & \text{for} \quad \text{otherwise.} \end{cases} \tag{3}$$

4.2 Laplacian coordinates

The Laplacian coordinate δ_i of each vertex v_i in V is the weighted difference between v_i and its neighboring vertices:

$$\delta_i = \mathcal{L}(\mathbf{v}_i) = \sum_{\{i,j\}\in\mathbf{E}} w_{i,j}(\mathbf{v}_j - \mathbf{v}_i) = \left[\sum_{\{i,j\}\in E} w_{i,j}\mathbf{v}_j\right] - \mathbf{v}_i, \tag{4}$$

where $\sum_{\{i,j\}\in E} w_{i,j} = 1$, and $\mathcal{L}$ is the Laplace operator for graphs.

The weights $w_{i,j}$ are determined as follows:

$$w_{i,j} = \frac{\omega_{ij}}{\sum_{\{i,k\}\in E} \omega_{ik}}, \tag{5}$$

where we use $w_{ij} = 1$ as the uniform weights.

We penalize changes of shape during deformation by minimizing the following energy function:

$$\sum_{i\in V} \|\mathcal{L}(\mathbf{v}_i) - \delta_i\|^2. \tag{6}$$

The deformed Laplacian coordinates δ_i are obtained as follows:

$$\delta_i = \mathbf{R}_i \delta_i^0 \tag{7}$$

where δ_i^0 is the Laplacian coordinate in the rest pose (4), and R_i is the rotation matrix for the i th vertex and depends on its neighbors. We will describe the derivation of R_i from the vertices of the deformed shape in Section 4.4.

The matrix form of (6) is:

$$\|\mathbf{LV} - \Delta\|^2, \tag{8}$$

where $\triangle$ represents the rotated initial Laplacian coordinates, and L is an n×n Laplacian matrix L, with the following elements:

$$\mathbf{L} = \begin{cases} -1 & \text{for} \quad i = j \\ w_{ij} & \text{for} \quad (i,j) \in \mathbf{E} \\ 0 & \text{for} \quad \text{otherwise} \end{cases} \tag{9}$$

4.3 Deformation energy

To control the deformation, we add a term $\| I_p V - P \|$ that expresses the position constraints, where I_p is the index matrix of V, so that I_p maps all the elements of V to P, which contains their target positions. Similarly, we add a term $\| I_e V - Q \|$ for the rotational constrains, where I_e is a matrix of indices to the edges, and Q contains the target vectors.

The deformed positions of the vertices V are then obtained by minimizing the following quadratic expression for energy:

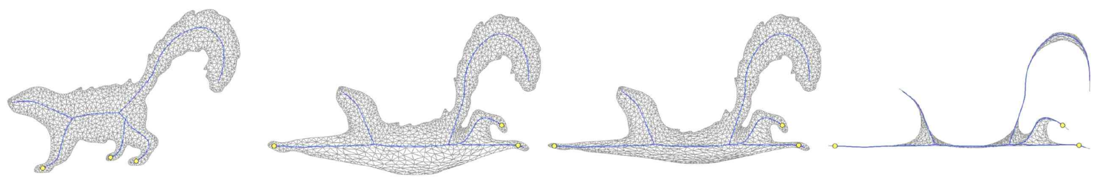

Fig. 5. We use a local thickness value along the curve-skeleton for approximate area preservation. Each vertex position is determined by this thickness ratio. However, areas where the weights are conflicting are not completely covered by this scheme when the thickness values are set to 0, as shown in (d).

$$\|\mathbf{HV}-e(\mathbf{V})\|^2+\alpha\|\mathbf{LV}-\delta(\mathbf{V})\|^2+ w_p\|\mathbf{I_p V}-\mathbf{P}\|^2+w_e\|\mathbf{I_e V}-\mathbf{Q}\|^2. \quad (10)$$

To achieve a balance between the preservation of edge lengths and Laplacian coordinates, while meeting the constraints, we apply weights to each term: the stiffness weight α is applied to the Laplacian coordinates, and w_p and w_e are global weights applied to the positional and rotational constraints respectively. Using a constant α produces a continuous deformation of the entire curve-skeleton, taking no account of its joints; whereas a variational Laplacian weight retains the appearance of an articulated structure of rigid elements, as shown in Fig. 4.

The energy minimization problem is thus reformulated as follows:

$$\min\|\mathbf{AV}-b(\mathbf{V})\|^2, \quad (11)$$

where

$$\mathbf{A}=\begin{pmatrix}\mathbf{H}\\ \alpha\mathbf{L}\\ w_p\mathbf{I_p}\\ w_e\mathbf{I_e}\end{pmatrix},\quad b(\mathbf{V})=\begin{pmatrix}e(\mathbf{V})\\ \alpha\delta(\mathbf{V})\\ w_p\mathbf{P}\\ w_e\mathbf{W}\end{pmatrix}. \quad (12)$$

This is a nonlinear least-squares problem, since b is dependent on the current vertex positions V, while the matrix A is only dependent on the initial configuration. We solve this problem using an iterative Gauss-Newton method [23].

4.4 Nonlinear least-squares optimization

To allow us to use the iterative Gauss-Newton method, we have to rewrite Equation (11) as follows:

$$\min_{V^{k+1}}\|\mathbf{AV}^{k+1}-b(\mathbf{V}^k)\|^2, \quad (13)$$

where V^k are the vertex coordinates obtained from the k^{th} iteration, and V^{k+1} are the target coordinates for the next iteration. Since we already know $b(V^k)$, this equation represents a standard linear least-squares problem and its solution:

$$\mathbf{V}^{k+1}=(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}(\mathbf{V}^k). \quad (14)$$

Since A is only dependent on the initial configuration, $(A^TA)^{-1}A^T$ can be precomputed.

During each iteration, we calculate b from the vertex coordinates V^k obtained in the previous iteration. Therefore we only need to compute $\delta(V^k)$ and $e(V^k)$ at each iteration. The term $e(V^k)$ can be obtained as follows:

$$e(v_i^k,v_j^k)=\frac{\widetilde{l_{i,j}}}{|v_i^k-v_j^k|}(v_i^k-v_j^k),\ for\ (i,j)\in\mathbf{E}. \quad (15)$$

To compute the Laplacian coordinates $\delta(V^k)$, we need the initial Laplacian coordinates. But the difference between these two sets of coordinates takes no account of rotation whereas we want the energy term to be zero if the object simply rotates without any distortion; but, this term is non-zero if we compute it directly. Therefore we explicitly rotate each vertex $v_i\in\mathbf{V}$: $\delta(v_i^k)=\mathbf{R}_i^k\,\delta(v_i^0)$, where $\delta(v_i^0)$ is the Laplacian coordinate before deformation, and the transformation matrix $\mathbf{R}_i^k$ is obtained by minimizing the following energy expression:

$$\mathbf{R}_i^k=\arg\min_{\mathbf{R}_i^k}\sum_{j\in\mathcal{N}(i)}\|\mathbf{R}_i^k(v_j^0-v_i^0)-(v_j^k-v_i^k)\|^2, \quad (16)$$

where $N(i)$ is the set of vertices connected to

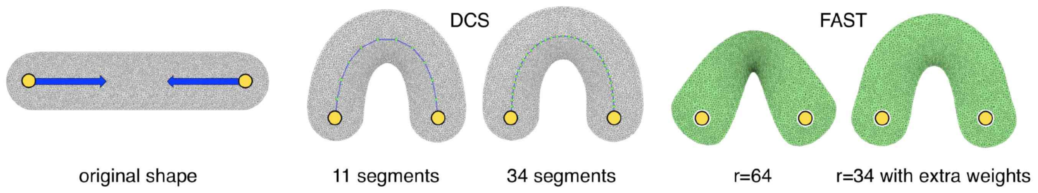

Fig. 6. Comparison with the F.A.S.T. algorithm [5]. Results from our method are not much affected by the number of segments in a curve-skeleton. Results from both methods are satisfactory, but F.A.S.T requires additional weights to compensate the shrink artifact.

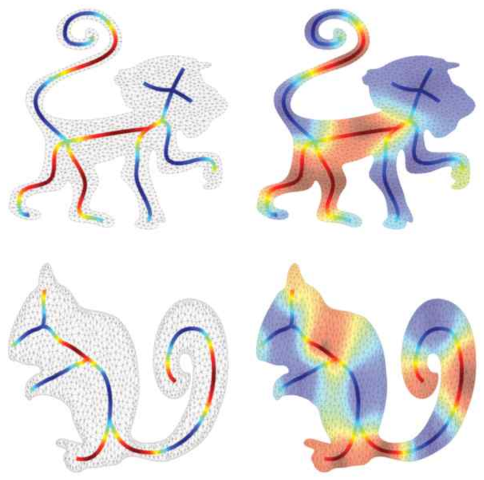
Fig. 7. LBS weights created automatically for a curve-skeleton using bounded biharmonic weights.

vertex i. Then the covariance matrix S_i can be expressed as follows:

$$\mathbf{S}_i = \sum_{j\in\mathcal{N}(i)} (v_j^0 - v_i^0)(v_j^k - v_i^k)^T. \tag{17}$$

We can obtain R_i from the singular value decomposition of $S_i = U_i\Sigma_i V_i$ [24]:

$$\mathbf{R}_i = \mathbf{V}_i\mathbf{U}_i^T. \tag{18}$$

5. DEFORMING THE FULL SHAPE

We now describe how we bound the curve-skeleton to the full shape, using linear blend skinning, also known as skeletal subspace deformation. Despite its well-known shortcomings, linear blend skinning is the most popular technique for calculating skin deformation due to its simplicity and computational efficiency. Since a curve-skeleton is composed of many segments, it can be seen as an extended form of a skeleton, but without the hierarchy. A curve-skeleton is bound to the full shape by assigning a correspondence weights to vertices on each curve. Setting appropriate weights is essential for deformation quality; manual weight painting can be expected to the best-looking result. However, to generate LBS weights automatically, we use bounded biharmonic weighting, which combines well with our approach and produces excellent results [4]. Fig. 7 shows the weights gen-

Table I. Performance of our algorithm on the examples presented in this paper

Model	Numbers of vertices	Numbers of triangles	Numbers of curve-skelton edges	DCS		Full-scale nonlinear	
				Runtime (ms)	Precomp. (ms)	Runtime (ms)	Precomp. (sec.)
Monkey	1344	2275	138	0.078	2.73	4.15	4.94
Polar bears	1406	2484	68	0.045	1.34	3.98	4.32
Chipmunk	1496	2492	84	0.044	0.91	4.99	5.69
Skunk	1357	2216	76	0.038	1.27	4.17	4.77
Duck	1277	2234	51	0.034	0.29	3.70	3.35
Alligator	1426	2430	73	0.042	1.33	5.68	6.25
Chess	1459	2439	84	0.041	0.80	5.71	6.42
Bar	1181	2099	42	0.022	0.26	3.22	3.48
Hand	1980	3366	96	0.03	1.133	22.53	46.02

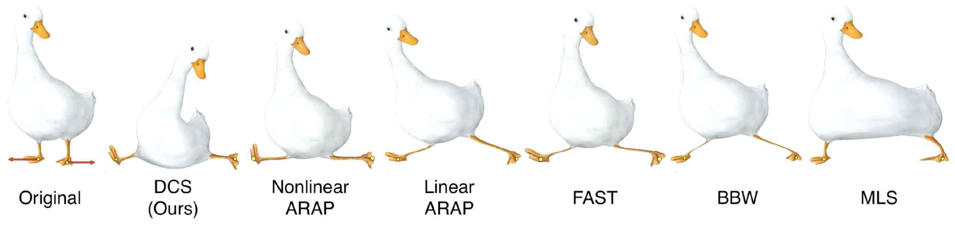

Fig. 8. Our method achieves more realistic behavior than methods without structural meaning. Note the unrealistic legs in the deformed goose model produced by other methods.

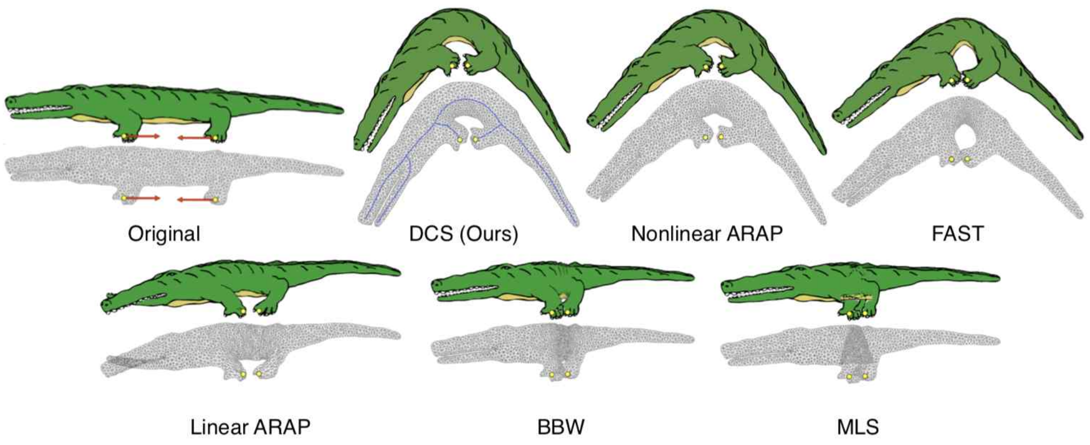

Fig. 9. Our method achieves a similar visual quality to the full nonlinear method (Nonlinear ARAP) while it runs as fast as other linear methods.

erated for monkey and chipmunk shapes using this method.

Area preservation is an essential factor in 2D shape manipulation.

It can be an overhead [2], and some deformation methods are unable to support it [1, 12]. The curve-skeleton offers approximate area preservation because of its local thickness property. Each edge of the curve-skeleton has its own thickness value, with an initial value of 1 in the rest state. During deformation, the lengths of edges change, and the system updates thickness of each edge using the formula $H^k = l^0 / l^k$, where l^0 is the edge's initial length and l^k is the its current length, so that the product of length and thickness remains constant. However, the areas of some regions of the model influenced by curves with low weights are not well preserved, as shown in Fig. 5: some joint areas do not shrink away even when the thickness value is zero. Although accurate area preservation is not achieved, we obtain a satisfactory approximation. Accounting for thickness, the positions of all the vertices are now determined by blending the edge influences as follows:

$$\mathbf{v}_i = \sum_{j \in \mathbf{E}} w_{ij} \mathbf{T}_j \mathbf{d}_{ij}, \tag{19}$$

where w_{ij} are the bounded biharmonic weights, d_{ij} is the local distance vector for vertex $\mathrm{V}i$ from edge e_j, and T_j is the transformation matrix of that edge. We obtain d_{ij} as follows:

$$\mathbf{d}_{ij} = (\mathbf{d}_x + h^k \mathbf{d}_y + \mathbf{d}_z)_{ij}.$$

6. RESULTS

We compared our deformation method to several previous techniques on various drawings and images. We precomputed the curve skeleton and LBS weight for the triangle mesh from each source image. Table 1 summarizes the characteristics of the models and the performance of the current unoptimized implementation running on a laptop computer with a 2.0 GHz Intel Core i7-3667U CPU and 8 GB of memory. We used the Eigen library to solve the sparse linear system and perform the singular value decomposition. We performed 15 iterations, each taking about 0.02 to 0.08 ms, resulting in total computation times of 0.5 to 1 ms, which is comparable to simple LBS methods.

Fig. 8 and 9 compare the quality of deformation achieved by several linear and nonlinear deformation methods. Nonlinear variational methods can be expected to produce more plausible and detailed results than linear methods. However, nonlinear methods are generally slow and prone to instability and slow convergence. As shown in Fig. 3, our curve-based approach is less affected by those problems; it converges faster and is more stable than full-scale nonlinear methods while producing deformations of the quality associated with non-linear techniques.

Apart from LBS methods, precomputation is essential to reduce the pose-time computation of variational methods. However, as shown in Table 1, our method requires relatively little precomputation due to the reduced problem size. This is important in interactive applications, as precomputation is required whenever the configuration of control handles or weights is changed.

The behavior of the parts of a curve skeleton that are not specified by the user is automatically determined. Thus, we can eliminate all the remaining degrees of freedom without the need to solve an inverse kinematic problem. A state-of-the-art deformation method [5] also supports inverse kinematics by optimizing the remaining degrees of freedom of the unspecified handles. However, our result achieves higher quality and smooth deformations due to the dense resolution (Fig. 6).

Users need to choose the appropriate type of control handle for the deformation that they require. Typical handles are points, bones, and cages, but these types of handle support only low-level control, whereas our curve-skeleton concept offers a more abstract level of control. It does not require users to position every part of a shape explicitly but allows them to specify a loose combination of significant influences. This intuitive approach permits a reduction in the number of control handles.

We implemented ourdeformation method on the Apple iPad, as shown in Fig. 2, to assess itscontrollability in a practical setting. The accompanying video demonstratesthat a user can efficiently produce realistic deformations using multi-touchinteraction.

7. CONCLUSIONS AND FUTURE WORK

We have presented a 2D shape-deformation method based on a deformable curve skeleton, which provides an intuitive and flexible method of control. This abstracted structure allows us to approach the deformation quality of full-scale non-linear variational methods of deformation at a much lower computational cost.

Like most other 2D deformation methods, our method may suffer from problems of overlapping. We currently deal with this issue by manually painting depth values. A method of dynamic depth adjustment would enhance interactively produced deformations.

Also, like the other nonlinear variational methods, our method requires some iterations to converge. Even though each iteration is quite short, certain conditions can extend the convergence time; in particular, this occurs with deformations that involve mostly Laplacian energy. We currently use a fixed number of iterations to guarantee performance. Fifteen iterations were used in every example in this paper, which usually produces satisfactory results without convergence problems. However, the accompanying video shows that incomplete convergence is sometimes preferable, as it can produce smooth in-betweening and natural follow-up deformations.

Extending our approach to 3D models is an obvious direction for future work. Because a curve skeleton can be extracted from 3D surface meshes or point clouds [21], many 3D models could be deformed without manual rigging. Extension to 3D case raises stability and controllability concerns, but early experiments show promising results, and we plan to explore 3D further.

REFERENCE

[1] T. Igarashi, T. Moscovich, and J.F. Hughes, "As-rigid-as-possible Shape Manipulation," *ACM Transactions on Graphics*, Vol. 24, No.

3, pp. 1134-1141, 2005.

[2] Y. Weng, W. Xu, Y. Wu, K. Zhou, and B. Guo, "2D Shape Deformation Using Nonlinear Least Squares Optimization," *The Visual Computer*, Vol. 22, No. 9, pp. 653-660, 2006.

[3] H. Guo, X. Fu, F. Chen, H. Yang, Y. Wang, and H. Li, "As-rigid-as-possible Shape Deformation and Interpolation," *Journal of Visual Communication and Image Representation*, Vol. 19, No. 4, pp. 245-255, 2008.

[4] A. Jacobson, I. Baran, J. Popovi´c, and O. Sorkine, "Bounded Biharmonic Weights for Real-time Deformation," *ACM Transactions on Graphics*, Vol. 30, No. 4, pp. 1, 2011.

[5] A. Jacobson, I. Baran, L. Kavan, J. Popovi´c, and O. Sorkine, "Fast Automatic Skinning Transformations," *ACM Transactions on Graphics*, Vol. 31, No. 4, pp. 1-10, 2012.

[6] J.P. Lewis, M. Cordner, and N. Fong, "Pose space Deformation: A Unified Approach to Shape Interpolation and Skeleton-driven Deformation," *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 165-172, 2000.

[7] S. Yoshizawa, A.G. Belyaev, and H.-P. Seidel, "Free-form Skeleton-driven Mesh Deformations," *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, pp. 247-253, 2003.

[8] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "FiberMesh," *ACM Transactions on Graphics*, Vol. 26, No. 3. pp. 41, 2007.

[9] S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popovi´c, "Interactive Skeleton-driven Dynamic Deformations," *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 586-593, 2002.

[10] T. Ju, S. Schaefer, and J. Warren, "Mean Value Coordinates for Closed Triangular Meshes," *ACM Transactions on Graphics*, Vol. 24, No. 3, pp. 561, 2005.

[11] F. Bookstein, "Principal Warps: Thin-plate Splines and the Decomposition of Deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 6, pp. 567-585, 1989.

[12] S. Schaefer, T. McPhail, and J. Warren, "Image Deformation Using Moving Least Squares," *ACM Transactions on Graphics*, Vol. 25, No. 3, pp. 533-540, 2006.

[13] P. Moore and D. Molloy, "A Survey of Computer-based Deformable Models," *IEEE International Machine Vision and Image Processing Conference*, pp. 55-66, 2007.

[14] G. Celniker and D. Gossard, "Deformable Curve and Surface Finite-elements for Free-form Shape Design," *ACM Siggraph Computer Graphics*, Vol. 25, No. 4, pp. 257-266, 1991.

[15] M. Botsch and O. Sorkine, "On Linear Variational Surface Deformation Methods," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 14, No. 1, pp. 213-230, 2008.

[16] M. Botsch, M. Pauly, M., and L. Kobbelt, "PriMo: Coupled Intuitive Surface Modeling," *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, pp. 11-20, 2006.

[17] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler, "A Local/global Approach to Mesh Parameterization," *Proceedings of the Eurographics Symposium on Geometry Processing*, Vol. 27, No. 5, pp. 1495-1504, 2008.

[18] I. Chao, U. Pinkall, P. Sanan, and P. Schro¨der, "A Simple Geometric Model for Elastic Deformations," *ACM Transactions on Graphics*, Vol. 29, No. 4, pp. 1-6, 2010.

[19] N.D. Cornea, D. Silver, and P. Min, "Curve-skeleton Properties, Applications, and Algorithms," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 13, No. 3, pp. 530-548, 2007.

[20] O.K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee, "Skeleton Extraction by

Mesh Contraction," *ACM Transactions on Graphics*, Vol. 27, No. 3, pp. 1-10, 2008.

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

[22] J.R. Shewchuk, "Delaunay Refinement Algorithms for Triangular Mesh Generation," *Computational Geometry: Theory and Applications*, Vol. 22, No. 1-3, pp. 21-74, 2002.

[23] K. Madsen, H. Nielsen, and O. Tingleff, *Methods for Nonlinear Least Squares Problems*, *Informatics and Mathematical Modelling*, Technical University of Denmark, Technical Report, 2004.

[24] O. Sorkine and M. Alexa, "As-rigid-as-possible Surface Modeling," *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, pp. 109-116, 2007.

[25] E. Sohn and Y. Choy, "Image Deformation Using Freeform Deformation Axis," *Journal of Korea Multimedia Society*, Vol. 17, No. 10, pp. 1229-1238, 2014.

Eisung Sohn

is an assistant professor of University College at Yonsei University. His research interests include computer graphics, human-computer interaction, and deep learning. He has a PhD in computer science from Yonsei University.