Morphing with Laplacian Coordinates and Spatial-Temporal Texture

Hongbo Fu, Chiew-Lan Tai, Oscar Kin-Chung Au
Department of Computer Science
Hong Kong University of Science and Technology

1. Introduction

In this paper, we present a novel method to address the trajectory problem in 2D shape morphing based on the Laplacian coordinates (LCs). Although the LCs capture the geometric details of a shape, they are built in the global coordinate system and thus not rotation invariant [1, 7]. By adding rotation coherence constraints to as-rigid-as-possible transformations applied on the LCs, we make the intermediate morphing shapes highly appealing. Our method successfully alleviates local self-intersections.

We also propose to interpolate the textures bounded by simple closed curves using a spatial-temporal structure. In previous texture morphing techniques, textures are usually encoded using triangulations [2]. Therefore, the morphing results depend on the quality of these triangulations. Given two simple closed curves and their interpolated shapes in the time domain, we construct uniformly distributed streamlines, which build a one-to-one mapping between the source and target textures, instead of encoding the textures using triangulation. The resulting mapping makes neighboring pixels morph coherently.

2. Coherent curve interpolation

We describe our curve morphing method in this section. Without loss of generality, we assume that the input of our algorithm is two polygonal curves $C(i) = (\mathbf{v}_1^i, \mathbf{v}_2^i, \ldots, \mathbf{v}_n^i)$, $i = 0, 1$, with one-to-one correspondence specified (i.e. $\mathbf{v}_j^0$ corresponds to $\mathbf{v}_j^1$, $1 \leq j \leq n$). The objective is to find an appropriate way to transform and then interpolate the corresponding LCs of $C(0)$ and $C(1)$. The intermediate morphing shapes $C(t)$ $(0 < t < 1)$ are then reconstructed from the interpolated LCs [1].

A straightforward way to orient the LCs is to linearly interpolate the transformations between the local frames defined at the corresponding vertices. However, the shearing part of the interpolated transformations will make the LCs deviate from the normals, causing undesirable artifacts. For triangle-to-triangle morphing, Alexa et al. [2] proposed an as-rigid-as-possible shape interpolation method by first decomposing an affine transformation $\mathbf{A}$ into a rotation matrix $\mathbf{R}$ and a symmetric matrix $\mathbf{S}$ (i.e. $\mathbf{A} = \mathbf{RS}$), and then interpolating them individually [2]:

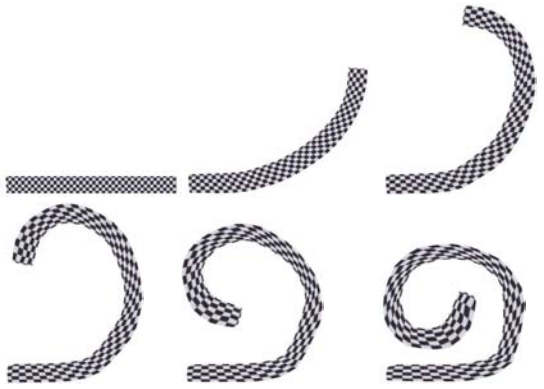
Figure 1: Morphing between a stick-like shape (top-left) and a swirl-like shape (bottom-right).

$$\mathbf{A}(t) = \mathbf{R}(t)((1-t)\mathbf{I} + t\mathbf{S}), \qquad t \in [0,1], \qquad (1)$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{R}(t)$ is a rotation matrix derived by linearly interpolating the rotation angles of $\mathbf{R}$. Xu et al. [9] used a similar idea to interpolate the gradient field for mesh morphing. Our framework also uses this as-rigid-as-possible transformation interpolation. In addition, to handle swirl-like shape morphing (Figure 1), we propose to interpolate coherent rotation angles.

We solve for the transformation $\mathbf{A}_j^{01}$ from the local frame at $\mathbf{v}_j^0$ to the local frame at $\mathbf{v}_j^1$ in the least squares sense (mapping the 1-ring structure and normal at $\mathbf{v}_j^0$ to the counterparts at $\mathbf{v}_j^1$). Similarly, we define the transformation $\mathbf{A}_j^{10}$ from $\mathbf{v}_j^1$ to $\mathbf{v}_j^0$. To get symmetry morphing, we define the interpolated LC at time t as

$$\delta_j(t) = (1-t)\mathbf{A}_j^{01}(t)\delta_j^0 + t\mathbf{A}_j^{10}(1-t)\delta_j^1, \qquad (2)$$

where δ_j^i $(= \mathbf{v}_j^i - 0.5\mathbf{v}_{j-1}^i - 0.5\mathbf{v}_{j+1}^i)$ is the LC defined at $\mathbf{v}_j^i$, $i = 0, 1$.

Computing the rotation angle α_j from the rotation matrix decomposed from $\mathbf{A}_j^{01}$ or $\mathbf{A}_j^{10}$ needs special attention, as α_j and $\alpha_j + 2\pi$ correspond to the same rotation matrix. Simply computing α_j in $[0, 2\pi)$ will produce undesirable morphing artifacts [3] (Figures 2 and 3 middle row). We observe that rotation angles for neighboring vertices must be coherent.

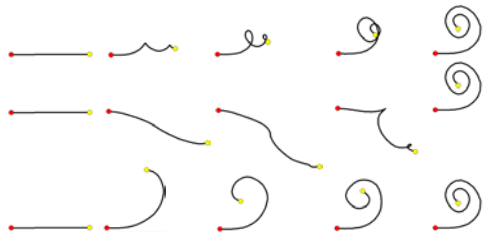

Figure 2: Morphing results for open curves.

We design a simple scheme to choose coherent rotation angles. We first choose an arbitrary vertex $\mathbf{v}_j$ and compute its rotation angle α_j in $[0, 2\pi)$, and then assign the rotation angle for the next vertex (counterclockwise or clockwise) such that the difference between the two rotation angles is within $[-\pi, \pi)$. This assignment process terminates when all the vertices are visited.

Figures 2 and 3 show some morphing examples of open/closed curves. The top rows are the results of linearly interpolating vertex positions. The middle rows show the results without using coherent rotation angles ($\alpha_j \in [0, 2\pi)$). The bottom rows illustrate the natural morphing results produced by our algorithm. Yellow circles indicate the key correspondences between two input curves and red circles constrain the reconstruction from the interpolated LCs [7]. Observe that with rotation coherence constraints, self-intersection problem can be alleviated.

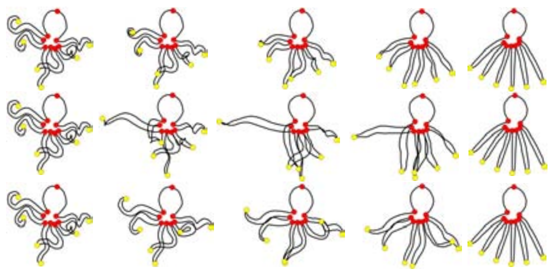

Figure 3: Morphing results for closed curves.

3. Spatial-temporal texture morphing

We define a *polygon* to be the region bounded by a simple, closed polygonal curve. Now we present a method to morph the textures within the polygons $P(t)$ produced by the curve morphing method in Section 2 (the textures in $P(0)$ and $P(1)$ are given as input). The underlying problem is to establish a one-to-one mapping between the polygons at different time points. A common technique is to build a compatible triangulation among these polygons [2]. However, it is computationally expensive and needs to introduce a lot of interior vertices.

To address the problem, we propose to build a streamline field within a spatial-temporal structure

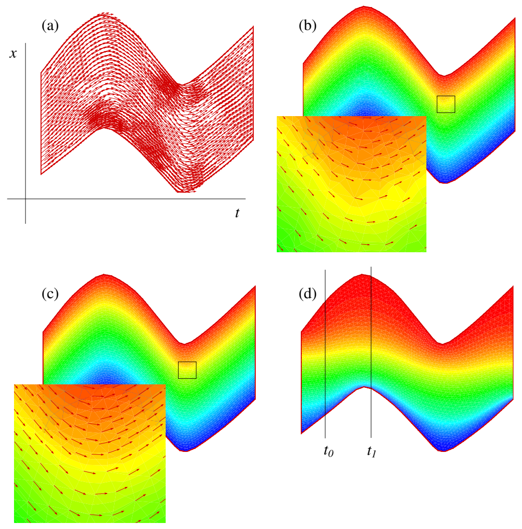

Figure 4: A 2D illustration of our texture mapping method. (a) The tangent field is constructed over a triangular mesh in 2D (corresponding to a tetrahedral mesh ST in 3D). (b) The correspondences at the left side of the triangular mesh are propagated to the right side along the tangent field by solving the transport equations in the least squares sense. The mesh is colored according to the computed correspondences. (c) The correspondences are smoothed by solving the transport equations locally. (d) The correspondence propagation using the method in [4]; the correspondence distributions at t_0 and t_1 are very different.

ST, which constitutes all the polygons $P(t)$ spanning the time domain. We want to find a streamline field such that for any point in the volumetric structure ST, there is a unique streamline passing through this point, starting at a point in $P(0)$ and ending at a point in $P(1)$. This basic requirement leads to a one-to-one mapping between two polygons at any two time points. However, there exists an infinite number of streamline fields satisfying this requirement. In practice, a uniformly distributed streamline field is preferred.

We construct a uniformly distributed streamline field in two steps, similar to the texture morphing method for implicit surfaces in [4]. In Step 1, we first build the structure ST by tetrahedralizing the boundary surface defined by the polygonal curves $C(t)$ sampled in the time domain and then assign a vector to each tetrahedral vertex (Figure 4 (a)). The vector field will serve as the tangent field of the streamlines to be contsructed in Step 2. To construct the vector field, we first build a tangent field over $C(t)$ by connecting the corresponding vertices of $C(t)$ at successive time points. We then propagate the tangent field at the boundary into the interior of ST. For each interior vertex $\mathbf{v}(t)$ of ST, its associated vector is assigned as $\sum_{j=1}^{n} \exp(-dist(\mathbf{v}(t), \mathbf{v}_j(t)))\mathbf{T}_j(t)$, where $\mathbf{v}_j(t)$ and $\mathbf{T}_j(t)$ are the vertices and their corresponding tangent vectors of $C(t)$ respectively, and $dist(\mathbf{v}(t), \mathbf{v}_j(t)))$ is the length of the shortest path lying within the interior of $C(t)$ from $\mathbf{v}(t)$ to $\mathbf{v}_j(t)$. Dinh et al. [4] used a gradient field of a harmonic field as the tangent field. This method cannot be applied in our context, as it will lead to non-uniformly distributed streamlines (Figure 4 (d)).

In Step 2, we build a streamline field from the tangent vector field. Like [4], we do not explicitly compute the streamlines using streamline integration methods, since the accumulated errors will make the integrated streamlines inconsistent with the streamlines at the boundary mesh of ST. Instead, we propagate the correspondence labels by solving a set of transport equations in the least squares sense [4]. The resulting labels do not change along their associated streamlines (to be computed). Therefore, when all the vertices have been assigned labels, for any point $\mathbf{p}$ within ST (not necessarily a tetrahedral vertex), we can get a streamline by connecting the points with the same label as $\mathbf{p}$. The label of the arbitrary point $\mathbf{p}$ is computed by barycentrically interpolating the labels of the four vertices of the tetrahedron in which $\mathbf{p}$ lies.

The optimal solution of the transport equations distributes the errors to all the vertices uniformly. However, these errors still lead to unsmooth streamlines (Figure 4 (b)). Thus we design a method to locally smooth the labels. For each originally unknown label (associated with an interior tetrahedral vertex), we locally re-solve the transport equation and iteratively update its label value (Figure 4 (c)). In all our experiments, we obtained smooth streamlines with 20 iterations. Note that we do not use the converged label values because they would be the same as those without smoothing. Since each iteration locally solves the transport equation, the result converges to the unique solution.

In Figure 1, we morph the texture defined in the source region to the target region. The texture changes smoothly thanks to the smooth streamlines. Figure 5 is another example. Since the tetrahedrons of ST are essentially used as finite elements to solve the transport equations, instead of explicitly encoding the textures [2], relatively coarse tetrahedral meshes are sufficient for creating smooth morphing results. However, the tetrahedralization step is still the performance bottleneck of the texture morphing method.

4. Discussions

The proposed methods have several limitations.

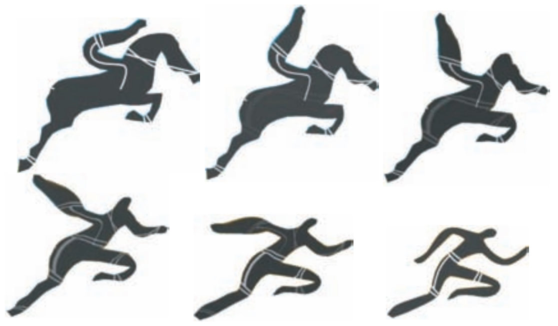
Figure 5: Texture morphing between a horse-rider and a runner.

Like other morphing techniques based on boundary representations [5, 6, 9], our method does not explicitly consider the interior regions while interpolating the boundary curves. To reduce distortion in the interior regions, more pairs of correspondence vertices have to be used to provide the boundary condition during the reconstruction of a shape from the interpolated LCs. In our current implementation, we simply linearly interpolate the constrained correspondence vertices. More complicated methods, like morphing each individual segment between two constrained correspondence vertices and then reconstructing the whole curve from the segments [8], may be used. Our texture morphing method involves tetrahedralizing a 3D boundary mesh. Therefore, the method is inapplicable when some interpolated curves have self-intersections, as the resulting 3D mesh cannot be tetrahedralized.

5. Reference

[1] M. Alexa, “Local Control for Mesh Morphing”, *Shape Modeling and Application 01*, pp. 209-215.

[2] M. Alexa, D. Cohen-Or and D. Levin, “As-Rigid-As-Possible Shape Interpolation”, Proceedings of ACM SIGGRAPH 2000, 2000, pp. 157–164.

[3] J. Choi and A. Szymczak, “On Coherent Rotation Angles for As-Rigid-As-Possible Shape Interpolation”, *CCCG*, 2003, pp. 111-114.

[4] H.Q. Dinh, A. Yezzi and G. Turk, “Texture Transfer During Shape Transformation”, *TR CS-2004-7, Dept. of Comp. Sci., Stevens Inst. of Tech.*.

[5] Y. Lipman, O. Sorkine, D. Levin and D. Cohen-Or, “Linear Rotation-Invariant Coordinates for Meshes”, *ACM Trans. Graph.*, 24, 2005.

[6] T. W. Sederberg, P. Gao, G. Wang and H. Mu, “2D Shape Blending: An Intrinsic Solution to the Vertex Path Problem”, *SIGGRAPH 1993*, 27, pp. 15–18.

[7] O. Sorkine, Y. Lipman, D. Cohen-Or, M. Alexa, C. Rössl and H.-P. Seidel, “Laplacian Surface Editing”, *SGP 2004*, pp. 179–188.

[8] T. Surazhsky and G. Elber, “Metamorphosis of Planar Parametric Curves Via Curvature Interpolation”, *Intl. J. of Shape Modeling*, 8(2): 2002, pp. 201–216.

[9] D. Xu, H. Zhang, Q. Wang and H. Bao, “Poisson Shape Interpolation”, *ACM Symposium on Solid and Physical Modeling*, 2005.