Tweening Boundary Curves of Non-simple Immersions of a Disk

Uddipan Mukherjee*
Department of Computer Science
University of California, Irvine
Irvine, CA 92697-3435, USA
umukherj@ics.uci.edu

M. Gopi
Department of Computer Science
University of California, Irvine
Irvine, CA 92697-3435, USA
gopi@ics.uci.edu

ABSTRACT

Tweening, also known as *shape morphing*, is an important concept in keyframe animation wherein an initial shape is transformed smoothly into a final shape. The huge body of existing literature in the areas of shape transformations and *tweening* in 2D is mostly restricted to transformations between simple non-self-intersecting polygonal shapes. In this paper we introduce a robust tweening algorithm which is capable of creating smooth transformations between non-simple polygonal shapes that are immersions of a disk. All the intermediate shapes generated by our algorithm are guaranteed to be disk immersions. We achieve this by calculating high quality triangulations, from given arbitrary triangulations, of the initial and final non-simple polygons and establishing a homeomorphism between the disks of the two immersions. If the triangulations of the source and target shapes are incompatible, we change the geometry as well as the triangulation smoothly during the morphing process.

1. INTRODUCTION

Tweening or *shape morphing* is the process of gradually transforming one geometric shape into another. The concept of morphing is widely used in the fields of computer animation and shape modelling wherein intermediate geometric shapes are generated between the *source* and *target* shapes, to give the appearance that the source shape evolves smoothly into the target shape.

Most of the work in the area of shape morphing in 2D are, however, restricted to transformations between simple polygons without self-intersections, and the tweening is expected to guarantee that all the intermediate shapes are also without self intersections. If the polygons are self-intersecting to start with, there exists no clear metric to evaluate the quality of an intermediate shape or property that it should satisfy. But there is an interesting class of self-intersecting closed curves that are boundaries of an immersed disk in 2D - a disk that is stretched and possibly self-overlapped, but not twisted. These boundary curves, also called the *self-overlapping curves* [25](Figure 1), can theoretically be morphed to each other with the constraint that all intermediate shapes are also boundaries of an immersed disk. In this paper, we present the first algorithm for such a tweening between boundary curves of immersions of the disk, even if the curves are self-intersecting.

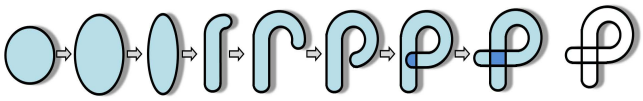


Figure 1: Obtaining a self-overlapping curve from a disk immersion: A disk painted blue on the front and red on the back side is stretched and overlapped (left to right) without twisting such that only the blue side is always visible. The boundary of the disk is called a self-overlapping curve (extreme right).

As Figure 1 suggests, a natural morphing sequence is implied by the process of forming a disk immersion. In order to compute this sequence, we require a mapping function between the interiors of the disk and the self-overlapping curve. A valid triangulation of the interior of the curve and a compatible triangulation of the disk serves as one such mapping function. Since there are many such triangulations possible, our goal is to re-mesh the interior of the curve from an arbitrary triangulation to one which is intuitive and suitable for morphing. The triangulation, thus obtained is used to transform the curve to a circle. This transformation algorithm can be extended to morph between two self-overlapping curves with correct vertex-to-vertex correspondence and same triangulation. In this context, we use the fact that both shapes are deformations of a disk to present a simple algorithm to find a good vertex-to-vertex correspondence. But if the two curves have different triangulations even after good vertex-to-vertex correspondence, then the transformation function becomes non-trivial as the triangulation also has to change during the morph from one shape to another. Naïve transformation without taking triangulation into account will introduce twists in the intermediate curves, and they will no longer be boundaries of an immersed disk.

Main Contributions: We present the first algorithm to transform between two self-overlapping curves. In order to achieve this, we design a novel re-triangulation algorithm specifically for self-overlapping curves that closely follows

*Corresponding author

ICVGIP '12, December 16-19, 2012, Mumbai, India

the disk deformation pattern and hence is suitable for morphing. Based on interesting observations on the dual graph of our triangulation, we present a simple algorithm to morph two compatible triangulations. Using similar observations, we demonstrate a method to find features in self-overlapping curves and to align two curves in order to reduce the difference in their triangulations. Finally, if the triangulations are incompatible, our transformation algorithm changes both shape and triangulation topology during the morph from one self-overlapping curve to another.

2. RELATED WORK

An algorithm to compute an immersion from a self-overlapping curve was first introduced in [25], wherein an arbitrary triangulation of the curve interior was obtained by means of a dynamic programming based approach. A completely different approach was taken in [19] to produce an abstract representation of a self-overlapping curve. This approach, however does not explicitly produce a triangulation of the curve. Since our morphing techniques rely on finding a correspondence between the interiors of the disk and its deformation, we start with a triangulation produced by [25]. However, any such arbitrary triangulation may not necessarily follow the deformation pattern of the disk, and hence may not yield high quality morphs. Given such a triangulation, we perform a series of edge swaps maintaining the *Delaunay* criterion, to obtain a triangulation which closely follows the deformation pattern of the disk. We have observed (Figure 6) and also statistically verified (Table 1) that a triangulation obtained in such a manner performs much better than an arbitrary triangulation in producing aesthetically pleasing morphs.

The most popular way of morphing geometric shapes is object space morphing which involves two important steps. The first step is to establish a good correspondence between the primitive objects, e.g. points and triangles, in the source and target shapes. The second step is to obtain a good locus for each such object in the morphing process, such that local deformations and undesired self-intersections in the intermediate shapes are reduced as much as possible [22, 21, 18, 26].

Morphing between boundary curves can also be performed by re-parametrization of one of the curves [5, 20], or by using implicit functions to represent the curves [27, 28].

While most of the above approaches interpolate between source and target curves explicitly, they have to take care of avoiding self-intersections in the intermediate shapes also explicitly (except when using implicit functions). One way to ensure suitable properties for intermediate shapes during morphing is to map the interiors of the source and target closed curves, and morph them. Triangulating both the source and the target shapes with a compatible triangulation is an example of such a mapping [8, 9, 7, 23]. A compatible triangulation between the source and target shapes can be obtained by introducing *Steiner vertices* [2] in the interiors of the shapes. [3] shows that two simple polygons can be triangulated in a compatible way using at most $O(n^2)$ Steiner vertices, where n is the number of points in each polygon. The number of Steiner vertices required to obtain a compatible triangulation depends largely on the geometric correlation of the source and target shapes making this method quite complex.

Another approach followed in morphing geometric shapes and genus 0 polyhedral surfaces is to use frequency domain analysis [13] or wavelet domain analysis [12].

When the source and target shapes are simple polygons, both of them can be mapped to a common convex shape, and the union set of the points in the source and target can be used for interpolation in morphing. Each point in this union set will have a representative in both the shapes and hence vertex correspondence is established [4, 17, 1, 14, 15, 24] Many morphing algorithms also employ user interaction for specifying certain feature points in the source and target shapes [10, 16].

The input to our problem is a triangulation of the interior of a self-overlapping curve obtained from [25]. This triangulation explicitly provides a mapping between the disk and the curve interior. We reformulate this mapping function by re-meshing the curve interior, such that the obtained mapping closely follows the deformation pattern of the disk. Since the curve is a boundary of an immersion of a disk, the triangulation of its interior is topologically equivalent to triangulation of a disk. This forces the dual graph of our triangulation to be a tree, and specifically with no vertex in the interior of the shape/disk. Hence in contrast to all the previous methods, our morphing algorithm does not use internal Steiner vertices, but re-models the mapping function itself. Once a good triangulation of the interior of the curve is obtained, each individual triangle can be morphed from source to target, using only affine transformations which are easy to compute and symmetric by construction. We also present an algorithm that makes use of the tree structure of the dual graph of our triangulation in order to compute the prominent features of the curve. These features in two curves can be aligned to provide us with the vertex correspondence that is required for morphing one self-overlapping curve into another.

3. ALGORITHM OVERVIEW

In this paper, we present the first algorithm to morph between two self overlapping curves which are boundaries of immersions of a disk, such that all the intermediate curves are also boundaries of immersions of a disk.

Triangulating self overlapping curves: Our first goal is to morph a circle, which is the boundary of a disk to a self-overlapping curve. Given a self-overlapping curve, there are many mappings or immersion functions from a disk to its deformation. The quality of the morph depends on the uniformity of this mapping in terms of various geometric parameters like area and distance between points. A direct approach of obtaining this mapping is to triangulate the disk and its immersion with a compatible triangulation[1]. Such a triangulation of the self-overlapping curve is computed by [25] in $O(n^3)$ time, where n is the number of vertices of the curve. However, there are many such triangulations possible, and computing all possible triangulations and choose the best triangulation for our application is of exponential complexity and hence prohibitively expensive. So, instead of computing the *best* triangulation directly, we start with an arbitrary triangulation and perform a series of edge swap

[1]Two triangulations are compatible if there exists a bijective function π between the vertices of the triangulations such that the vertices ijk form a triangle in the first shape if and only if the vertices $\pi(i)\pi(j)\pi(k)$ form a triangle in the second shape (Figure 2).

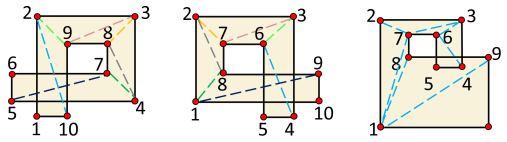


Figure 2: Compatible and incompatible triangulations. The left and center triangulations are compatible. Given a vertex i in the left figure, its corresponding vertex in center figure is $\pi(i) = (i + 8) mod 10 + 1$. The right triangulation is incompatible with the other two. A simple observation is that the vertex 1 in the right triangulation has degree 5 while no vertex in the other triangulations have degree 5, implying that they are incompatible.

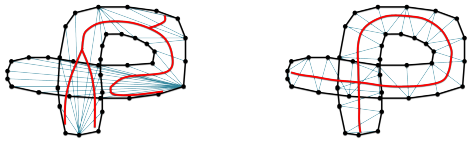


Figure 3: The dual graphs and their simplifications of two different triangulations of the interior of a self-overlapping curve. Left: arbitrary triangulation, right: triangulation obtained by re-meshing, which gives an indication of the deformation pattern of the disk.

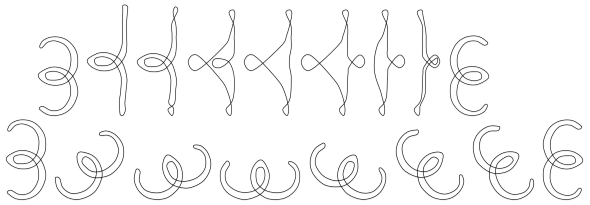


Figure 4: Top: Morphing a self-overlapping curve into another without taking into account their interiors introduces twists. Bottom: Morphing of the same curves taking into account their interiors produces a simple rotation.

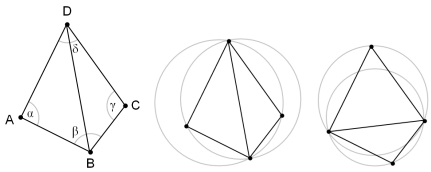


Figure 5: Edge flipping to attain Delaunay condition. Left: The sum of the angles α and δ is less than $180°$. Center: This triangulation does not meet the Delaunay condition as the circumcircles contain more than three points. Right: Flipping the common edge produces a triangulation which meets the Delaunay condition.

operations to change the triangulation to the one that satisfies constrained Delaunay properties. We observe that both qualitatively (Figure 3) and quantitatively (Table 1), such a triangulation is more suitable for morphing applications than any arbitrary triangulation.

Morphing compatible triangulations: Once a triangulation is obtained for the deformed disk, the same triangulation is imposed on the (convex shape) disk, thus yielding a compatible triangulation. If the source and target shapes are two self-overlapping curves which are triangulated independently, a simple $O(n^2)$ search can be done to check if the two triangulations are compatible, and if they are, we find the vertex-vertex correspondences between two curves that results in compatible triangulation (Section 5).

As a side note, we would like to mention that without considering the mapping between the interiors of the curves, no curve-to-curve morphing algorithm between the self overlapping curves can ensure that all the intermediate shapes are also self overlapping (Figure 4).

Morphing incompatible triangulations: Since the dual graph of the re-triangulation closely follows the disk-deformation pattern (Figure 3), we use this observation to identify dominant features of the shape and present a simple method to align these features in two shapes. This significantly reduces the incompatibility in the triangulation (Section 6.1). The remaining incompatibility is gradually removed during the morphing process by a series of *edge flip* operations(Section 6.2).

4. TRIANGULATING SELF-OVERLAPPING CURVES

Given a self-overlapping curve, an arbitrary triangulation of its interior can be obtained from [25] in $O(n^3)$ time, where n is the number of points on the curve. Since our morphing algorithm is based on transforming each individual triangle, the quality of the morph depends largely on the triangulation. For example, the presence of many sliver triangles degrades the quality of morph considerably. In order to attain a uniform triangulation by eliminating sliver triangles as much as possible, we perform a series of edge swaps on the given triangulation based on the Delaunay criterion (Figure 5). Note that, any number of valid edge swaps will still produce a valid triangulation of the interior of the curve. Effectively, at the end of this step we obtain a Constrained Delaunay Triangulation (CDT) of the interior of the curve. Further, note that traditional CDT cannot be performed on a self-overlapping curve because it is by definition a self-intersecting polygon. As Figure 3 shows, the triangulation obtained using the edge-swapping process clearly brings out the deformation pattern of the disk.

The edge flipping algorithm [6] is summarized in Algorithm 1. Next we verify that the algorithm terminates. Let A denote a vector of interior angles of all the triangles in an arbitrary triangulation T, sorted in ascending order, and A' be the same for a Delaunay triangulation. Since a Delaunay Triangulation maximizes the minimal angle, A' is lexicographically larger than A. As each edge flip increases the minimal angle of the quadrilateral concerned, the new angle vector is always lexicographically larger, upper bounded by A'. Hence the algorithm terminates.

4.1 Compatibility of Triangulations

A valid triangulation of the interior of a self-overlapping curve can be imposed on the disk, which is a convex shape. While morphing between two self-overlapping curves, we

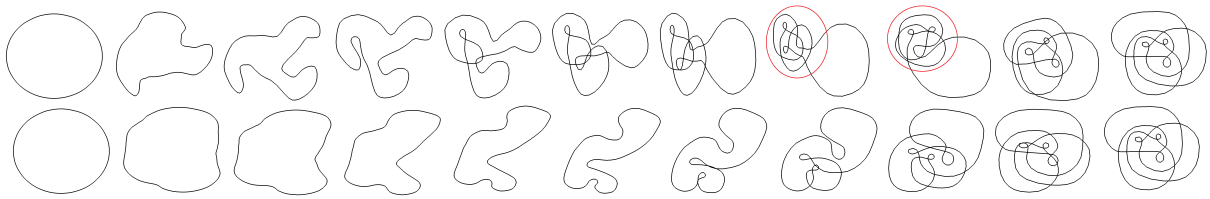


Figure 6: Morphing using a random triangulation (top) shows all the changes happen in a small region, and using our re-triangulation method (bottom) shows that the morphing among the regions of the curve is more uniformly distributed.

Algorithm 1 Edge Flipping algorithm

```
Input: An arbitrary triangulation, T, of the curve interior
Output: CDT, T′ of the curve interior
S: A stack containing all the edges of T
while S is non-empty do
POP an edge ab from S and mark it
if ab is not locally Delaunay then
FLIP ab to cd
for xy ∈ ac, cb, bd, da do
if xy is marked then
unmark xy and PUSH into S
end if
end for
end if
end while
```

first introduce vertices on the curves such that both are represented using the same number of vertices, say n. After triangulation, we need to check if their triangulations are compatible. Since both shapes do not have any internal vertices, and have the same number of vertices on the boundary, there is an implicit ordering of vertices in either of the curves. Assuming a fixed orientation for both curves, let the ordering of vertices in sequence be p_i and q_i, $0 \leq i < n$, on the first and second curves respectively. Let $d(v)$ denote the degree of vertex v. We observe that, in our specific problem, two triangulations are compatible if and only if there exists a k, $0 \leq k < n$ such that $d(p_i) = d(q_{(i+k) \mod n}), \forall i, 0 \leq i < n$. This compatibility can be checked by a simple $O(n^2)$ method. Further the value of k that satisfies the above condition gives the vertex-vertex correspondence between the curves that is required for morphing between compatible triangulations.

5. MORPHING COMPATIBLE TRIANGULATIONS

Given compatible triangulations of the interiors of two self-overlapping curves, we transform each triangle from its source position in one triangulation to its corresponding target triangle in the other triangulation. Since the dual graph of a triangulation of a disk, and hence the interior of a self-overlapping curve, is a tree, we choose a triangle as a starting triangle (the root node) to propagate the transformations through the entire tree. Given such a rooted tree, every node has a unique parent and hence the transformation propagation direction is also unique. For every triangle, we call the edge that is shared by its parent as the *base* of that triangle, the vertex not on the base as its *third vertex*, and the edge(s) shared by its children as *propagation edge(s)* of the triangle. Let the shape interpolation factor t be 0 for the source shape, 1 for the target shape, and between 0 and 1 for all the intermediate shapes. Given the source triangle, its corresponding target triangle, and the shape transformation factor t, the goal is to find the new shape, T of the source triangle.

We transform the source triangle such that the vertices on the base of the source and target triangles align with each other. Let this transformation be D. Then we *move* the third vertex of the source toward the third vertex of the target as much as dictated by t, to get the intermediate triangle. Moving of the third vertex from source to target can follow any reasonable path and this path affects the quality of morphing. We discuss more about this path later in this section. This movement of third vertex is followed by the inverse transformation D^{-1} of the intermediate triangle to get the desired morphed triangle. This process is repeated for every triangle from the root triangle to all the leaf triangles in the tree in order to get the intermediate deformed shape of the source curve corresponding to t. The above morphing is repeated in sequence for discrete values of t from 0 to 1 to get the entire morphing sequence.

Note that during the transformation of one triangle, the two vertices of the base edge are not moved at all. With the transformation D^{-1}, the vertices of the base edge transforms back to their original positions. Only the third vertex is moved in the process. But when the third vertex is moved, the edges incident on it, which may be propagation edges, change, thus affecting the base edges of the children of this triangle. Thus the transformation implicitly propagates through the tree through the movement of just one vertex (namely, the third vertex) for every triangle. This simple procedure also converges to the target shape when $t = 1$, despite the intermediate transformation D. In other words, the transformation D for all triangles (except for the root node) also converge to an identity matrix when $t = 1$. At the end of the morphing process, the entire source and target self-overlapping curves differ only by transformation D of the root node, which is also smoothly interpolated through the morphing sequence in order to become the target root triangle.

As is clear from the above procedure, interpolation of the third vertex for each triangle is critical in arriving at a good morphing. A simple procedure is to linearly interpolate the position of the third vertex from its source and target positions. However, the position of the third vertex dictates the length of the propagation edge, if any, of the triangle, and hence the base edge of its children, say T_c. Ideally, for

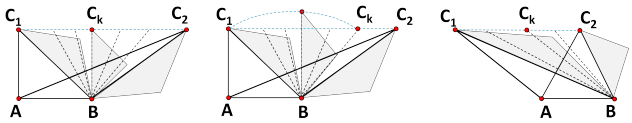


Figure 7: Transforming a source triangle ABC_1 to the target triangle ABC_2. BC_1 and BC_2 are the propagation edges in the source and target respectively. Left: a simple sliding of the propagation edge will first reduce its length, and hence the area of the child triangle (shown in grey) and then increase it. So a rotate and slide mechanism is used (center) to calculate the length of the propagation edge at an intermediate stage. The propagation edge retains its length from the source triangle upto the point C_k and then starts increasing monotonically. Right: when the propagation edge is monotonically decreasing from the source to target a simple sliding of the third vertex from C_1 to C_2 is sufficient.

a smooth morph, the base length of T_c should increase or decrease monotonically during the morph, which cannot always be guaranteed by a linear interpolation (Figure 7). So, instead of simply sliding the third vertex from its source to target positions, we use the following approach that ensures that the length of the (propagation) edge that is the base edge of a child is always between the lengths of the corresponding source and target edges.

Let l_1 and l_2 be the lengths of the corresponding propagation edges of the source and target triangles, and l_t be the same in any intermediate shape. We use linear interpolation between the third vertices of source and target triangles, if during this process l_t increases or decreases monotonically. Otherwise, a part of this path is replaced by a circular curve segment with radius $\min(l_1, l_2)$, while the rest remains a line segment (Figure 7). The comparison of the results of the morphing process, one using linear transformation and the other using our new slide-and-rotate method is shown in Figure 8.

The above method works even if both edges incident on the third vertex are propagation edges. In this case, at most one of these edges will have non-monotonic changes in its length during linear morph, and the above algorithm has to be applied to that edge. Our method not only makes the length variation monotonic for the propagation edge that it is applied to, but also preserves monotonicity of the length of the other propagation edge.

While a linear interpolation of the third vertex is better at preserving or smoothly interpolating the areas of the triangles, it can adversely affect the transformations of its children. Our above modified path is better at achieving predictable variations of the edge lengths, and thus isolates the side-effects of transformation of the parent from propagating to the children (Figure 8).

6. MORPHING INCOMPATIBLE TRIANGULATIONS

Let us now consider incompatible triangulations of two self overlapping curves. We first insert vertices on the curve to ensure both curves have the same number of vertices. Then we identify a single vertex per shape that in some sense represents an important feature of that shape. These vertices, called the pivot vertices, are aligned between shapes and the vertices are re-indexed. This alignment of features improve the quality of the morph. Since the number of vertices are the same in both self-overlapping curves, one triangulation can be transformed to another through a sequence of edge flips, and be made into compatible triangulations. The only problem is that not all edge flips are valid in the presence of concavities in the curve. We perform edge flips gradually when the shape morphs and becomes more and more convex. Once the triangulation of the other shape is reached, they become compatible triangulations, morphing between which can be done as described in Section 5.

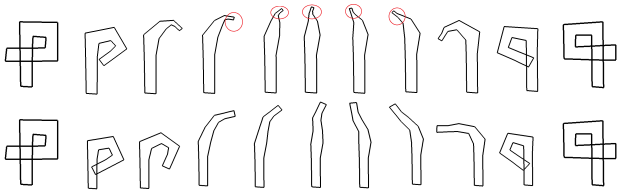


Figure 8: Top: Morphing with linear interpolation of the third vertex. Bottom: Interpolation with rotation and sliding. The shrinkage in shape in the top row is highlighted with red circles, showing that rotation and sliding is better than simple interpolation.

6.1 Vertex correspondence by Pivot Triangle Matching

Given a self-overlapping curve, we present a simple algorithm to identify its visually prominent vertex. Since the triangulation of the interior of self-overlapping curve is just a triangulation of a disk, its dual graph is a tree. The triangles corresponding to the leaf nodes of the dual graph represents visually important features of the shape, and the path between the two farthest leaf nodes of the graph would represent the 'spine' of the deformation. The algorithm to find the 'spine' in the tree proceeds as follows. Let the nodes in the dual graph with degree 1 be called *leaf nodes*, with degree 2 be called *path nodes* and others, *internal nodes*. First a leaf node that is closest (in terms of number of hops) to an internal node is identified. This leaf node and all the edges in its path to the internal node are removed. This reduces the degree of that internal node, and in the dual graph of a triangulation, this internal node will become a path node. The above process is repeated until there are only two leaf nodes left, and we define the path between them as the spine of the shape. We choose one of the two end triangles of the spine, and label it a *pivot triangle*, and the vertex of the pivot triangle that is not shared by its neighboring triangle as the *pivot vertex*. We align the pivot vertices of the two shapes and reindex their vertices in order to achieve the required alignment of features.

The alignment of pivot vertices helps significantly during the morphing process. Figure 9 shows the morphing between two shapes with and without this alignment. Pivot vertices are marked in the last two rows in Figure 11. Pivot triangles capture in some sense, the semantics of the shape and hence has proved critical for morphing.

6.2 Edge Flips between two Triangulations

Figure 9: Morphing with (bottom) and without(top) matching stable triangles.

Section 5 describes an algorithm to morph between shapes with compatible triangulations. Two triangulations of the same point set can be transformed from one to another, and made compatible by applying a finite number of edge-flip operations [11].

Let us now consider two triangulations of a disk with the same point set. In general, computing the minimum number of edge flips to convert one triangulation to another is not trivial. As [11] has shown, the upper bound of the number of such edge flips required is the number of intersections between the edges of the source and target triangulations. However, the complexity of the algorithm to compute such a minimum sequence of edge flips is unknown, even for a convex polygonal shape. We follow a simple algorithm to determine a sequence of edge flips, which may not be minimum, but guarantees the conversion of one triangulation to another. The algorithm is based on reducing the number of intersections between the edges of the two triangulations. First, we sort the edges in the target triangulation of the disk in an increasing order of the number of its intersections with the edges in the source triangulation of the same disk. Then, for each edge in this sorted sequence, we flip all the edges in the source triangulation that intersect it.

Consider two self overlapping curves S_1 and S_2 with same number of points, and their triangulations T_1 and T_2. Using the algorithm described in Section 5, we can impose the triangulation T_1 on a disk with same number of points and morph S_1 to disk. At the end of the morph sequence, we can change the triangulation of the disk to T_2 with the index-aligned set of vertices between S_1 and S_2. With this new triangulation T_2 of the disk being compatible with the triangulation of S_2, we can again morph between this disk to S_2, thus achieving a sequence of morph between S_1 and S_2.

Let the intermediate shapes of morphing S_1 to a disk be $S_1(u)$, and that of morphing from S_2 to disk be $S_2(v)$. Instead of performing all the edge flips only after S_1 is fully morphed to a disk, we propose to do the edge flips during the morphing process as and when it is allowed by the convexity of $S_1(u)$ (Figure 10). When the edge flips are done to $S_1(u)$, the corresponding edge flips are also performed on the disk to which it is morphing, in order to maintain the compatibility of the triangulations. Once the triangulation of T_2 is reached on $S_1(u)$, for some $u \leq 1$, since it now has a compatible triangulation with S_2, instead of morphing to a disk, $S_1(u)$ can be directly morphed to S_2. A similar value for v can be found at which $S_2(v)$ allows T_1, the triangulation of S_1.

Ideally, we would like the morphing function to be symmetric. In other words, if $M(S_1, S_2)$ gives all the intermediate shapes when morphing from S_1 to S_2, then we would like $M(S_1, S_2) = M(S_2, S_1)$. So we compute both u and v and choose the minimum of u and v. Without loss of generality, let v be the minimum. We use all the sequence of shapes between S_1 to $S_2(v)$ and from $S_2(v)$ to S_2 to be the total sequence of shapes for morphing between S_1 and S_2.

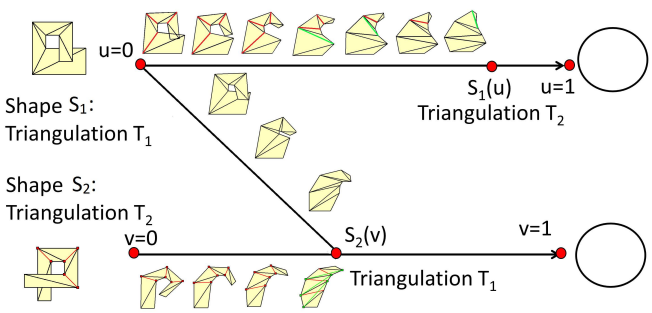


Figure 10: Morphing two shapes through a sequence of edge-flips. Shape 1 has a triangulation T_1 and shape 2 has a triangulation T_2 to start with. Each shape is morphed to a disk with gradual edge flips, such that the triangulation of the shape and the corresponding disk changes at every step. At position u, the morphed shape 1, $S_1(u)$ has attained the triangulation T_2, and at v, the morphed shape 2, $S2(v)$ has attained the triangulation T_1. Assuming $v < u$, $S_2(v)$ is morphed to shape 1, each intermediate morphing having the triangulation T_1.

7. EXTENSION, ANALYSIS AND DISCUSSION

Evaluation: The quality of the morphing is dependent on the triangulation of the deformed disk. We would like to quantitatively evaluate the quality of the triangulation for its suitability to morphing application. Second, we have observed that if in different geometric parameters (like area), the different triangles vary uniformly throughout the morph sequence, then we get a smooth morph. Based on this observation we construct a matrix of triangle areas where each row of the matrix denotes the area of each triangle in the triangulation, and each column denotes a morph sequence or frame. The rank of this matrix gives us a measure of the uniformity of areas of different triangles over different morph frames. If the areas of the triangles vary perfectly uniformly over the morph sequence, which is characteristic of a good morph, the columns of this matrix will be linearly dependent upon one another, and we will get a very low rank of the matrix. We also compute the absolute change in area of each triangle between every two adjacent frames i and $i+1$. This gives us an n vector of differences for every frame, where n is the total number of triangles. The length of this n-vector gives the magnitude of the difference between frames i and $i+1$. For a uniform morph, we can expect this magnitude to be the same between any two adjacent frames. Variance in this magnitude should be minimum for a smooth morph. Results show that our triangulation quantitatively produces a much better morph when compared to the those produced by an arbitrary triangulation (Table 1).

Discussion: As discussed earlier, the dual graph of a triangulation of a self-overlapping curve is a tree, and the transformation propagation of the triangles for morphing begins by selecting a node of this tree as a the root. We have observed that different choices of the starting triangle

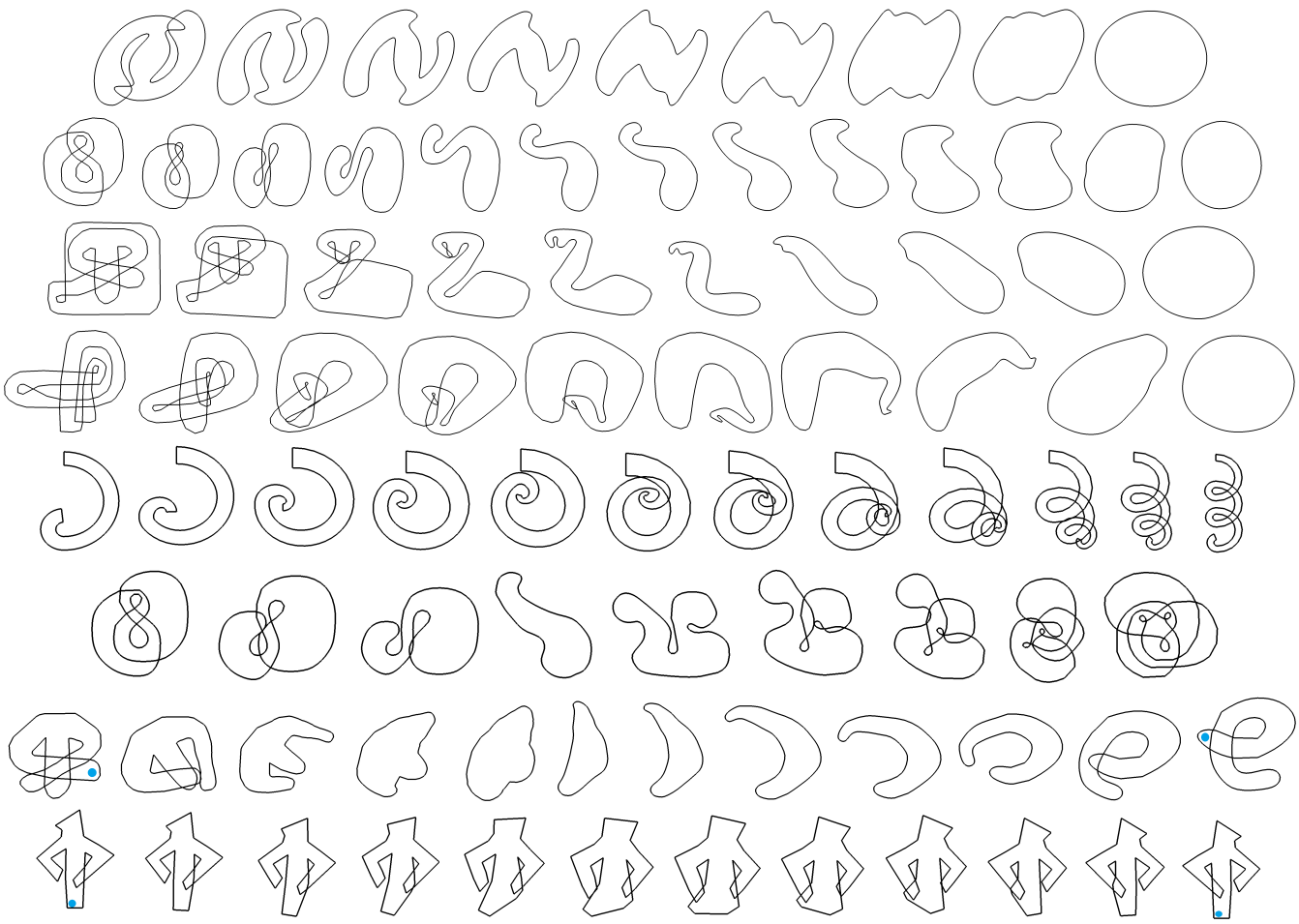
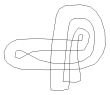


Figure 11: Morphing two self-overlapping curves using our re-triangulation method. The last four rows show the results of morphing between incompatible triangulations. The blue dots in the source and target shapes denote the matched pivot vertices.

Table 1: Comparison of the quality of morph, based on statistical parameters of triangulations. The first row shows the curves for which these parameters are evaluated. The third row gives the rank of the matrix of areas (AR), and the fourth row gives the variance of difference in areas of the triangles for a given shape between successive frames. The Y column under each shape gives the results from an arbitrary triangulation and the X column gives the same for the morph obtained by re-meshing.

Curve								
Re-meshed/arbitrary	X	Y	X	Y	X	Y	X	Y
AR	7	31	17	42	10	13	23	36
Var	0.01	0.11	0.01	0.02	.007	0.008	0.007	.03

have an effect on the quality of morphing. Experimentally, we have chosen the starting triangle to be the one which lies closest to the center of the disk. This has the effect of uniformly propagating the transformation in all directions. Deeper investigation is required to find the best possible starting triangle for the best morphing results.

When the triangulations of the two shapes are not compatible, we perform a sequence of edge flips to convert the triangulation of one to another. Since the edge flips are in sequence, certain edge flips cannot be performed until an edge flip appearing earlier in the sequence has been performed. Since an edge flip is possible only if the corresponding vertex is convex, it may so happen that most of the edge flips are possible when a shape is very near to a circle in the morphing sequence. Ideally, we would want all the edge flips to be complete as early as possible to produce a smooth morph. Although it may not be theoretically possible to compute the minimum number of edge flip sequence, more investigation is required for ordering of given set of edge flips (not necessarily minimum) such that all the edge flips can be completed as early as possible in the morphing process.

Feature alignment of the two shapes while morphing is also an important factor which affects the smoothness of

morph. The alignment produced by our pivot triangle matching discussed earlier assumes that with one feature alignment, all other features (leaf triangles) may also get aligned. We need a better method to align multiple features, and retriangulate the curve based on this alignment. We believe that this method can produce better morphing results. We are also working on alternative structures for representing self-overlapping curves that may be suitable for morphing.

Finally, our algorithm can be extended to morph between closed-loop curves which are obtained by twisting the disk at multiple points. If the points of such twists can be identified, the curve can be split into smaller parts, and each individual part can then be morphed. Although there exists algorithms to detect whether a curve has twists([25, 19]), such algorithms cannot explicitly identify the twist positions. We are currently working on methods to identify these twist points precisely.

8. SUMMARY

Our core contribution of this work is a new morphing algorithm for a class of self-intersecting curves that are immersions of a disk. Our morphing algorithm ensures that all the intermediate shapes during the morph are also self-overlapping. Our algorithm is also extended to handle self-overlapping curves that have incompatible triangulations. We have presented a progressive edge flip based algorithm, that converts both the geometry and the triangulation during morphing, in order to achieve compatible triangulation. This algorithm is guaranteed to terminate as any self-overlapping curve can be morphed to a circle and this circle, being convex, can allow the triangulations of both the source and the target self-overlapping curves.

9. REFERENCES

[1] M. Alexa. Merging Polyhedral Shapes with Scattered Features. *The Visual Computer*, 16(1):26–37, 2000.

[2] M. Alexa, D. Cohen-Or, and D. Levin. As-rigid-as-possible shape interpolation. *SIGGRAPH*, pages 157–164, 2000.

[3] B. Aronov, R. Seidel, and D. Souvaine. On compatible triangulations of simple polygons. *Comput. Geom. Theory Appl.*, 3:27–35, June 1993.

[4] E. Carmel and D. Cohen-Or. Warp-guided object-space morphing. *The Visual Computer*, 13:465–478, 1998.

[5] S. Cohen, G. Elber, and R. Bar-Yehuda. Matching of freeform curves. *Computer-Aided Design*, 29(5):369 – 378, 1997.

[6] H. Edelsbrunner. Triangulations and meshes in comp. geometry. *Acta Numerica*, 9:133–213, 2000.

[7] M. Etzion and A. Rappoport. On compatible star decompositions of simple polygons. *IEEE TVCG*, 3:87–95, January 1997.

[8] M. S. Floater and C. Gotsman. How to morph tilings injectively. *Journal of Computational and Applied Mathematics*, 101:117–129, 1999.

[9] C. Gotsman and V. Surazhsky. Guaranteed intersection-free polygon morphing. *Computers and Graphics*, 25(1):67 – 75, 2001.

[10] A. Gregory, A. State, M. Lin, D. Manocha, and M. Livingston. Feature-based surface decomposition for correspondence and morphing between polyhedra. In *Proc. of Computer Animation*, pages 64 –71, 1998.

[11] S. Hanke, T. Ottmann, and S. Schuierer. The edge-flipping distance of triangulations. *Journal of Universal Computer Science*, 2(8):570–579, aug 1996.

[12] T. He, S. Wang, and A. Kaufman. Wavelet-based volume morphing. In *Proc. of IEEE Visualization 94*, pages 85–92, Washington, D.C., Oct. 1994.

[13] J. F. Hughes. Scheduled fourier volume morphing. SIGGRAPH, 1992, pages 43–46, 1992.

[14] T. Kanai, H. Suzuki, and F. Kimura. 3d geometric metamorphosis based on harmonic maps. In *Proc. of Pacific Graphics '97*, pages 97–104, 1997.

[15] J. R. Kent, W. E. Carlson, and R. E. Parent. Shape transformation for polyhedral objects. SIGGRAPH, 1992, pages 47–54, 1992.

[16] A. W. F. Lee, D. Dobkin, W. Sweldens, and P. Schröder. Multiresolution mesh morphing. SIGGRAPH, 1999, pages 343–350, 1999.

[17] A. Lerios, C. D. Garfinkle, and M. Levoy. Feature-based volume metamorphosis. SIGGRAPH, 1995, pages 449–456, 1995.

[18] L. Liu, G. Wang, B. Zhang, B. Guo, and H.-Y. Shum. Perceptually based approach for planar shape morphing. In *Pacific Conf. on Computer Graphics and Applications*, pages 111 – 120, oct. 2004.

[19] U. Mukherjee, M. Gopi, and J. Rossignac. Immersion and embedding of self-crossing loops. In *Proc. ACM/ Eurographics Symp. on Sketch-Based Interfaces and Modeling*, pages 31–38, 2011.

[20] T. Samoilov and G. Elber. Self-intersection elimination in metamorphosis of two-dimensional curves. *The Visual Computer*, 14:415–428, 1998.

[21] T. W. Sederberg, P. Gao, G. Wang, and H. Mu. 2-d shape blending: an intrinsic solution to the vertex path problem. SIGGRAPH, 1993, pages 15–18, 1993.

[22] T. W. Sederberg and E. Greenwood. A physically based approach to 2-d shape blending. SIGGRAPH, 1992, pages 25–34, 1992.

[23] M. Shapira and A. Rappoport. Shape blending using the star-skeleton representation. *IEEE Comput. Graph. Appl.*, 15:44–50, March 1995.

[24] A. Shapiro and A. Tal. Polyhedron realization for shape transformation. *The Visual Computer*, 14:429–444, 1998.

[25] P. W. Shor and C. J. Van Wyk. Detecting and decomposing self-overlapping curves. *Comput. Geom. Theory Appl.*, 2(1):31–50, 1992.

[26] H. Tian, Y. He, H. Cai, and L. Feng. Efficient metamorphosis of point-sampled geometry. In *Proc. 16th Intl. Conf. on Artificial Reality and Telexistence–Workshops*, pages 260 –263, 2006.

[27] G. Turk and J. F. O'Brien. Shape transformation using variational implicit functions. SIGGRAPH, 1999, pages 449–456.

[28] B. Whited and J. Rossignac. B-morphs between b-compatible curves in the plane. In *SIAM/ACM Joint Conference on Geometric and Physical Modeling*, SPM '09, pages 187–198, 2009.