Ball-morph: Definition, Implementation and Comparative Evaluation

Brian Whited and Jarek Rossignac

(Invited Paper)

Abstract—We define *b-compatibility* for planar curves and propose three ball morphing techniques between pairs of *b-compatible* curves. *Ball-morphs* use the automatic *ball-map* correspondence, proposed by Chazal et al. [1], from which we derive different vertex trajectories (*linear*, *circular*, *parabolic*). All three morphs are symmetric, meeting both curves with the same angle, which is a right angle for the *circular* and *parabolic*. We provide simple constructions for these *ball-morphs* and compare them to each other and to other simple morphs (*linear-interpolation*, *closest-projection*, *curvature-interpolation*, *laplace-blending*, *heat-propagation*) using six cost measures (*travel-distance*, *distortion*, *stretch*, *local acceleration*, *average squared mean curvature*, and *maximum squared mean curvature*). The results depend heavily on the input curves. Nevertheless, we found that the *linear ball-morph* has consistently the shortest *travel-distance* and that the *circular ball-morph* has the least amount of *distortion*.

Index Terms—Morphing, Curve Interpolation, Medial Axis, Curve Averaging, Surface Reconstruction from Slices, Ball-map

1 Introduction

The animation of a planar curve may be specified by drawing the shape of the curve at specific time values. These drawings are called key-frames or simply *keys*. Then, the problem is one of constructing an animation that continuously deforms the curve from one key to the next, while respecting the timing provided. Each segment of the animation between two consecutive keys is a morph and may be addressed independently, if one does not have to enforce derivative continuity across the keys. We explore here a new formulation of such morphs and their automatic construction and animation.

1.1 Problem statement

A variety of techniques have been proposed for computing automatically a morph between two curves P and Q in the plane (see [2] and [3] for examples). In this paper, we present a new family of three related morphs, which we call the *ball-morphs*, and discuss two related issues: (1) How to compare different morphing solutions and (2) How do the *ball-morphs* introduced here compare to each other and to other morphing approaches.

1.2 Motivation and applications

Morphing is a fundamental tool in animation design where *in-between* [4] frames are produced from a sparse set of key-frames that are often designed by lead artists [5]. Although several successful attempts at automating the construction of in-between frames have been proposed [6], the artist responsible for in-betweening like to have control over correspondence and over the trajectories for selected landmarks or stroke end-points. These specifications are difficult to automate because they involve aesthetic judgement, style guidelines, and context semantics about the relative 3D motions of the strokes and their mutual occlusions.

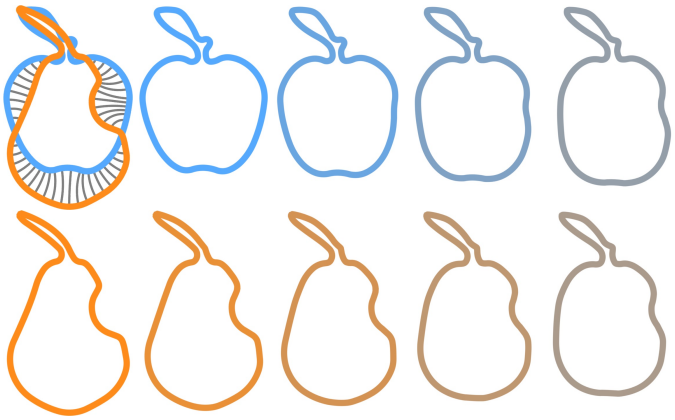


Fig. 1. A morph between an apple and pear along *circular ball-morph* trajectories (top left).

Once these matching and control trajectories are given, the overall problem is naturally broken into a series of tight in-betweening tasks [7] [8]. These are viewed as tedious and hence are a prime candidate for artist-supervised automation. In most of such tight in-between tasks, the goal is to generate intermediate frames between two reasonably simple and similar curve segments.

The help the artist select the inbetween technique best suited to a particular need, we show and compare several of these techniques to better assess the strength of each. This paper is a modest–although we

- *B. Whited is with Walt Disney Animation Studios, Burbank, CA, 91506. E-mail: brian.whited@disney.com*
- *J. Rossignac is with Georgia Institute of Technology, Atlanta, GA, 30332. E-mail: jarek@cc.gatech.edu*

hope useful–step in this direction. It may not be the final answer to tight in-betweening for several reasons: (1) The quantitative measures that we use may not reflect artistic concerns. (2) For practical reasons, we compare the proposed *ball-morphs* to our simple and unoptimized implementations of candidate techniques, and not to state-of-the-art solutions. More effective implementations of these competing approaches may exist. (3) We do not take into account the broader context of the whole animation, but instead focus on interpolating only the instances of the same stroke in two consecutive key-frames. Nevertheless, we hope that the experiments described here are useful and that the conclusions we draw from them about the specific benefits of the *ball-morphs* will help the reader appreciate their potential.

Furthermore, the problem (encountered in the segmentation of medical scans) of constructing a surface in 3D that interpolates between each pair of consecutive planar *cross-sections* may be solved [9] using the morphing between the projection, onto the same plane, of the two cross-section curves. This problem of surface reconstruction has been studied extensively [10][11][12][13][14]. Hence, we have included quality measures of the resulting surface in our set of metrics. As it was the case for tight in-betweening, our investigation of the benefit of *ball-morphs* to the problem of cross-section interpolation has limitations. For example, it only considers two consecutive slices, instead of building a smooth surface through the whole series, as proposed in [15]. However, because the *circular ball-morph* reaches the interpolated contours at right angles, the projection of these trajectories on the slice plane is C^1. We expect that this property may help researchers devise solutions that smoothly connect surface sections generated by *ball-morphs*. Furthermore, the approach is limited to *b-compatible* curves and hence is not suited for dealing with topological changes, as discussed for example in [16].

In these applications, the quality of the morph is important as one typically favors a solution where the animation or interpolating surface is smooth and free from self-intersections [17] and of unnecessary distortions. We show that when the curves are *b-compatible*, the *ball-morph* always satisfies these properties.

1.3 Contributions

We propose a family of three new morphing techniques (that we call *ball-morphs*) for which the correspondence and the vertex trajectories are both derived from the maximal disks and their tangential contact points with the curves.

We propose six cost measures for comparing morphs: *travel-distance*, *distortion*, *stretch*, *local acceleration*, *average squared mean curvature*, and *maximum squared mean curvature*.

We use these measures to compare the three *ball-morphs* to each other and also to a benchmark of five simple morphing techniques which we have implemented; *linear-interpolation*, *closest-projection*, *curvature-interpolation*, *laplace-blending*, *heat-propagation*.

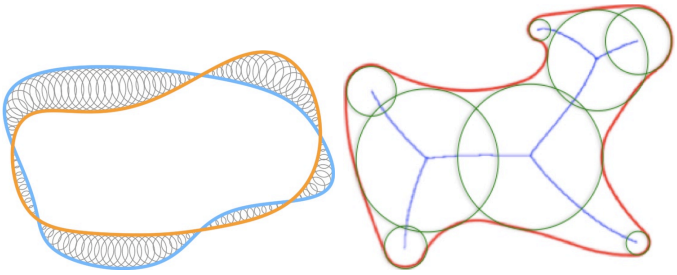


Fig. 2. Maximal disks (left) and medial axis (right) with bifurcation disks shown in green

1.4 Limitation

Our *ball-morph* constructions assume that the two curves have been registered and are sufficiently similar. We provide a formal definition of compatibility that captures these assumptions for the two situations considered here:

1) P and Q are each a simple closed loop.
2) P and Q are open curve segments and share the same two end-points.

Loosely speaking, our compatibility conditions require that each maximal disk [18] in the finite region bounded by the union of the two curves have exactly one contact point with each curve (see Fig. 2). Note that curves with concave sharp features relative to the symmetric difference of their interior are not *b-compatible* due to this condition.

Where P and Q are similar but not properly registered, one may consider combining a *ball-morph* with the animation of a rigid or non-rigid registration [19] or of a smooth space warp [20], as was done with other morphs for morphing images [21] and for tight-inbetweening [8]. Numerous solutions to the automatic registration problem have been proposed using ICP [22], automatically identified landmarks [23] [24] [25], or distortion minimizing parameterization [26] [27].

1.5 Structure of the paper

Section 2 briefly reviews prior art in curve morphing and slice interpolation. Section 3 provides a precise definition of *b-compatibility* and contrasts it with a previously proposed notion of normal compatibility. Section 4 presents our three *ball-morphs* and compares them to morphs obtained using closest projection and linear trajectories. Section 6 defines our six cost measures and explains our strategy for sampling and for a fair integration of these measures over the set of all trajectories. Section 8 discusses our results.

2 Prior Art

A large variety of techniques have been investigated for the automatic generation of in-betweening frames or animations that morph between two planar curves.

We only discuss techniques that are appropriate to the tight in-betweening problem addressed here. Hence, we do not discuss complementary techniques for registration or landmark (salient feature) identification.

First, we review techniques that assume that the correspondence between curve samples (or vertices of polygonal approximations of the curves) on both curves is either given by the artist or computed automatically during preprocessing using, for example, uniform geodesic sampling, minimization of area or travel [13], curvature-sensitive sampling [28], or optimization of matching to affine transformations [29].

If the correspondence is given, the simplest approach is to use a linear interpolation between corresponding pairs. Linear trajectories are computed between these pairs of points on P and Q in order to produce polygonal approximations of the evolving curve, the vertices of which move with time t as $\mathbf{v_i} = \mathbf{p_i} + t(\mathbf{q_i} - \mathbf{p_i})$. We refer to this solution as the *linear-interpolation*.

This naïve approach may lead to unpleasant artifacts, such as self-intersections in the intermediate frames (as, for example, pointed out by [30]). The *linear-interpolation* is oblivious to the relative orientation and curvature of the curves at the corresponding points.

To take the orientation and curvatures of both curves into account, a popular morphing technique proposed by [31] for polygonal curves interpolates the lengths of corresponding edges and the angles at corresponding vertices and uses optimization to ensure that the curve closes properly. We include a simple version of this approach, which we call *curvature-interpolation* in our benchmark set. When it is applied to open curve segments, we ensure that the interpolating frames meet, throughout the morph, at their two endpoints by retrofitting them through a trivial similarity transformation (rotation, scaling, and translation).

A different approach that takes into account the relative orientation and curvature of the two curves at the corresponding samples is to compute the local coordinates of each vertex in the coordinate system defined by its neighbors on each curve. Then, the corresponding local coordinates are averaged linearly to produce a *desired* set of local coordinates for a given frame. Iterative techniques may be used to construct a curve that satisfies the two endpoint constraints and minimizes the discrepancy between the actual and *desired* local coordinates. Variations of these techniques have been successfully used [32][33][34]. We include a simple version of this approach, which we call *laplace-blending*, in our benchmark set.

Vertex trajectories and correspondences may also be obtained by solving a PDE or by computing a gradient field that interpolates the two contours and then following the steepest gradient to obtain the trajectory of each point. Equivalently, the in-between frames may be obtained as iso-contours of that field. A heat propagation formulation may be used to characterize the desired field [35]. We include a simple version of this approach, which we call *heat-propagation*, in our benchmark set.

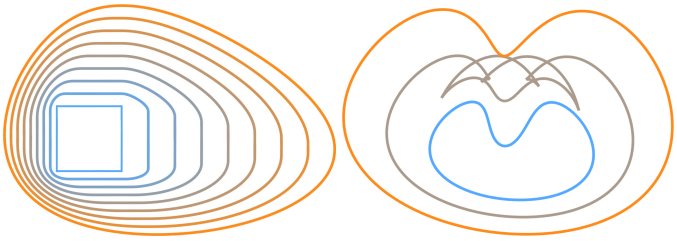


Fig. 3. Example Minkowski morphs between convex (left) and non-convex (right) shapes.

Several approaches for morphing closed curves use compatible triangulations [36] of their interior [37][38][3] or compatible skeletons to ensure rigidity [39][40]. Other approaches blend distance fields to both shapes [41][14].

Now, instead of relying on the global optimization or feature recognition techniques discussed above, let us focus on techniques that define an explicit geometric formulation of the correspondence. We separate them into three categories: (1) proximity-based, (2) orientation-based, and (3) both proximity-and-orientation-based.

The most popular distance-based approach is the closest point projection, which to each point $\mathbf{p}$ on P maps a point $\mathbf{q}$ on Q that minimizes the distance to $\mathbf{p}$. Variations of this approach are used for Iterative Closest Point (ICP) registration [42]. We include a simple version of this approach, which we call *closest-projection* in our benchmark set.

The simplest orientation-based approach is the Minkowski morph [43] and yields satisfying results for convex shapes (see Fig. 3 left), even when the shapes are not aligned. For smooth curves, the approach establishes a correspondence between points with the same normal. Unfortunately, as shown in Fig. 3 (right), the approach may yield surprising, sometimes self-intersecting frames when the two curves are not convex. Hence, we do not include it in our benchmark.

The *ball-map* [1], upon which the *ball-morphs* proposed here are based, takes into account both proximity and orientation.

3 Compatibility

3.1 Terminology and notation

Let us start by defining our terminology and notation.

Let P and Q be *manifold* curves in the plane. Recall that a manifold curve is free from self-intersections. When a curve is homeomorphic to a line segment, we call it a *stroke*. When it is homeomorphic to a circle, we call it a *loop*. We say that curves P and Q are *disjoint*, when their intersection is empty. We say that they *overlap* when their intersection contains at least one one-dimensional component. We say that they *cross* if they are not disjoint and do not overlap. We say that strokes P and Q are *quasi-disjoint* when they only intersect at their two endpoints.

Let S and T be arbitrary point sets in the plane. Let $!S$ denote the complement of S. We use the notation $S.i$, $S.b$,

and $S.c$ to refer to the topological interior, boundary and closure of S. The notation $S.e$ refers to the topological exterior, defined as $S.e = !S.c$. A set is *regularized* [44] when $(S.i).c = S$. A set S is *finite* when there exists a disk of finite radius containing it. Let $S \cup T$ denote their set-theoretic union, $S \cap T$ their intersection, and $S - T$ their difference. Let $S \oplus T$ denote their symmetric difference (XOR), defined as $(S \cup T) - (S \cap T)$.

Consider a one-dimensional subset B of the plane. We say that B is a *border* when there exists a finite regularized set S such that $B = S.b$. Note that, in that case, S is unique. When P and Q are loops, each one is a border, but the union of two loops needs not be a border (Fig. 4).

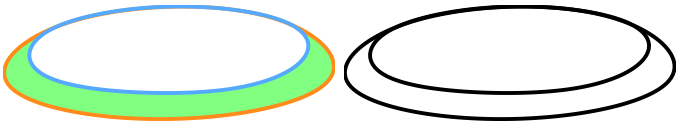


Fig. 4. Even when the union (right) of overlapping loops P (blue) and Q (blue drawn over orange) is not a border, we define their gap (green).

We define the *inside* $i(B)$ of a border B as the interior $S.i$ of the finite regularized set S such that $B = S.b$. Similarly, we define the *outside* $o(B)$ of a border B as the exterior $(i(B)).e$ of the inside of B. Note that $i(B)$ and $o(B)$ are topologically open sets. To test whether a point $\mathbf{p}$ that is not on B lies in $i(B)$, shoot a ray R (see black arrows in Fig. 9 from $\mathbf{p}$ that does not intersect any non-manifold point of B and is not tangent to B anywhere. If R crosses B an odd number of times, then $\mathbf{p}$ is in $i(B)$. Otherwise it is in $o(B)$.

Given a closed and regularized [44] set S, following [18], we say that a disk in S is *maximal* if it is not contained in any other disk in S and we define the *medial axis* as the closure of the union of the centers of maximal disks in S.

3.2 Topological validity

The set of valid configurations for which the *ball-morphs* can be computed has both topological and morphological limitations. In this subsection, we address the topological ones. First we present two general restrictions and one simplification.

We orient each curve and ensure that the orientations are compatible. For example, each loop is given a clockwise orientation and the strokes are oriented to have the same starting point. A configuration is invalid when P and Q overlap and have opposite orientations along any portion of the overlap.

When P and Q overlap with compatible orientations, we simplify the validity discussion by removing the overlap segments (except for their endpoints) and by identifying components of the remaining part as matching pairs of separate strokes. During the morph, the overlap segments remain static, hence we need not worry about them anymore. Each stroke of P shares its endpoints with a corresponding stroke of Q. We discuss below the morph of a pair of such strokes. From this point throughout the paper, we assume that we have identified and separated the overlap portions and hence that P and Q are not overlapping.

Finally, to avoid further complications, when P and Q are strokes with common endpoints, we require that the oriented loop obtained by combining P with reversed Q (Q for which we have reversed the orientation) has winding number (total number of turns made by the tangent vector as one traverses the loop) equal to one. Hence, configurations such as those in Fig. 5 are excluded.

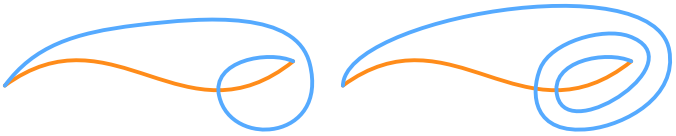


Fig. 5. Invalid configurations with winding number greater than one.

The two restrictions (on reverse orientation overlap and on winding number) and the simplification (remove overlaps) discussed here reduce the number of topological configurations to be discussed. Amongst the remaining ones, only four are valid. We define and illustrate them below.

Configuration 1: P and Q are disjoint loops and the intersection $i(P \cap Q)$ of their insides is not empty (Fig. 6).

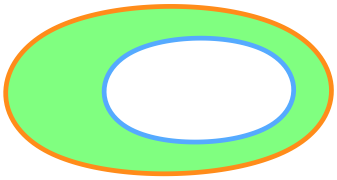


Fig. 6. Valid configuration 1: Disjoint loops with overlapping insides.

Configuration 2: P and Q are quasi-disjoint strokes (disjoint except for their shared endpoints) (Fig 7).

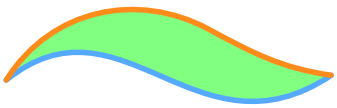


Fig. 7. Valid configuration 2: Quasi-disjoint strokes, with common endpoints

Configuration 3: P and Q are crossing loops and the intersection $i(P) \cap i(Q)$ of their insides has a single non-empty connected component (Fig. 8).

Configuration 4: P and Q are crossing strokes (Fig. 9) and the outside $o(P \cup Q)$ of their union is connected.

Note that we can decompose configurations 3 and 4 into one or more instances of configurations 2. In general, such a decomposition of P and Q into quasi-disjoint strokes may not be desired, since it imposes artificial constraints, forcing the evolving curves to interpolate

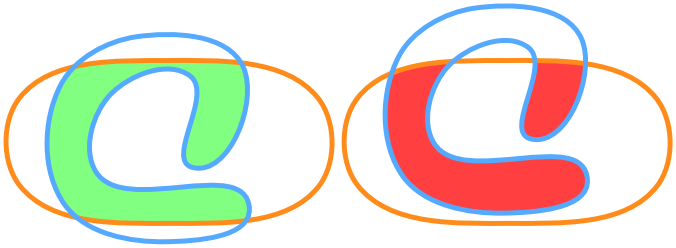


Fig. 8. Valid configuration of crossing loops (left) and invalid configuration (right) where $i(P) \cap i(Q)$ has two connected components (red).

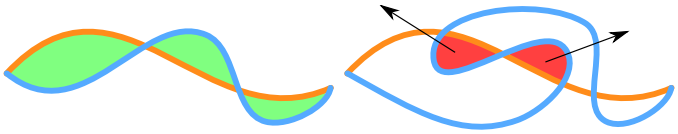


Fig. 9. Valid configuration of crossing strokes (left) and an invalid one (right) in which $o(P \cup Q)$, which is an open set, has three disjoint components, two of which are finite and are identified by outgoing black arrows which cross the union of the two curves an even number of times.

the points where P and Q intersect. Several of the morphing techniques discussed here (*ball-morphs*, *heat-propagation*, and *closest-projection*) naturally interpolate these intersection points anyway, hence the proposed decomposition does not alter their result. Other techniques, such as the *linear-interpolation* and *curvature-interpolation*, do not necessarily interpolate intersection points. These morphs are however highly dependent on the correspondence (parameterization map) between the two curves, which must be defined by the artist or computed independently. Furthermore, the techniques that interpolate intersection points may be combined with preprocessing steps that identify and match salient points, split each curve at these points, and execute a *composite* morphing of the of the corresponding pairs of strokes with a carrier (rigid, spiral or affine) motion that preserves coincidence of their endpoints [8]. Since we limit our attention to the comparison of morphing techniques, and not of techniques for computing correspondence, matching, and carrier motions, we force all morphs to interpolate the intersection points.

Hence, from now on, we only consider configurations 1 and 2.

3.3 Morphological validity

In this subsection, we define additional morphological validity constraints for the topologically valid configurations (1 and 2) listed above.

We use the notion of gap and its medial axis to define these constraints.

When P and Q are disjoint loops in configuration 1, their *gap* $G(P,Q)$ is defined as the symmetric difference $i(P) \oplus i(Q)$ of their insides. When P and Q are quasi-disjoint strokes in configuration 2, their *gap* $G(P,Q)$ is defined as the inside $i(P \cup Q)$ of their union. Note that the *gap* is a connected open set.

For topological configure 1, we require that the *medial axis* of the *gap* $G(P,Q)$ be a loop. Figure 10 shows a geometrically valid and a geometrically invalid configuration 1.

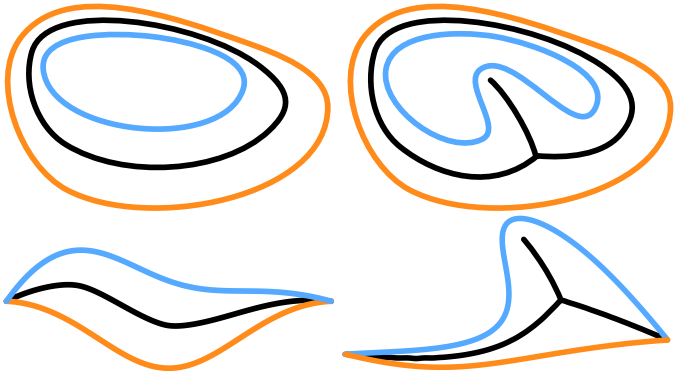


Fig. 10. Geometrically valid configurations (left) and invalid ones (right), with bifurcations of the medial axis, are shown for disjoint loops (top) and for quasi-disjoint strokes (bottom).

For topologically configuration 2, we require that the medial axis of the *gap* $G(P,Q)$ be a strokes with the same endpoints as P and Q. This additional precision is necessary to exclude situations such as the one depicted in Figure 11.

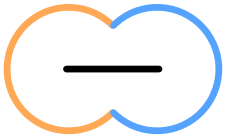


Fig. 11. The medial axis of a morphologically invalid configuration does not join the two endpoints of P and Q, which are concave vertices of the *gap*.

3.4 Definition and extension of *ball-compatibility*

The concept of *ball-compatibility* (abbreviated *b-compatibility*) has been introduced in [1] for two smooth loops. We say that loops P and Q are *b-compatible* when for every point $\mathbf{p}$ of P there exists a disk D in the *gap* $G(P,Q)$ that is tangent to P at $\mathbf{p}$, that does not intersect P anywhere else, and that is tangent to Q at a single point $\mathbf{q}$.

Note that the topological and morphological validity conditions presented above for loops imply *b-compatibility*.

When two curves are *b-compatible*, each can be expressed as the *ball-offset* of the other, i.e., as a portion of the envelope swept by a variable radius disk as it rolls on the other curve (Fig. 12 bottom).

Our topological and morphological constraints extend this notion of *b-compatibility* from loops to strokes.

3.5 Comparison with c-compatibility

The topological and morphological validity conditions discussed above may appear as a strong limitation of

the set of configurations to which the *ball-morph* may be applied. In this subsection, we dispel this perception by comparing them to conditions required by the popular *closest-projection-morph*.

The *closest-projection* of a point $\mathbf{m}$ onto a curve P is a set of points $\mathbf{p} \in P$ for which $distance(\mathbf{m}, \mathbf{p}) = distance(\mathbf{m}, P)$.

P and Q are *c-compatible* (i.e., closest-projection compatible) when every point of P or Q has a single closest projection on the other curve. Figure 13 shows a configuration where P and Q are *b-compatible* but not *c-compatible*.

Sufficient conditions for *b-compatibility* and *c-compatibility* have been discussed in [45], [46] and [1]: P and Q are *b-compatible* if $H(P,Q) < f$ and *c-compatible* when $H(P,Q) < cf$, where $H(P,Q)$ denotes the Hausdorff distance [47] between P and Q, f is the smallest of their minimum feature sizes [48], and $c = 2 - \sqrt{(2)} \approx 0.5858$. Note that a significant set of configurations satisfy the condition for *b-compatibility*, but not for *c-compatibility*. Hence, we argue that *b-compatibility* conditions are in fact less restrictive than the *c-compatibility* ones.

4 *Ball-morphs*

In this section, we describe the correspondence used for our *ball-morphs* and present the various options for *ball-morph* trajectories between corresponding pairs of points. The definitions are independent of the nature of the two curves and of their representation. We have implemented these techniques for two domains: (1) smooth (C^1) piecewise circular curves (*PCCs*) [49], and (2) relatively smooth polygons (such as those obtained through smoothing or subdivision). Our implementation on pairs of *b-compatible* piecewise-circular curves is numerically precise and yields the theoretically correct *ball-morph*. Clearly, the implementation for polygons is not theoretically correct. Indeed, the polygonized versions of two *b-compatible* curves are not *b-compatible*, because the region they bound must have convex vertices and hence bifurcations in its medial axis. Nevertheless, when the polygonal curves are reasonably smooth and densely sampled, our polygonal algorithm computes *ball-morphs* that closely approximate the *ball-morphs* of the original smooth curves and are acceptable for animation or surface reconstruction. Because most other morphing schemes to which we compare our *ball-morphs* work on polygonal curves, we use the polygonal *ball-morph* implementation to ensure consistency in our experiments.

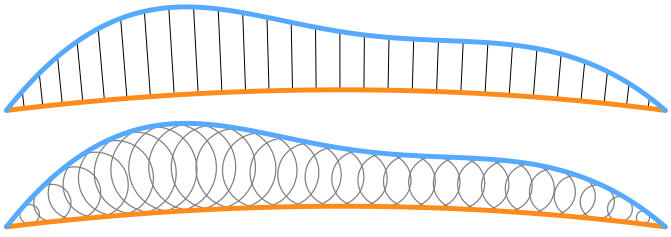


Fig. 12. The blue stroke can be expressed as the normal offset (top) or as the *ball-offset* (bottom) of the orange stroke.

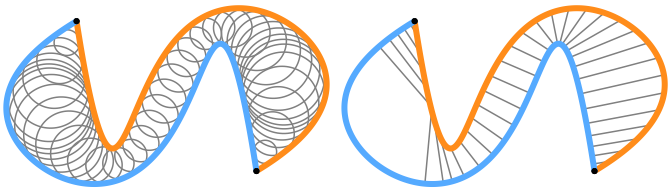


Fig. 13. Two curves that are *b-compatible* (left), but not *c-compatible* (right).

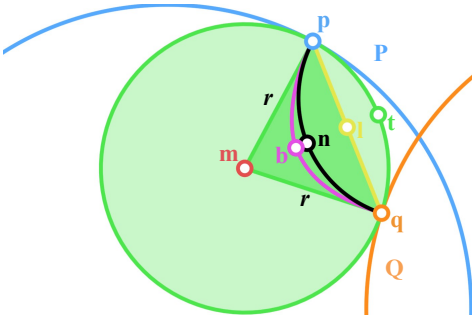


Fig. 14. To obtain the point $\mathbf{q}$ on Q that corresponds, through the *ball-map*, to point $\mathbf{p}$ on P, we compute the smallest positive r such that $\mathbf{m} = \mathbf{p} + r\widehat{\mathbf{N}}_P(\mathbf{p})$ is at distance r from Q and return its closest projection $\mathbf{q}$ on Q. Point $\mathbf{m}$ is on the median and defines the center of the circle tangent at both $\mathbf{p}$ and $\mathbf{q}$. The *circular* (black) and *parabolic* (purple) *ball-morph* trajectories are defined by the inscribing isosceles triangle $\Delta\mathbf{pmq}$. The *linear ball-morph* trajectory is the line segment $\mathbf{pq}$.

4.1 *Ball-map* correspondence

Consider the maximal disk centered at point $\mathbf{m} \in M$. The *ball-map* [1] establishes the correspondence between the closest projection $\mathbf{p}$ of $\mathbf{m}$ onto P and the closest projection $\mathbf{q}$ of $\mathbf{m}$ onto Q. The maximal disk D centered at $\mathbf{m}$ touches P at $\mathbf{p}$ and Q at $\mathbf{q}$, as shown in Fig. 14. The *ball-map* may be viewed as a continuous version of an approach proposed by [41] for establishing correspondences between surfaces by considering their distance fields. A uniform sampling of the *ball-map* correspondence may be computed in several ways: (1) By initially computing M as the medial axis of $G(P,Q)$ between two curves P and Q using efficient medial axis construction techniques [50][51] and then generating the closest projections $\mathbf{p}$ and $\mathbf{q}$ for a set of uniformly spaced sample points $\mathbf{m} \in M$; (2) By computing the radii of the maximal disks that touch P at a set of uniformly spaced samples $\mathbf{p}$; or (3) By simultaneously advancing the corresponding points, $\mathbf{p}$ and $\mathbf{q}$, until one of them has travelled from the previous sample on its curve by a prescribed geodesic distance. To ensure a fair comparison with techniques that lack the symmetry of the *ball-morphs*, we will use the second (asymmetric)

approach, although the first one yields the best results.

The details of the construction of this mapping for the case when P and Q are piecewise-circular and when they are polygonal approximations of smooth curves are provided below.

4.2 *Ball-morph* trajectories

For each maximal disk, we consider five *paths* (curve segments) from $\mathbf{p}$ to $\mathbf{q}$ (Fig 14):

- *Hat*: The broken line segment from $\mathbf{p}$ to $\mathbf{m}$ to $\mathbf{q}$ (Fig 14 green).
- *Linear*: The straight line segment from $\mathbf{p}$ to $\mathbf{q}$ (Fig 14 yellow).
- *Tangent*: The shorter of the two circular arc segments of the boundary of D that joins $\mathbf{p}$ and $\mathbf{q}$ (Fig 14 green).
- *Circular*: The circular arc segment that is orthogonal to P at $\mathbf{p}$ and to Q at $\mathbf{q}$ (Fig 14 black).
- *Parabolic*: The parabolic arc segment that is orthogonal to P at $\mathbf{p}$ and to Q at $\mathbf{q}$ (Fig 14 violet).

The *circular* and *parabolic* paths are trivially defined by their enclosing isosceles triangle $\Delta\mathbf{pmq}$. The *parabolic* path is the quadratic Bézier curve with control vertices $\mathbf{p}$, $\mathbf{m}$ and $\mathbf{q}$. The center of the circle supporting the *circular* path is the intersection of the tangent to P at $\mathbf{p}$ and the tangent to Q at $\mathbf{q}$.

All paths, including the *linear* path, are symmetric in that the angles where they meet P and Q are equal. Swapping the role of P and Q does not affect these segments. Hence, the *ball-morphs* derived here are *symmetric* and may be inverted easily by swapping the role of P and Q.

Let $\mathbf{l}$ be the midpoint of the *linear* path and let L be the set of all points $\mathbf{l}$. L is the midpoint locus proposed by Asada and Brady [52]. Let $\mathbf{t}$ be the midpoint of the *tangent* path and T be the set of all points t. T is the Process-Inferring Symmetry Axis (PISA) proposed by Layton [53] as a variation of the medial axis. Let $\mathbf{n}$ be the midpoint of the *circular* path and N the set of all points $\mathbf{n}$. Let $\mathbf{b}$ be the midpoint of the *parabolic* path (quadratic B-spline) and B be the set of all points $\mathbf{b}$. The construction of these 4 points, along with $\mathbf{m}$ is illustrated in Fig. 14. The curves M, L, T, N and B usually differ from one another, but may all be viewed as averages of P and Q.

A *ball-morph* advances, with time, each point $\mathbf{p}$ according to uniform arc-length parameterization along one of the five aforementioned paths. A result for the *circular ball-morph* is shown in Fig. 1 using seven inbetween frames.

5 Implementation Details

In this section, we provide implementation details for *PCCs* and then for polygonal curves.

5.1 Details of the *ball-map* construction for *PCCs*

We include here the details of an exact implementation (except for numerical round-off errors) for the case of piecewise-circular curves in 2D, where P and Q are each a series of smoothly connected circular-arc edges.

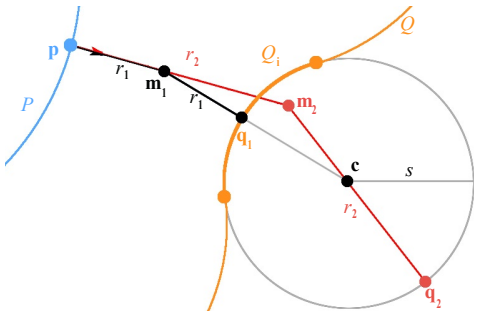


Fig. 15. Computing r, $\mathbf{m}$ and $\mathbf{p}$ from $\mathbf{q}$ for a circular arc Q_i. $\mathbf{q}_2$ is discarded since it does not lie on the arc Q_i.

We first explain how we compute the *ball-map* of a sample point. Then, we explain how to produce these samples and how to reduce the computational complexity of the whole process. We sample points on one curve and for each such sample, say $\mathbf{p}$ on P, we compute the corresponding point $\mathbf{q}$ on Q as explained below. Here we assume that $\mathbf{p}$ is not on Q, as discussed above. Consider the parameterized offset point $\mathbf{m} = r\widehat{\mathbf{N}}_P(\mathbf{p})$, whose distance from $\mathbf{p}$ is defined by the parameter r. $\widehat{\mathbf{N}}_P(\mathbf{p})$ is the normal of P at $\mathbf{p}$ and it is oriented so that it points towards the interior of the *gap* $G(P,Q)$. By construction, $\mathbf{m}$ is the center of a circle of radius r that is tangent to P at $\mathbf{p}$. We want to compute the smallest positive r for which $\mathbf{m}$ is at distance r from Q, and hence for which the circle is tangent to Q.

First consider a circular edge Q_i of Q with center $\mathbf{c}$ and radius s (Fig. 15). We compute r_1 and r_2 as the roots

$$\frac{s^2 - \overrightarrow{\mathbf{cp}}^2}{2\widehat{\mathbf{N}}_P(\mathbf{p}) \cdot \overrightarrow{\mathbf{cp}} \pm 2s} \tag{1}$$

of

$$\overrightarrow{\mathbf{cm}}^2 = (r \pm s)^2. \tag{2}$$

In order for $\mathbf{m}$ to define the center of a disk of radius r that is tangent to Q, $\mathbf{m}$ must have either a minimum or maximum distance of r from Q, or in other words, $\mathbf{m}$ must be at distance $r \pm s$ from $\mathbf{c}$, as defined by Equation 2. Substituting $\mathbf{m} = r\widehat{\mathbf{N}}_P(\mathbf{p})$ and expanding yields a second degree equation in r, the roots of which are given by Expression 1.

We apply the above approach to all edges Q_i of Q. We compute the r-value for a circle supporting each edge, compute the corresponding candidate point $\mathbf{q}$ on the circle, discard it if it is not contained within the arc (such as $\mathbf{q}_2$ in Figure 15), and select amongst the retained $(r, \mathbf{q})$ pairs with the one with the smallest r-value. Since we assume that P and Q are *b-compatible*, there is exactly one $(r, \mathbf{q})$ pair for each point $\mathbf{p} \in P$.

The above process computes the *ball-map* correspondence for any desired sampling of P or Q.

To accelerate the computation of the *ball-map* for *b-compatible PCCs* and produce a sampling-independent representation from which different sampling densities

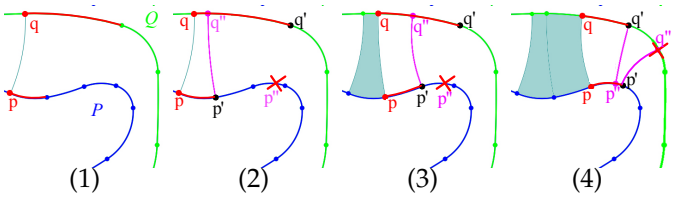


Fig. 16. The lacing algorithm for *b-compatible PCCs*. (1) An initial mapping is given between 2 points $\mathbf{p}, \mathbf{q}$. They each sit at the start of a circular arc edge segment (red). (2) A *ball-map* is computed from the endpoints of the current arcs $(\mathbf{p}', \mathbf{q}')$ to the current arc on the other curve (red). Only $\mathbf{p}'$ produces a valid *ball-map* to the edge segment $[\mathbf{q}, \mathbf{q}']$, so only it is kept. (3) $\mathbf{p}$ and $\mathbf{q}$ step forward to positions occupied by $\mathbf{p}'$ and $\mathbf{q}''$, and the process repeats. Again, only $\mathbf{p}'$ produces a valid *ball-map*, so only it is kept. (4) $\mathbf{p}$ and $\mathbf{q}$ again step forward. This time, $\mathbf{q}'$ produces the valid *ball-map*.

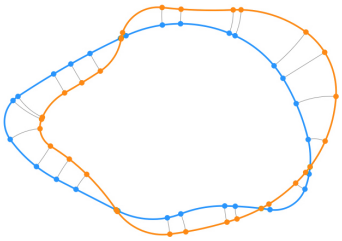


Fig. 17. Lacing splits the *gap* $G(P, Q)$ into *arc-quads*. Each one is bounded by 4 circular arcs (2 edge-segments and 2 *ball-map* arcs).

can be quickly derived, we perform a "lacing" process (Fig. 16), to split the *gap* into *arc-quads*, each bounded by 4 circular arcs: one being a circular-arc edge-segment of P, one being a circular-arc edge-segment of Q, and two being *ball-map* arcs from a vertex of P or Q to its image on the other curve.

To perform the lacing, we first pick a vertex $\mathbf{p} \in P$, where two edges of P meet and compute its image $\mathbf{q} \in Q$ as described above. Then, we perform a synchronized walk to "lace" the *gap*, one vertex of P or Q at a time. At each step, $\mathbf{p}$ is the start of an edge-segment P_i of P not yet laced and $\mathbf{q}$ is the start of an edge-segment Q_k of Q not yet laced. Let $\mathbf{p}'$ be the end of P_i and $\mathbf{q}'$ be the end of Q_k. Let $\mathbf{q}''$ be the corresponding point for $\mathbf{p}'$ and $\mathbf{p}''$ be the corresponding point for $\mathbf{q}'$. If $\mathbf{p}''$ falls on P_i, we record that the edge-segment $[\mathbf{q}, \mathbf{q}']$ of Q_k maps to the edge-segment $[\mathbf{p}, \mathbf{p}'']$ of P_i, close the current *arc-quad* with the arc from $\mathbf{q}'$ to $\mathbf{p}''$, and set $\mathbf{p}$ to $\mathbf{p}''$ and $\mathbf{q}$ to $\mathbf{q}'$ to continue the lacing process, as shown in Fig. 16.

This lacing process splits the edges of P and Q into edge-segments and establishes a bijective mapping between edge-segments of P and edge-segments of Q that bound the same *arc-quad*. The cost of this pre-computation is $O(n)$ in the number of edges in P and Q since at every step, only the current arc-edges P_i and Q_i are used to compute *ball-map* correspondences. It can be performed in real-time, as the curves are edited, which is convenient for the interactive design of *b-compatible* curves. For disjoint loops, the lacing starts at any vertex of P and terminates when it returns to the starting point. For quasi-disjoint strokes, the lacing starts at one common endpoint and finishes at the other endpoint. A small variation of this approach, described in [54], permits in linear time to either perform the lacing when the two curves are *b-compatible* or to detect that they are not.

5.2 Details of *ball-map* construction for *PLCs*

Because they are not smooth, piecewise-linear curves cannot be *b-compatible*. However, we propose here an approach for computing an approximate *ball-map*, treating piecewise-linear curves as approximations of smooth curves. We treat vertices as circular arcs with infinitely small radii and ignore incompatibilities as long as adjacent *ball-maps* do not intersect

First we show how to compute the *ball-map* from a point $\mathbf{p}$ on P to a polygonal curve Q by computing the radius of a circle that is tangent to P at $\mathbf{p}$ and touches Q at a vertex or an edge. Assuming $\mathbf{p}$ is not a point of intersection between P and Q. We estimate the normal $\widehat{\mathbf{N}}_P(\mathbf{p})$ to P at $\mathbf{p}$ such that it is orthogonal to the line passing through the two neighboring vertices of $\mathbf{p}$ along P and that it points towards the interior of the *gap*. First, for every vertex $\mathbf{q}$ of Q, we compute the r-value for a ball with center $\mathbf{m} = \mathbf{p} + r\widehat{\mathbf{N}}_P(\mathbf{p})$ such that $|\mathbf{m} - \mathbf{p}| = |\mathbf{m} - \mathbf{q}| = r$ as follows:

$$r = -\frac{\overrightarrow{\mathbf{qp}}^2}{2\overrightarrow{\mathbf{qp}} \cdot \widehat{\mathbf{N}}_Q(\mathbf{q})} \tag{3}$$

Notice that we do not need to check explicitly that the direction of vector $\overrightarrow{\mathbf{qm}}$ is a plausible normal to Q at $\mathbf{q}$. If it were not, the *ball-map* for edges of Q incident upon $\mathbf{q}$, computed as explained below, would return a smaller radius.

For every edge Q_i of Q with oriented edge-normal $\widehat{\mathbf{N}}_Q(Q_i)$ and vertices $\mathbf{c}, \mathbf{d}$, we compute the r-value for a ball with center $\mathbf{m} = \mathbf{p} + r\widehat{\mathbf{N}}_P(\mathbf{p})$ as follows:

$$r = \frac{\overrightarrow{\mathbf{cp}} \cdot \widehat{\mathbf{N}}_Q(Q_i)}{1 - \widehat{\mathbf{N}}_P(\mathbf{p}) \cdot \widehat{\mathbf{N}}_Q(Q_i)} \tag{4}$$

The corresponding point $\mathbf{q}$ is then computed as:

$$\mathbf{q} = \mathbf{p} + r\widehat{\mathbf{N}}_P(\mathbf{p}) - r\widehat{\mathbf{N}}_Q(Q_i) \tag{5}$$

The candidate mapping is discarded if $\mathbf{q}$ lies outside the bounds of Q_i.

The minimum r *ball-map* candidate among all vertex-vertex and retained vertex-edge mappings is then selected for point $\mathbf{p}$.

For conciseness, we omit the discussion of singular configurations where the denominators of these equations are 0. We trap these using a numeric tolerance and use trivial formulations for the corresponding singular (parallel or coincident) cases.

The *ball-map* construction for *PLCs* presented above may fail when the curves are insufficiently smooth or

when the sampling process is not sufficiently dense. Our experiments show that the application of a few simple subdivision steps [55] produces curves for which our construction works without problem. In fact, all results in this paper were computed using the *PLC* method described here unless explicitly stated otherwise. When working on *PLCs*, we use a modified version of lacing.

Recall that in the *PCC* lacing algorithm, two points ($\mathbf{p}$ and $\mathbf{q}$) "walk" on each curve, where the maximum step-size is determined by the length of the current circular-arc. Instead, we now set a constant step-size $dStep$. Therefore at each step, $\mathbf{p}'$ and $\mathbf{q}$ are computed by walking along P and Q, respectively, a geodesic distance of $dStep$. Our experiments show that using a step-size that is twice larger than the longest edge of the subdivided curves works well.

6 Measures

We first discuss how we sample space and time. Then, we provide details of the measures used here to compare morphs.

Three of the studied morphs (*linear-interpolation*, *curvature-interpolation*, and *laplace-blending*) assume a given correspondence. For simplicity, we use a uniform arc-length sampling to produce the same number of uniformly distributed samples on each curve. The three *ball-morphs* use the *ball-map* correspondence. The other morphs compute their own correspondence. This sampling disparity makes it difficult to compute measures for a fair comparison.

Consider for example the problem of measuring the average *travel-distance*. This should be the integral of travel distances. The problem is how to fairly select the integration element. If for example we use the *linear-interpolation-morph*, then the average distance measured for a set of uniformly distributed samples will depend on whether we start form P or Q. Since the average *travel-distance* is a property of the mapping, and not the sampling, a measure that so blatantly depends on the sampling is clearly incorrect.

To overcome this problem, each reported measures is the average of two measures, one computed by sampling P and one computed by sampling Q. For the first measure, we sample the departure curve P using a dense set of samples that are uniformly distributed on each curve so as to be separated by a prescribed geodesic distance u. For each sample $\mathbf{p}_i$ on P, we compute the corresponding point $\mathbf{q}_i$ on the arrival curve Q so that $\mathbf{q}_i$ is the image of $\mathbf{p}_i$ by the mapping associated with the particular morphing scheme. We compute a measure m_i associated with the trajectory from $\mathbf{p}_i$ to $\mathbf{q}_i$ and the associated weight $w_i = (|\overrightarrow{\mathbf{p}_{i-1}\mathbf{p}_i}|+|\overrightarrow{\mathbf{q}_{i-1}\mathbf{q}_i}|+|\overrightarrow{\mathbf{p}_i\mathbf{p}_{i+1}}|+|\overrightarrow{\mathbf{q}_i\mathbf{q}_{i+1}}|)/4$. Then, we report the normalized weighted average $(\Sigma w_i m_i)/(\Sigma w_i)$. For the second measure, we sample the arrival curve Q as before using the same geodesic distance u. For each sample $\mathbf{q}_i$ on Q, we compute the corresponding point $\mathbf{p}_i$ on the departure curve P, so that $\mathbf{q}_i$ is the image of $\mathbf{p}_i$ by the mapping associated with the particular morphing scheme. Then, we proceed as above.

We have implemented the following six measures of morph quality.

***Travel-distance*:** For each sample $\mathbf{p}_i$, we measure m_i as the arc length of the trajectory to the corresponding point $\mathbf{q}_i$. Then, as explained above, we report the weighted average of these from P to Q and vice-versa.

***Stretch*:** We define *stretch* $S(P,Q)$ as the average of the integral over time of the stretch factor for an infinitesimal portion of the curve. We compute its discrete approximation as follows. Let $\mathbf{p}$ and $\mathbf{p}'$ be consecutive samples on P. Let $L(\mathbf{p},t)$ be the length of the segment of $P(t)$ between $\mathbf{p}(t)$ and $\mathbf{p}'(t)$. We compute $S(P,Q)$ as

$$\begin{aligned} S(P,Q) = &\sum_{t\in[0,1-\epsilon]}\left(\sum_{\mathbf{p}\in P}|L(\mathbf{p},t+\epsilon)-L(\mathbf{p},t)|\right) \\ + &\sum_{t\in[0,1-\epsilon]}\left(\sum_{\mathbf{q}\in Q}|L(\mathbf{q},t+\epsilon)-L(\mathbf{q},t)|\right) \end{aligned}$$

***Acceleration*:** *Acceleration* is defined as the derivative of the expression of velocity in the local, time-evolving frame, and measures the lack of steadiness [19] of the motion.

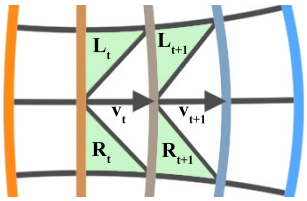


Fig. 18. *Acceleration* (lack of steadiness) for a given vector v of the morph trajectory is computed relative to the neighboring triangles (green).

Let $\mathbf{p}_t$ denote the position of a sample $\mathbf{p}$ at a time t. We approximate the instantaneous velocity of $\mathbf{p}_t$ by the vector $\mathbf{p}_t\mathbf{p}_{t+\epsilon}$. For each such velocity on a morph trajectory, we compute two barycentric coordinate vectors $B_L(\mathbf{p}_t\mathbf{p}_{t+\epsilon})$ and $B_R(\mathbf{p}_t\mathbf{p}_{t+\epsilon})$ relative to the left and right neighboring triangles L_t and R_t as shown in Fig. 18. The steadiness at a point $\mathbf{p}_t$ is then computed as:

$$\begin{aligned} g_t = &\tfrac{1}{2}\|B_L(\mathbf{p}_{t-\epsilon}\mathbf{p}_t)-B_R(\mathbf{p}_{t-\epsilon}\mathbf{p}_t)\| \\ + &\tfrac{1}{2}\|B_L(\mathbf{p}_t\mathbf{p}_{t+\epsilon})-B_R(\mathbf{p}_t\mathbf{p}_{t+\epsilon})\| \end{aligned}$$

We compute the *acceleration* measure m_i as the sum of the g_t terms over the trajectory of each point $\mathbf{p}_i$ and report their weighted average, as described above.

***Distortion*:** At each point along the evolving curve and at each time, the amount of distortion is proportional to $1/cos\theta$, where θ is the angle between the direction of travel and the normal to the evolving curve.

Let $\mathbf{p}$ and $\mathbf{p}'$ be consecutive samples on $P(t)$ and $\mathbf{L}(\mathbf{p},t)$ define the unit vector in the direction $\overrightarrow{\mathbf{p}\mathbf{p}'}$. Let $\mathbf{V}(\mathbf{p},t)$ define the unit vector in the direction $\overrightarrow{\mathbf{p}_t\mathbf{p}_{t+\epsilon}}$. We compute

$$r_i = \sum_{t\in[0,1-\epsilon]} \frac{1}{2}(\mathbf{L}(\mathbf{p},t)+\mathbf{L}(\mathbf{p},t+\epsilon))\cdot\frac{1}{2}(\mathbf{V}(\mathbf{p},t)+\mathbf{V}(\mathbf{p}',t))$$

where $\cdot$ denotes dot product.

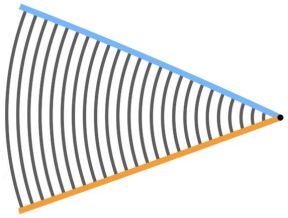


Fig. 19. The *ball-morph* produces a pure rotation with zero *distortion* between linear segments.

It was shown in [1] that the *circular ball-morph* is free from *distortion* when morphing between linear segments (in 2D) of P and Q (Fig. 19).

6.1 Mesh measures

In addition to the 2D measures, we also present results of 3D measures of *average squared mean curvature* and *maximum squared mean curvature* [56] of the resulting triangle mesh surfaces constructed by interpolating the input curves along the *z-axis*. In applications of surface reconstruction from 2D planar contours, smoothness of the resulting reconstruction is often desirable (see Fig. 20).

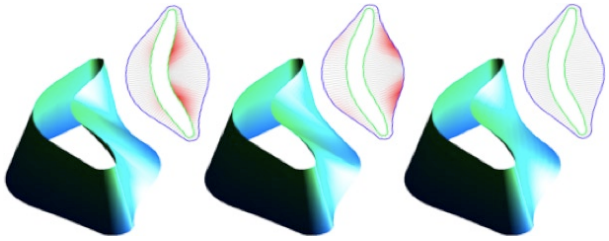


Fig. 20. We show the slice-interpolating surface reconstructed using a *closest-projection-morph* from-green-to-blue (left), the reverse *closest-projection-morph* from-blue-to-green (center), and the symmetric *circular ball-morph* (right) which appears smoother. The amount of local *distortion* is shown in red on the 2D drawings.

7 *Composite ball-morphs*

To produce morphs between curves that are not *b-compatible*, we compute the *relative blendings* [57] $P' = R_Q(P)$ and $Q' = R_P(Q)$ and the *ball-morph* M_0 between the resulting curves P' and Q'. The *relative blendings* of two curves P and Q are computed by trimming away the parts of the curve that violate the conditions of *b-compatibility* and replacing them with circular arcs defined by maximal disks in the *gap* $G(P, Q)$.

This solution produces only the central part of the morph, which must be concatenated in time and possibly on both ends with other extension-morphs that fill the incompatible features. There are 3 situations for computing the next extension-morph (morph M_1):

1) If Q' and Q are *b-compatible* we compute their *ball-morph* M_1.

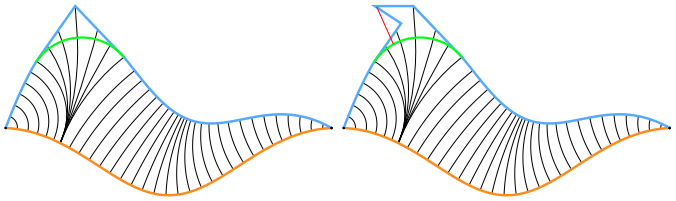


Fig. 21. Left: *ball-map* arcs between the *relative blended* versions of the curves and *closest-projection* linear trajectories from the blue curve to its relative blended version. Right: The *closest-projection-morph* is not a homemorphism, so additional *relative blending* operations will be necessary.

2) Else if the *closest-projection-morph* from Q to Q' is a homeomorphism, we produce an extension-morph M_1 with the reversed straight line trajectories (Figure 21-left)
3) Otherwise, we compute the *relative blending* $Q'' = R_{Q'}(Q)$, then compute the *ball-morph* M_1 between Q' and Q''.

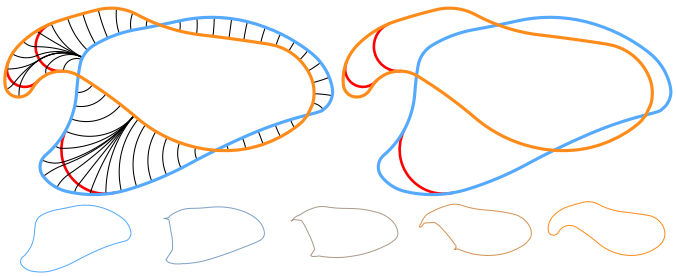


Fig. 22. Simple *composite ball-morph* on curves P (blue) and Q (orange). The circular arcs corresponding to the *relative blending* operations that produce P', Q', and Q'' are shown in red. The final *PCC* trajectories are shown on the right. Three inbetween frames are also shown (bottom).

Note that in all cases, the trajectories of M_1 leave Q' along its local normal and are hence smoothly joined with the trajectories of M_0. If we run into situation (3), we recurse on the gap between Q'' and Q. This process fills the gap between Q' and Q by a series of *ball-morphs* and possibly a final *closest-projection-morph*, generating piecewise-circular trajectories (Fig. 21 left) with possibly a straight line at the end of each one. Note that without handling situation (2), the recursive process would never converge since a ball will never reach the sharp feature. We have produced a concatenation of morphs M_1, M_2, M_3... We perform a similar iterative process to invade the gap between P' and P, producing in this manner a series of morphs, which we reference with negative integers: M_{-1}, M_{-2}, M_{-3}... The final combined morph is: ... M_{-3}, M_{-2}, M_{-1}, M_0, M_1, M_2, M_3 ... Fig. 22 shows a simple example which produces a *composite* of four *circular ball-morphs*.

Although different synchronization approaches are possible, the simplest one is to move each sample at

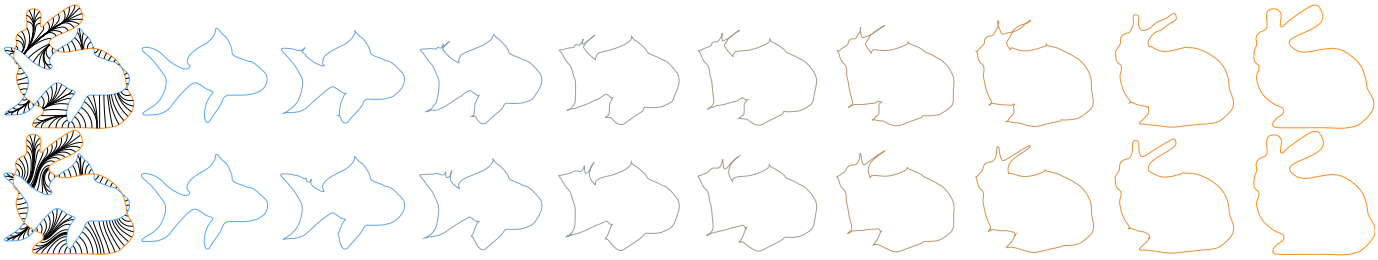


Fig. 23. A comparison between the *composite ball-morph* (top) and the *heat-propagation-morph* (bottom) on curves that are not *b-compatible*. For each morph, the trajectories (left) are sampled uniformly through time to obtain the inbetween curves (right).

a constant speed along its *PCC* trajectory during the desired interval. Note that smooth inbetween curves are not generated when using the *composite ball-morph*. However, the inbetween curves and trajectories are similar to those produced by the *heat-propagation-morph*, as shown in Fig. 23.

8 Results

We first compare the *ball-morphs* to our benchmark set using two different test cases, as shown in Figures 24 and 25. Then, we compare two of the *ball-morphs* to the best two other morphs (*laplace-blending* and *heat-propagation*) on a test case between an apple and a pear. The *closest-projection-morph* is not shown for these tests because the pairs of curves are not *c-compatible* (Fig. 13).

The first test case (Fig. 24) shows a morph between a circle and an ellipse. Our experiments demonstrate that the average *travel-distance* is the shortest when using the *linear ball-morph* and that the *circular ball-morph* has the least amount of *distortion*. Note that the *heat-propagation-morph* is similar in terms of appearance to the *circular ball-morph*. However due to its reliance on a discrete grid and other sampling issues, it is very susceptible to *acceleration* and *squared mean curvature* errors in regions where P and Q are very close relative to the chosen grid size. The minimum *squared mean curvature* measures (maximum and average) are produced by the *curvature* and *laplace-blending-morphs*.

The test case shown in Fig. 25 shows a set of symmetric 'S' shaped curves. This example highlights the strength of the morphs which compute their own correspondence (*heat-propagation*, *ball-morphs*). The other morphs, which define correspondence through uniform arc-length parameterization, exhibit extreme distortion and travel lengths and also produce self-intersections with the original curves. As with the previous example, the *travel-distance* and *distortion* measures are minimized by the *linear* and *circular ball-morphs*, respectively. The *heat-propagation-morph* is very similar in terms of appearance and measure to the family of *ball-morphs* and produces meshes with the smallest values of maximum and average *squared mean curvature*.

The test case in Fig. 26 uses contours representing an apple and a pear. We show the "best" four morphs (*linear ball-morph*, *circular ball-morph*, *heat-propagation* and *laplace-blending*) and compare their measures. Travel distance and distortion are still minimized for the *linear* and *circular ball-morphs*, respectively. The *laplace-blending* approach performs best in terms of *stretch*. The *ball-morphs* perform the best in terms of *acceleration* and the worst in terms of *squared mean curvature* of the resulting meshes.

The *parabolic ball-morph* is barely distinguishable from the *circular* one, even though the surface it produces has a higher *maximum squared mean curvature*. It may be preferred in some applications, where the non-rational quadratic parameterization of the trajectories simplifies numeric calculations.

9 Conclusion

We have proposed a family of morphs between curves which are *b-compatible*. All are based on variations of the medial axis construction. We have compared them to one another and to several other simple morphs. We used four measures of morph quality in our comparison, as well as surface measures for comparing them as surface reconstruction techniques.

Although the *heat-propagation-morph* produces very similar results to the *ball-morphs*, it has the disadvantage of requiring rasterization to a grid and a PDE solve. However, this method easily maps to more extreme cases that are not *b-compatible* without need for special extensions (other than a higher resolution grid).

We conclude that for the cases of *b-compatible* shapes, the *ball-morphs* offer a precise and desirable result in terms of *distortion*, *travel-distance*, as well as *curvature*.

Ball-morphs have many advantages. For example [1], the *circular ball-morph* produces curves of C^{k-1} continuity that do not intersect one another for input curves of C^k for $k \geq 2$.

Acknowledgement

This work has been partially supported by NSF grant 0811485 and by the Walt Disney Corporation.

References

[1] F. Chazal, A. Lieutier, J. Rossignac, and B. Whited, "Ball-map: Homeomorphism between compatible surfaces," *International Journal of Computational Geometry and Applications*, April 2010.

[2] J. Gomes, L. Darsa, B. Costa, and L. Velho, *Warping and morphing of graphical objects*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998.

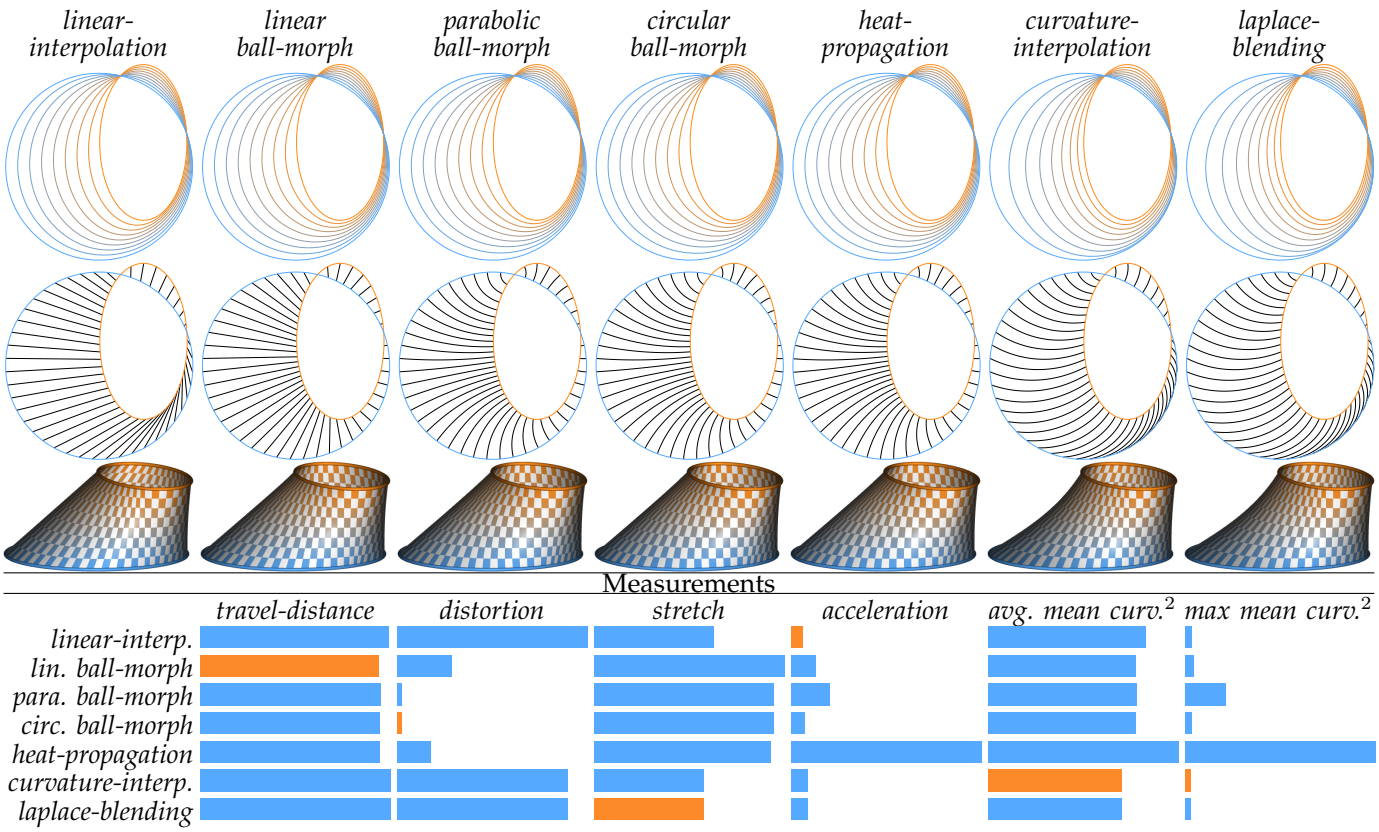


Fig. 24. Morph results for a circle and ellipse, showing the morph curves (top), the morph trajectories (middle) and the surface created by sweeping the evolving curve, changing its height at a constant rate (bottom). Also displayed are the measures for each morph with the smallest (best) value of each measure highlighted in orange.

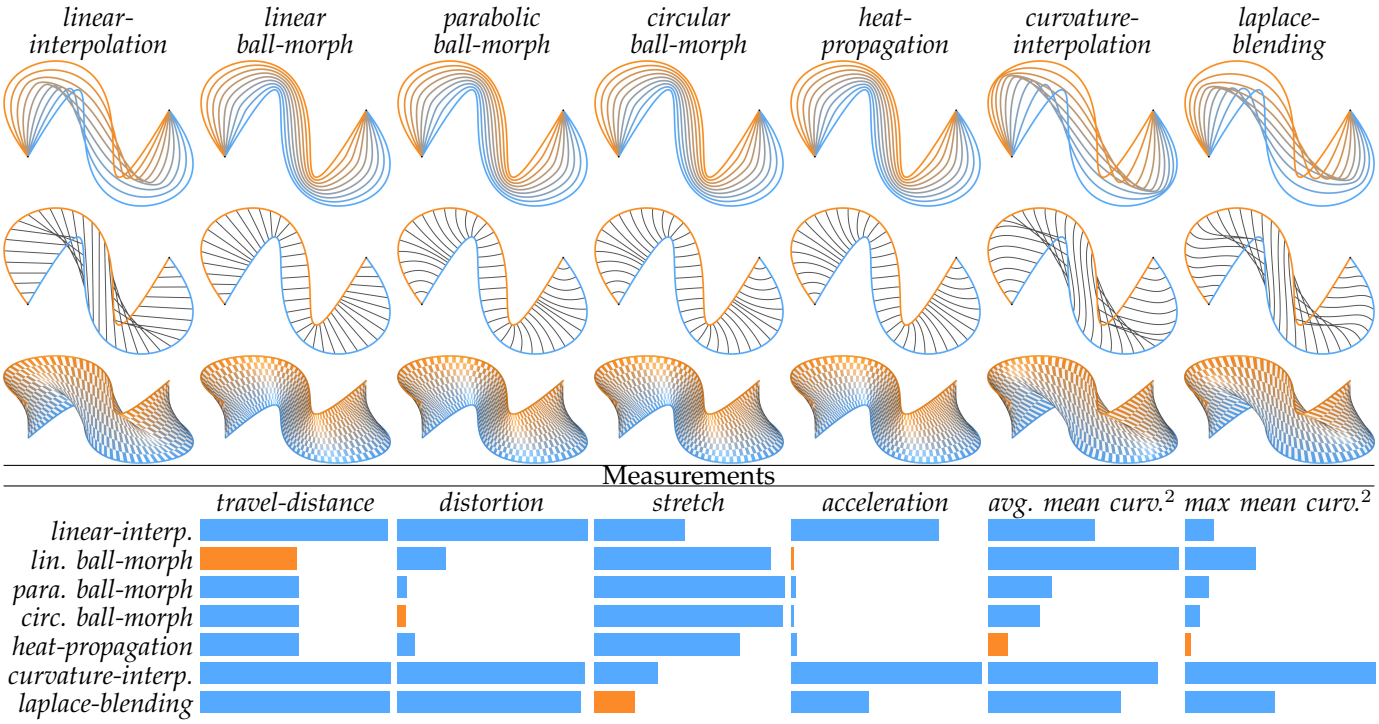


Fig. 25. Morph results for a set of 'S'-shaped curves, showing the morph curves (top), the morph trajectories (middle) and the surface created by sweeping the evolving curve, changing its height at a constant rate (bottom). Also displayed are the measures for each morph with the smallest (best) value of each measure highlighted in orange. Note that some of the morphs do not remain within the bounds of the input curves.

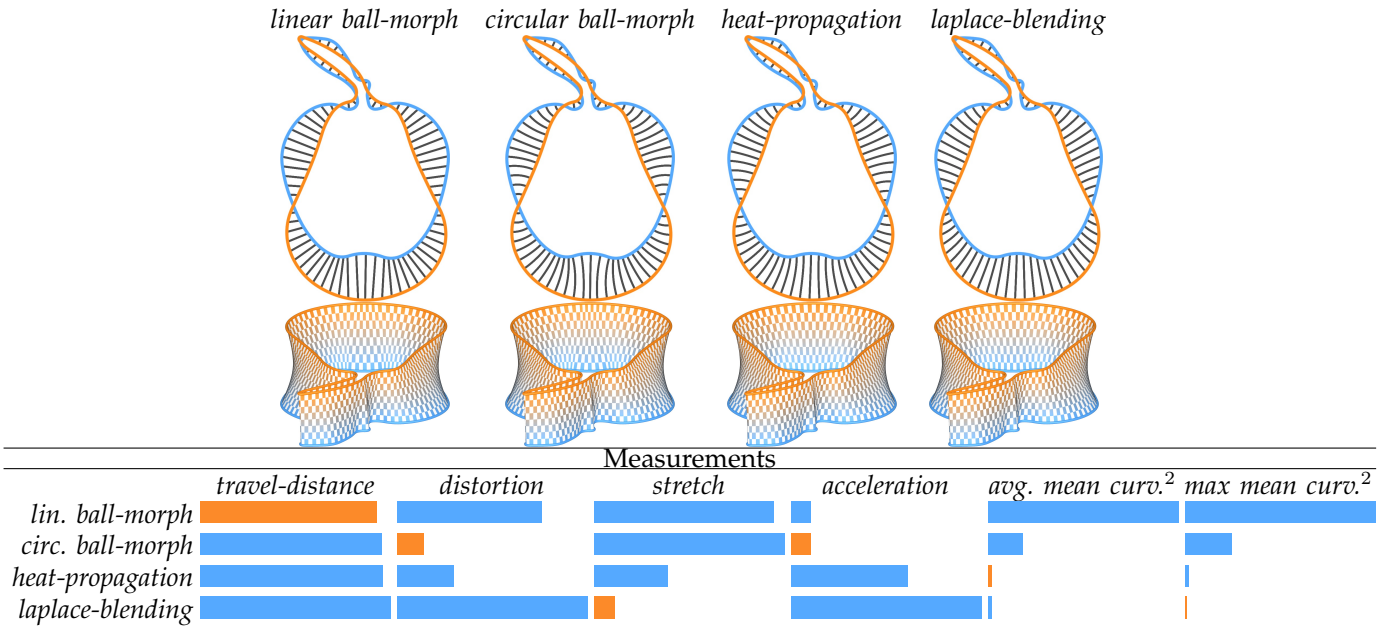


Fig. 26. Morph results for a set of apple and pear shaped curves, show the morph trajectories (top) and the surface created by sweeping the evolving curve, changing its height at a constant rate (middle). Also displayed are the measures for each morph with the smallest (best) value of each measure highlighted in orange.

[3] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 157–164.

[4] E. Catmull, "The problems of computer-assisted animation," *SIGGRAPH Comput. Graph.*, vol. 12, no. 3, pp. 348–353, 1978.

[5] F. Thomas and O. Johnston, *The Illusion of Life: Disney Animation*, revised ed. Disney Editions, 1995.

[6] A. Kort, "Computer aided inbetweening," in *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*. New York, NY, USA: ACM, 2002, pp. 125–132.

[7] W. T. Reeves, "Inbetweening for computer animation utilizing moving point constraints," *SIGGRAPH Comput. Graph.*, vol. 15, no. 3, pp. 263–269, 1981.

[8] B. Whited, G. Noris, M. Simmons, R. Sumner, M. Gross, and J. Rossignac, "Betweenit: An interactive tool for tight inbetweening," *Comput. Graphics Forum (Proc. Eurographics)*, vol. 29, no. 2, pp. 605–614, 2010.

[9] S. E. Chen and R. E. Parent, "Shape averaging and it's applications to industrial design," *IEEE Comput. Graph. Appl.*, vol. 9, no. 1, pp. 47–54, 1989.

[10] S.-W. Cheng and T. K. Dey, "Improved constructions of delaunay based contour surfaces," in *SMA '99: Proceedings of the fifth ACM symposium on Solid modeling and applications*. New York, NY, USA: ACM, 1999, pp. 322–323.

[11] G. Barequet and M. Sharir, "Piecewise-linear interpolation between polygonal slices," *Comput. Vis. Image Underst.*, vol. 63, no. 2, pp. 251–272, 1996.

[12] S. Akkouche and E. Galin, "Implicit surface reconstruction from contours," *Vis. Comput.*, vol. 20, no. 6, pp. 392–401, 2004.

[13] H. Fuchs, Z. M. Kedem, and S. P. Uselton, "Optimal surface reconstruction from planar contours," *Commun. ACM*, vol. 20, no. 10, pp. 693–702, 1977.

[14] D. Cohen-Or, A. Solomovic, and D. Levin, "Three-dimensional distance field metamorphosis," *ACM Trans. Graph.*, vol. 17, no. 2, pp. 116–141, 1998.

[15] G. Barequet and A. Vaxman, "Nonlinear interpolation between slices," in *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*. New York, NY, USA: ACM, 2007, pp. 97–107.

[16] N. C. Gabrielides, A. I. Ginnis, P. D. Kaklis, and M. I. Karavelas, "G1-smooth branching surface construction from cross sections," *Comput. Aided Des.*, vol. 39, no. 8, pp. 639–651, 2007.

[17] A. Efrat, S. Har-Peled, L. J. Guibas, and T. M. Murali, "Morphing between polylines," in *SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2001, pp. 680–689.

[18] K. Siddiqi and S. Pizer, *Medial Representations: Mathematics, Algorithms and Applications*. Springer, 2008, 450pp. In Press.

[19] J. Rossignac and A. Vinacua, "Sam: Steady affine morph," *to appear in the IEEE Transactions on Computer Graphics and Visualization*, 2010.

[20] A. H. Barr, "Global and local deformations of solid primitives," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1984, pp. 21–30.

[21] T. Beier and S. Neely, "Feature-based image metamorphosis," *SIGGRAPH Computer Graphics*, vol. 26, no. 2, pp. 35–42, 1992.

[22] E. Ezra, M. Sharir, and A. Efrat, "On the icp algorithm," in *SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry*. New York, NY, USA: ACM, 2006, pp. 95–104.

[23] F. Mokhtarian, S. Abbasi, and J. Kittler, "Efficient and robust retrieval by shape content through curvature scale space," in *Proc. International Workshop IDB-MMS96*, 1996, pp. 35–42.

[24] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2005, p. 197.

[25] G. Mori, S. Belongie, and J. Malik, "Efficient shape matching using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1832–1837, 2005.

[26] S. Wang, Y. Wang, M. Jin, X. Gu, and D. Samaras, "3d surface matching and recognition using conformal geometry," in *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2453–2460.

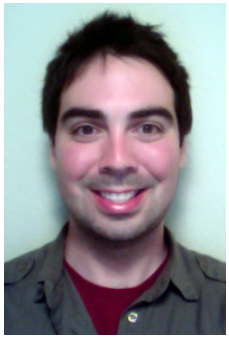
[27] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe, "Inter-surface mapping," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*. New York, NY, USA: ACM, 2004, pp. 870–877.

[28] M. Cui, J. Femiani, J. Hu, P. Wonka, and A. Razdan, "Curve matching for open 2d curves," *Pattern Recogn. Lett.*, vol. 30, no. 1, pp. 1–10, 2009.

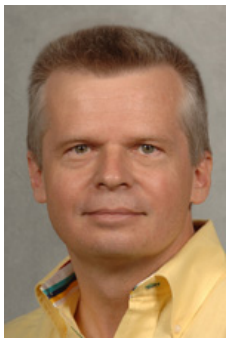
[29] R. W. Sumner, M. Zwicker, C. Gotsman, and J. Popović, "Mesh-based inverse kinematics," in *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*. New York, NY, USA: ACM, 2005, pp. 488–495.

[30] T. W. Sederberg and E. Greenwood, "A physically based approach to 2–d shape blending," in *SIGGRAPH '92: Proceedings of the 19th*

annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, 1992, pp. 25–34.
[31] T. W. Sederberg, P. Gao, G. Wang, and H. Mu, "2-d shape blending: an intrinsic solution to the vertex path problem," in *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1993, pp. 15–18.
[32] M. Alexa, "Differential coordinates for local mesh morphing and deformation," *The Visual Computer*, vol. 19, pp. 105–114, 2003.
[33] D. Xu, H. Zhang, Q. Wang, and H. Bao, "Poisson shape interpolation," in *SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling*. New York, NY, USA: ACM, 2005, pp. 267–274.
[34] H. Fu, C.-L. Tai, and O. K. Au, "Morphing with laplacian coordinates and spatial temporal texture," in *Proc. Pacific Graphics '05*, 2005, pp. 100–102.
[35] G. Cong and B. Parvin, "A new regularized approach for contour morphing," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 1, 2000, pp. 458–463 vol.1.
[36] B. Aronov, R. Seidel, and D. Souvaine, "On compatible triangulations of simple polygons," *Comput. Geom. Theory Appl.*, vol. 3, no. 1, pp. 27–35, 1993.
[37] Y. Weng, W. Xu, Y. Wu, K. Zhou, and B. Guo, "2d shape deformation using nonlinear least squares optimization," *Vis. Comput.*, vol. 22, no. 9, pp. 653–660, 2006.
[38] H. Guo, X. Fu, F. Chen, H. Yang, Y. Wang, and H. Li, "As-rigid-as-possible shape deformation and interpolation," *J. Vis. Comun. Image Represent.*, vol. 19, no. 4, pp. 245–255, 2008.
[39] M. Shapira and A. Rappoport, "Shape blending using the star-skeleton representation," *IEEE Comput. Graph. Appl.*, vol. 15, no. 2, pp. 44–50, 1995.
[40] W. Che, X. Yang, and G. Wang, "Skeleton-driven 2d distance field metamorphosis using intrinsic shape parameters," *Graphical Models*, vol. 66, no. 2, pp. 102–126, 2004.
[41] R. Klein, A. Schilling, and W. Straβer, "Reconstruction and simplification of surfaces from contours," in *PG '99: Proceedings of the 7th Pacific Conference on Computer Graphics and Applications*. Washington, DC, USA: IEEE Computer Society, 1999, p. 198.
[42] X. Chen, M. R. Varley, L.-K. Shark, G. S. Shentall, and M. C. Kirby, "An extension of iterative closest point algorithm for 3d-2d registration for pre-treatment validation in radiotherapy," in *MEDIVIS '06: Proceedings of the International Conference on Medical Information Visualisation–BioMedical Visualisation*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 3–8.
[43] J. Rossignac and A. Kaul, "Agrels and bips: Metamorphosis as a bézier curve in the space of polyhedra," *Comput. Graph. Forum*, vol. 13, no. 3, pp. 179–184, 1994.
[44] R. Tilove, "Set membership classification: A unified approach to geometric intersection problems," *Computers, IEEE Transactions on*, vol. C-29, no. 10, pp. 874–883, Oct. 1980.
[45] F. Chazel, A. Lieutier, and J. Rossignac, "Orthomap: Homeomorphism-guaranteeing normal-projection map between surfaces," in *ACM Symposium on Solid and Physical Modeling (SPM)*, 2005, pp. 9–14.
[46] F. Chazal, A. Lieutier, and J. Rossignac, "Normal-map between normal-compatible manifolds," *Int. J. Comput. Geometry Appl.*, vol. 17, no. 5, pp. 403–421, 2007.
[47] M. Guthe, P. Borodin, and R. Klein, "Fast and accurate hausdorff distance calculation between meshes," *Journal of WSCG*, vol. 13, no. 2, pp. 41–48, February 2005.
[48] H. Federer, *Geometric Measure Theory*. Springer Verlag, 1969.
[49] J. Rossignac and A. A. G. Requicha, "Piecewise-circular curves for geometric modeling," *IBM J. Res. Dev.*, vol. 31, no. 3, pp. 296–313, 1987.
[50] M. Foskey, M. Lin, and D. Manocha, "Efficient computation of a simplified medial axis," in *ACM Symposium on Solid Modeling and Applications*, 2003, pp. 96–107.
[51] Y. Yang, O. Brock, and R. N. Moll, "Efficient and robust computation of an approximated medial axis," in *SM '04: Proceedings of the ninth ACM symposium on Solid modeling and applications*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2004, pp. 15–24.
[52] H. Asada and M. Brady, "The curvature primal sketch," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 1, pp. 2–14, 1986.
[53] M. Leyton, *Symmetry, Causality, Mind*. MIT Press, 1992.
[54] B. Whited, "Tangent-ball techniques for shape-processing," Ph.D. dissertation, College of Computing, Georgia Institute of Technology, Atlanta, GA, 2009.
[55] J. Rossignac and S. Schaefer, "J-splines," *Computer Aided Design*, vol. 40, no. 10-11, pp. 1024–1032, 2008.
[56] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," CalTech, Tech. Rep.
[57] B. Whited and J. Rossignac, "Relative blending," *Computer Aided Design*, vol. 41, no. 6, pp. 456–462, 2009.



Brian Whited is a recent Ph.D. graduate in Computer Science from Georgia Tech and is now a Senior Software Engineer and Researcher at Walt Disney Animation Studios. His research interests include geometric representation, design and visualization as well as animation, morphing and interpolation. During his Ph.D. he worked as a research intern for both Siemens Coroporate Research and Walt Disney Animation Studios, which resulted in 4 joint patents and 5 peer-reviewed articles in addition to 6 other published works in the areas of computational geometry and interactive surgery planning.



Jaroslaw (Jarek) Rossignac is a Full Professor of Computing at Georgia Tech. His research focuses on the design, representation, simplification, compression, analysis and visualization of highly complex 3D shapes, structures, and animations. Before joining Georgia Tech in 1996 as the Director of the GVU Center, he was Senior Manager and Visualization Strategist at the IBM T.J. Watson Research Center. He holds a Ph.D. in E.E. from the University of Rochester, a Diplme d'Ingnieur from the Ecole Nationale Suprieure en lectricit et Mcanique (ENSEM), and a Matrise in M.E. from the University of Nancy, France. He authored 25 patents and 130 peer-reviewed articles for which he received 23 Awards. He created the ACM Solid Modeling Symposia and expanded them into the annual Solid and Physical Modeling (SPM) conferences; chaired 30 conferences and program committees; delivered about 30 Distinguished or Invited Lectures and Keynotes; and served on the Editorial Boards of 7 professional journals and on 74 Technical Program committees. Currently he heads the NSF Aquatic Propulsion Lab (APL) and the Modeling, Animation, Graphic, Interaction, and Compression (MAGIC) Lab at Georgia Tech, which hosts the Disney-sponsored Feature Animation Production Automation (FAPA) project. Rossignac is a Fellow of the Eurographics Association and the Editor-in-Chief of GMOD (Graphical Models).