Available online at www.sciencedirect.com
SCIENCE DIRECT®
Graphical Models 66 (2004) 102–126
Graphical Models
www.elsevier.com/locate/gmod

Skeleton-driven 2D distance field metamorphosis using intrinsic shape parameters

WuJun Che,* XunNian Yang, and GuoZhao Wang

Department of Mathematics, Institute of computer graphics and image processing, Zhejiang University, Hangzhou 310027, PR China

Received 15 February 2003; accepted 18 November 2003

Abstract

In this article a novel algorithm is presented for 2D shape interpolation using the intrinsic shape parameters of a piecewise linear curve. The skeletons of two given shapes are computed and the smooth transformation of distance fields is driven by metamorphosis from the skeleton of the source object to that of the target one. We introduce feature graphs, linear forms of skeletons, to guide the construction of intermediate skeleton. If the topologies of the source object and the target one are different, their feature graphs will be automatically extended with equivalent topologies. Then we apply the technique of intrinsic shape parameters to the smooth transition of the extended feature graphs, which will guide the metamorphosis of the skeletons. Not only can the new approach be capable of morphing between objects with different topological genus, but also the topologies and the shapes of the intermediate objects can be controlled efficiently.

Keywords: Skeleton; Medial axis transformation; Distance transformation; Morphing; Shape interpolation; Shape blending

* Corresponding author.
E-mail addresses: zdgao@163.net (W. Che), yxn@zju.edu.cn (X. Yang), wgz@math.zju.edu.cn (G. Wang).

1524-0703/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.gmod.2003.11.001

1. Introduction

Morphing or Metamorphosis, is a fluid transformation and gradual interpolation from one shape to another. It has received much attention in recent years because of its wide application to movie and television, computer animation, computer graphics, and industrial design. Shape interpolation is also playing an important role even in the bio-medical field, where high veracity is required.

1.1. The problem

Given two shapes, there are incalculable transformations that take one shape into the other. Although it difficult to define an intrinsic morphing sequence between two arbitrary shapes, an intuitive solution clearly exists to equal human perception. The metamorphosis should be smooth, and it should keep as much as possible of the two shapes during the transformation. The morphing problem is usually dealt with as two subproblems. The first one is to find a correspondence between primitives of the two shapes. The second one is to find trajectories that corresponding primitives travel during the morphing process.

A reasonable correspondence between the features of two given shapes is a prerequisite to produce an acceptable intermediate sequence. Of course it is a purely subject aesthetic criteria and relies on the context in which the transformation is performed. For this reason, user input is crucial for good morphs of the objects. Feature matching is related to the significant intrinsic characteristic of the objects, or even the way as the user wants to be. Moreover, we believe that a reasonable correspondence involves not only the correspondence of geometric elements, such as feature point and feature line, but the correspondence of topological information as well. Take, for an instance in Fig. 1. It is a transformation between two shapes of 'Δ' and 'Λ,' in which a reasonable vertex correspondence pattern should be from: v_A^i to v_B^i $(i = 1, 2, 3)$. However, there are at least two morphing sequences satisfying

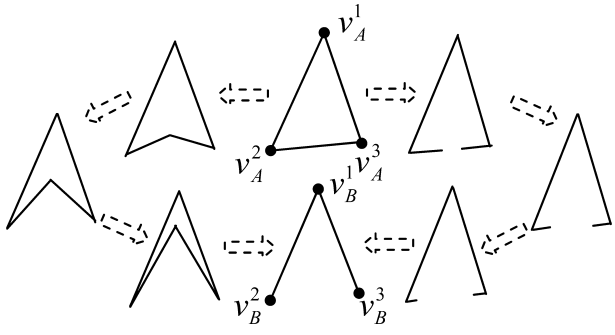

Fig. 1. Morphing between figures and the two morphing schemes are acceptable although they possess the same vertex correspondence: $v_A^i \leftrightarrow v_B^i$, $i = 1, 2, 3$.

our subjective aesthetic criteria and no one dissents in choosing such a suitable correspondence manner.

The key to such multi-feasibility of feature correspondence is how to distinguish between the differences of their topologies. We adopt medial axis to represent the geometric structure of a shape. One property of the medial axis (transform) is that it preserves the topological information of the original domain and they are homotopically equivalent.

In Section 1.2 we will review some related works and will give an overview of our algorithm in Section 1.3.

1.2. Previous works

The representation types have a strong impact on the algorithms for shape transformation, where polygons in 2D or polyhedrons in 3D are the popular geometric models in computer graphics at present. The algorithms for polygon or polyhedron morphing usually consist of two steps: establishing a mapping from each point of the source object to a point on the target one, then interpolating between each pair of corresponding points. More details about variants of mesh morphing algorithms developed up to now can be found in a survey [1].

Mesh morphing can produce surprising effect in virtue of technology of computer graphics, but it will get into trouble in the case of two shapes which are not topologically homeomorphous, partly because the current methods are based on the parametrization of the shapes and are difficult to address the correspondence issue. That is to say, performing topological changes to a model can be challenging with a surface description alone.

Sederberg et al. [17] presented a shape blending method using intrinsic variables, in which interpolated entities are edge lengths and angles between edges rather than vertex locations, and this method is also referred as edge-angle interpolation. In [12], the feature curves of the objects are connected by dependency graphs and the intermediate features are generated by applying this method to the two graphs. By decomposing two polygons into equivalent star-shaped pieces associated with each other by skeleton, respectively, Shapira and Rappoport [18] proposed a method that the trajectories of star-shaped pieces are controlled by skeleton via edge-angle interpolation. Blanding generalized the idea of skeleton to the 3D case, in which the topological difference is no longer sensitive as before [5]. The trimmed skeleton of the symmetric difference between two polygons is the intermediate shape.

Techniques that use implicit function interpolation can handle the change in topologies gracefully and self-intersection surface is avoided [21]. But the user has little control over the intermediate shapes. In [8], intermediate objects are constructed by a distance field metamorphosis and the interpolation of the distance field is controlled by a set of correspondence anchor points with the aid of a warp function. In another paper [21], the solution to the shape transformation problem is solved as a problem of scattered data interpolation and it can be generalized to any number of dimensions. Another piece of related work in shape interpolation by distance field was presented in [16], in which the model of object representation is Union of Circles (UoC).

Barequet and Sharir [3] presented a novel algorithm for piecewise-linear surface reconstruction from a series of parallel polygonal cross sections. This algorithm handles slices with multiple nested polygons and does not rely on any resemblance between them. But if the topologies of two successive polygons are different, it will abruptly change just at the beginning or at the end of the morph sequence, rather than evolving gradually. Surazhsky et al. [19] presented an improved method for morphing between two polygonal shapes on the plane, in which the resulting surface does contain any horizontal triangles and guarantees that all the topology changes occur at mid-height.

Beier and Neely [4] proposed a feature-based metamorphosis method. An object is treated as a collection of points that could be influenced by feature lines having a field of influence. Using this method, an animator begins with establishing correspondence with pairs of line segments between two images. The warp between these images is determined from the local coordinate systems of a pair of feature line segments or their weighted average if more pairs are specified. Lerios et al. [13] extended the feature-based image metamorphosis technique to the metamorphosis of 3D solid objects based on their geometric features. This method allows the operator to treat 3D geometry in a similar way as 2D image, but more types of feature elements have to be designed.

These mentioned methods offer the advantage that they are insensitive to the different topologies of objects during the transformation [3–5,8,13,16,19,21]. But they suffer from the disadvantage that the topologies of intermediate shapes are hard to be controlled. As pointed out above they cannot be exactly specified just by the feature correspondence or anchor points.

There are some methods aiming at topological evolution. Takahashi et al. [20] did research on it about 3D mesh and presented an approach to explicitly and precisely control a topological transition by inserting and intermediating shape called a key-frame in between the input 3D meshes at the point where the topological transition should take place. The resultant 3D mesh is created by interpolating the source meshes and the key-frame with a tetrahedral 4D mesh and then intersecting the interpolating mesh with another 4D hypersurface. This procedure cannot provide the users with an intuitive approach to the problem for practical use. A method is presented based on the modeling of Reeb-based construction for complex shapes [10]. Similar to the skeleton of a shape, the Reeb graph plays an important role in controllable topological changes, as it does here, so the evolution of topology can be specified by the transformation of the Reeb graph and it allows users to more precisely and more intuitively design how the morph should be. One drawback of this method is that the generation of Reeb graph is sensitive to the selection of the height axis.

1.3. Overview

The main contribution of this article is a novel algorithm for planar shape morphing with controllable interpolation of topologies. The general idea is first to compute the feature graphs of the skeletons when the medial axes of two given shapes

are obtained and the correspondence between the resulting skeletons have been established; then we can construct the isomorphic feature graphs and generate the in-between ones via intrinsic shape parameters; the intermediate feature graphs will guide the metamorphosis of the skeletons and we obtain the blending shapes with combination of their corresponding distance fields. The main steps are listed as below:

1.3.1. Medial axis transformation

There is a rich history of research on skeletonization, and a complete discussion of its details is beyond the scope of this paper. A brief account of medial axis transformation is given in Section 2.1 and the readers interested in it are referred to extensive literature on this problem. Many algorithms can be applied to the skeletal computation of a shape [6,9,11,14]. In this paper, the skeleton of a shape are calculated as presented in [6]. Based on the image thinning technology, we compute the initial skeleton with the same topology of the object; then the initial skeleton is deformed and led to its accurate locations in distance field using Snake model technique. Thus the generated skeleton is not only locating at accurate positions, but also with correct connectivity, and topology.

1.3.2. Feature correspondence of skeletal graphs

The correspondence problem is solved as a skeleton matching process. A skeleton is a geometric graph, called skeletal graph here. In a sense, it is difficult to access objectively the quality of a given match, being true of human's aesthetic value. The user interaction is indispensable to achieve more spectacular, impressive, and accurate results. We perform the feature correspondence of skeletal graphs in computer-aided manual manner.

1.3.3. Feature graph

Once the feature correspondence of skeletons is specified, we then create the skeletal feature graphs, linear forms of the skeletons, based upon the corresponding feature vertices. In this paper our discussion will focus on the topic, including definition, isomorphic construction, and intermediate generation. The details will be described in Sections 3–5.

1.3.4. Skeleton-driven morphing

With the aid of intermediate feature graphs, the in-between skeletons can be obtained and drive the medial axis transformation of the two original shapes to reconstruct the intermediate shapes. Furthermore, we show that the solution has a desirable property that the intermediate skeletal graphs are symmetric with respect to time step.

In the next section we describe some preliminary works which will be used in the following sections. In Section 3 we introduce the concept of feature graph for a skeletal graph. The detailed algorithms of how to construct two isomorphic feature graphs and the in-between graphs will be given in Sections 4 and 5. In Section 6 we present the algorithm for the skeleton-driven metamorphosis of distance fields

Table 1
Glossary

Symbol	Description
$\wedge$, $\vee$	logical AND and OR operators
A or B	object
∂A	boundary of A
$MA(A)$	medial axis of A
$MAT(A)$	medial axis transformation of A
G	skeletal graph
V	vertex set
E	edge set
$\bar{V}$	feature vertex set
$\bar{G}$	extended geometric graph of G after feature correspondence
T	feature graph of G, a linear form of $\bar{G}$
Λ'	temporary denotation of Λ during some operations, such as vertex split, edge correspondence
C	a curve or an edge of a skeletal graph
v or $\bar{v}$	vertex
ℓ	edge
$\|V\|$	element number of V if V is a set or length if V is a vector
$\bar{\varphi}_A$	mapping of vertex correspondence on $\bar{V}_A$ **to** $\bar{V}_B$
F	relation: x and y are in the same family if xFy holds
R	denotation: it is denoted as xRy that the families of x and y are adjacent
S	extended geometric graph of $\bar{G}$ after vertex split and edge correspondence
$\Lambda_A(t)$	intermediate counterpart from Λ_A to Λ_B at time t, where Λ_A and Λ_B are corresponding variables, say intrinsic angles, about A and B, respectively

under the guide of feature graphs and give a compact proof of its symmetric property with respect to time step. Examples and conclusion are shown in the last two sections. And notation is summarized in Table 1.

2. Preliminaries

In this section we will review some related techniques and algorithms upon which our solution is based.

2.1. Skeletal graph

The term skeleton, or medial axis, has been popularly used to describe a line-thinned caricature of the binary image which summarizes its shape and conveys information about its size, orientation, and connectivity. There are quite a few equivalence definitions for the skeleton of a shape in a plane, such as the prairie fire model, the skeleton points being the locations where the propagating wavefront initiated on the shape boundary "intersects" itself [11].

Let A be a domain in a plane. We denote the medial axis of A as $MA(A)$. It is defined as

$$MA(A) = \{p \mid p \in A \ \wedge (\ \exists q_1, q_2 \in A \wedge q_1 \neq q_2 \Rightarrow d(p, q_1) = d(p, q_2))\}.$$

Here, $d(p, q)$ is the Euclidian distance of p and q.

The medial axis transformation of a domain A is the ordered pairs of points in $MA(A)$ and corresponding distances to the boundary of A, say, ∂A. This is

$$MAT(A) = \left\{ (p, r) \;\middle|\; p \in MA(A) \ \wedge \ r = \inf_{q \in \partial A} d(p, q) \right\}.$$

A very important property of medial axis transform is the ability to reconstruct the object boundary by the reverse transformation.

A skeleton defines a geometric graph, whose vertices correspond to branching points or leaf points and whose edges correspond to curves connecting two vertices on which there exist no vertices except for their end-points. In this paper, we mean the medial axis of a domain by the term skeletal graph without special declaration.

For simplicity and clarification, we assume that any two edges do not share the same end-point pair. That is, there is only one edge between two adjacent vertices. If not, new vertices will be inserted to satisfy this assumption. Furthermore, no vertices in the original skeletal graph coincide.

2.2. Morphing between two curves

Let C_A and C_B be two simple curves without self-intersection, the morphing between them can be performed by establishing correspondence between feature points and tracing their travel trajectories.

We can work out the correspondence between C_A and C_B in the same way of [7], which indicated that an analytic solution to the optimization problems of matching is too difficult in the general case of C_A and C_B. One can provide an approximative solution over the discrete sample sets of the two curves, which are determined by polygonalizing the curves with a simple recursive divide and conquer algorithm. After correspondence calculation, they are merged to form a new discrete sample set for each curve. Let $\{v_A^0, v_A^1, \ldots, v_A^n\}$ and $\{v_B^0, v_B^1, \ldots, v_B^n\}$ be such one-to-one matching discrete sample sets of $C_A$and C_B, respectively. In this paper, we represent C_A and C_B approximatively as piecewise linear curves or spline curves interpolating sample points, rather than the curves themselves.

We write the intermediate curve from C_A to C_B as $C_A(t) = \{v_A^0(t), v_A^1(t), \ldots, v_A^n(t)\}$. It can be computed using edge-angle technique, but an equality constraint is enforced by $v_A^n(t) - v_A^0(t) = X_A(t)$, where $X_A(t)$ is achieved in another way (see Section 5). Its detail can be found in (Appendix A).

3. Feature graph

Let G_A and G_B be the skeletal graphs of two original shapes A and B, respectively. The vertex sets of $G_A$and G_B can be denoted as V_A and V_B. In this section we aim at defining the feature graphs of G_A and G_B. It concerns vertex correspondence deeply.

3.1. Vertex correspondence

The necessity for aligning prominent features is evident. In our algorithm, V_A and V_B are such features.

Generally speaking, graph matching is a difficult computational problem and some works addressed this issue to some extent [10,15,16]. In [15], the comparison between two skeletons focus on branching node, the similarity of which is defined in terms of the combination of their incident arcs which maximizes the sum of the related similarity contributes. The branching nodes in the two skeletons are matched by their similarity via visiting the graphs from these matched nodes until nodes belonging to the boundary are reached. Yet this algorithm requires that the nodes should have at most degree 3. Kanongchayos et al. [10] applied a back-tracking algorithm to the vertex classification and then a digraph isomorphism algorithm to the correspondence problem of skeletal graphs.

In this paper, we use a totally automatic matching method presented in [16]. In this method, the distance between every circle of A and B are firstly calculated after computation of UoC and alignment. Then a bipartite graph, where nodes correspond to the circles in A and B, respectively, and the weight on the edges are the distance between them, is defined. A maximum match can be solved by reducing it to a networking flow problem. This matching process procures a good solution to such a problem when the shapes of A and B are similar but it could fail if they are not. One feasible choice is to manually introduce some key corresponding vertices to break A and B down into several shape components. And the global correspondence is established among corresponding subgraphs (see Section 7 for an example). After the above match process, there are still some remaining vertices of A (or B) having no counterparts. In [16], these remaining vertices are matched to those in B which matches their nearest neighbor in A. But this is not a good strategy here because the distance between two neighbor vertices is probably too big. We use the method presented in [7] to establish the approximated correspondence of the unmatched vertices according to their matched neighbor vertices. As showed in Fig. 2, the remaining vertex v_A^3 could be mapped to v_B^1 or v_B^2 using the strategy of [13]. But for better visual effects, it should be matched to a more appropriate point v_B^3, which is not a vertex of B.

Suppose $\varphi_A : V_A \to G_B$ is a desirable correspondence, mapping every vertex in V_A to its corresponding target in G_B during the transformation from A to B. In the same way we can define $\varphi_B : V_B \to G_A$. Unfortunately, φ_A and φ_B are not always mappings between V_A and V_B, as mentioned above.

Now we define

$$\bar{V}_B = V_B \cup \varphi_A(V_A), \quad \bar{V}_A = V_A \cup \varphi_B(V_B).$$

Then the mappings of vertex correspondence, $\bar{\varphi}_A$ and $\bar{\varphi}_B$, are written as extensions of φ_A to domain $\bar{V}_A$ and of φ_B to $\bar{V}_B$, respectively:

$$\bar{\varphi}_A : \bar{V}_A \to \bar{V}_B, \quad \bar{\varphi}_B : \bar{V}_B \to \bar{V}_A.$$

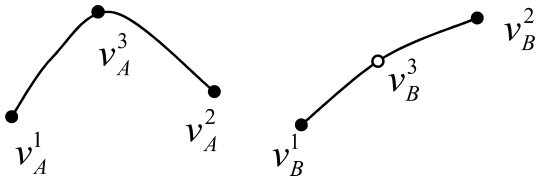

Fig. 2. The correspondence of the unmatched vertex. v_A^i is matched to $v_B^i (i = 1, 2)$ but v_A^3 has no corresponding vertex.

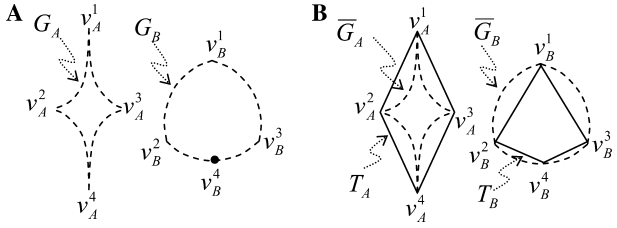

Fig. 3. (A) Original skeletal graphs. The vertex set of G_A is $V_A = \{v_A^i | i = 1, 2, 3, 4\}$ and that of G_B is $V_B = \{v_B^i | i = 1, 2, 3\}$. Here v_B^4 is not a vertex of G_B. (B) Isomorphic skeletal graphs (dashed) and feature graphs (solid). The vertex set of $\bar{G}_A$ is $\bar{V}_A = \{v_A^i | i = 1, 2, 3, 4\}$ and that of $\bar{G}_B$ is $\bar{V}_B = \{v_B^i | i = 1, 2, 3, 4\}$. Now v_B^4 is a vertex of $\bar{G}_B$. Skeletal graphs and feature graphs. The dashed graphs are skeletal graphs and the solid ones are their corresponding feature graphs. In (A) or (B), the right corresponds to the object A and the left to B.

As showed in Fig. 3A, the original mappings of feature correspondence is defined as: $\varphi_A(v_A^i) = v_B^i \;\; i = 1, 2, 3, 4$, and $\varphi_B(v_B^i) = v_A^i, \;\; i = 1, 2, 3$, respectively. They are not mappings between V_A and V_B.[1] So they are extended to those between $\bar{V}_A$ and $\bar{V}_B$:$\bar{\varphi}_A(v_A^i) = v_B^i$, $i = 1, 2, 3, 4$, and $\bar{\varphi}_B(v_B^i) = v_A^i, \;\; i = 1, 2, 3, 4$.

Definition 1. Each element in $\bar{V}_A$ or $\bar{V}_B$ is called a feature vertex about $\bar{\varphi}_A$ or $\bar{\varphi}_B$, respectively, or a feature vertex for short.

Definition 2. A geometric graph $\bar{G}_A$ is obtained when the vertex set is extended from V_A to $\bar{V}_A$ for G_A, and named as the extended geometric graph of G_A about $\bar{V}_A$. And $\bar{G}_B$ is defined so for B.

In Fig. 3B, $\bar{G}_B$ whose vertex set is $\bar{V}_B$, is an extended geometric graph of G_B whose vertex set is V_B. It is obvious that V_B is a true subset of $\bar{V}_B$.

After definition of $\bar{G}_A$ and $\bar{G}_B$, the map of vertex correspondence is a bijection from $\bar{V}_A$ to $\bar{V}_B$, but not one-to-one map yet.

[1] Note that v_B^4 is not a vertex of B.

3.2. Feature graph

Definition 3. Given a geometric graph G, whose vertex set is V, another geometric graph T satisfies the following conditions:

(1) V is the vertex set of T;

(2) $\forall u, v \in V$, u, v are adjacent in $G \Longleftrightarrow u, v$ are adjacent in T;

(3) Each edge of T is a line segment.

then we call T as linear (geometric) graph of G.

It is evident that G and T are isomorphic.

Definition 4. Given two geometric graphs of G_A and G_B, $\bar{G}_A$ and $\bar{G}_B$ are their extended geometric graphs about $\bar{\varphi}_A$ and $\bar{\varphi}_B$, respectively. Then the linear geometric graph of $\bar{G}_A$ is the feature graph of G_A with feature vertex set $\bar{V}_A$, denoted as T_A. And T_B is defined similarly.

For instance as in Fig. 3B, T_A is a linear graph of $\bar{G}_A$ and G_A, and it is also a feature graph of G_A; T_B is a linear graph of $\bar{G}_B$ instead of G_B, yet a feature graph of G_B.

The trajectories of feature vertices during morphing are controlled by feature graphs.

4. Construction of isomorphic feature graphs

The interpolation of G_A and G_B is driven by T_A and T_B. In this section, we address the issue of graph isomorphism of T_A and T_B, and the solution is based on the one-to-one correspondences of vertices and edges.

There are only two sorts of geometric primitives in T_A or T_B, point and line segment, if a closed curve (edge) is regarded as the one whose two end-points coincide. So the interpolation of the intermediate primitives between T_A and T_B consists of three types. That is

(1) Vertex $\leftrightarrow$ Vertex

(2) Edge $\leftrightarrow$ Edge

(3) Vertex $\leftrightarrow$ Edge

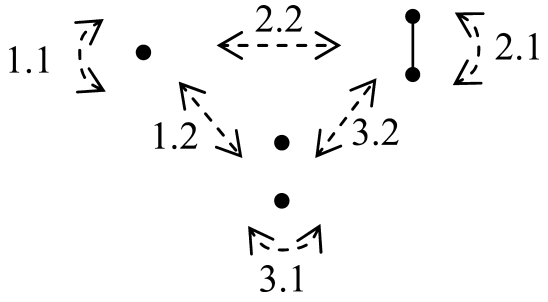

Fig. 4. Transformation type. 1.1 is 'one vertex to one vertex'; 1.2 is 'one vertex to multi vertices'; 2.1 is 'one edge to one edge'; 2.2 is 'one vertex to one edge'; 3.1 is 'two vertices to two vertices'; 3.2 is 'two vertices to one edge.' 1.2 and 3.2 may cause the topological change.

Each transformation type of geometric primitives consists of two operations: Type 1 includes "one vertex to one vertex" and "one vertex to multi vertices"; Type 2 comprises "one edge to one edge" and "one vertex to one edge"; Type 3 is composed of "two vertices to two vertices" and "two vertices to one edge." It is evident that the latter of Type 1 or Type 3 may cause the change of topology (see Fig. 4). In this section, we will fulfill the process of topology change.

The main goal of this section is to construct isomorphic feature graphs for T_A and T_B, which can be applied to $\bar{G}_A$ and $\bar{G}_B$ accordingly. The process includes vertex split and edge correspondence.

4.1. Vertex split

The vertex split in T_A is performed first. T_A and $\bar{G}_A$ have the same feature vertex set $\bar{V}_A$, and we call each one in $\bar{V}_A$ as a mother vertex.

$\forall v_A^0 \in \bar{V}_A$, denote $v_B^0 = \varphi_A(v_A^0) \in \bar{V}_B$ and $V' = \{v | v \in \bar{V}_B \text{ and } v \neq v_B^0 \text{ and } \varphi_B(v) = v_A^0\} \subseteq \bar{V}_B$. Let $n = |V'|$, and denote every element of V' as $v_B^i, i = 1, \ldots, n$ in turn. The mother vertex v_A^0 puts forth n vertices $v_A^1, \ldots, v_A^n$ (v_A^i, $i = 0, 1, \ldots, n$, coincide at one geometric location). Adding the n vertices into $\bar{V}_A$, we obtain a new (extended) vertex set $\bar{V}_A'$ while extending the function $\bar{\varphi}_A$ from domain $\bar{V}_A$ to $\bar{V}_A'$ (see Fig. 5):

$$\bar{\varphi}_A'(v) = \begin{cases} \bar{\varphi}_A(v), & v \in \bar{V}_A \\ v_B^i, & v = v_A^i (i = 1, \ldots, n). \end{cases}$$

Definition 5. $\forall v \in \{v_A^i | i = 1, \ldots, n\}$, v is a child vertex of v_A^0 and v_A^0 is the mother vertex of v.

To express such a vertex relation, we introduce a dashed-oriented edge by which a mother vertex is connected with one child (see Fig. 5). A dashed-oriented edge usually has nothing to do but to represent the relation of mother and children. If, however, the feature graph is not connected after vertex split and edge correspondence, some dashed edges will participate in the morphing step as connectors among its subparts (see Section 7).

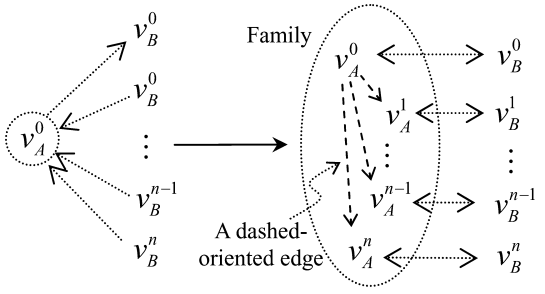

Fig. 5. Vertex split. v_A^0 is duplicated n times to establish one-to-one correspondences between v_A^i and v_B^i ($i = 0, 1, \ldots, n$).

Definition 6. Suppose $\bar{V}'_A$ is the vertex set after vertex split of $\bar{V}_A$. If $x, y \in \bar{V}'_A$ satisfy any one of the following conditions:

(1) x and y are in relation of a mother and its child;

(2) x and y have the same mother vertex;

(3) $x = y$,

then we say x is in relation F with y.

It is evident that $\bar{V}'_A$ is partitioned by the equivalence F into $|\bar{V}_A|$ subsets, each one of which we call a family. We denote that x and y are in the same family as xFy, and $x\bar{F}y$ means that xFy fails. The family of x is written as $F(x)$. It is obvious that $F(x) = F(y)$ if xFy holds. From a viewpoint of geometry, all vertices in a family have the same coordinates.

Definition 7. Suppose v_1 and v_2 are two adjacent vertices of $\bar{V}'_A$ or $\bar{V}'_B$. Then $F(v_1)$ and $F(v_2)$ are adjacent if $v_1\bar{F}v_2$ holds. We write xRy for any $x \in F(v_1)$ and $y \in F(v_2)$.

Repeat the above process for all mother vertices in $\bar{V}_A$ and obtain the extended $\bar{V}'_A$ of $\bar{V}_A$, leading to the extended T'_A of T_A whose vertex set is $\bar{V}'_A$. Meanwhile, the correspondence mapping of $\bar{\varphi}_A : \bar{V}_A \to \bar{V}_B$ is extended to $\bar{\varphi}'_A : \bar{V}'_A \to \bar{V}_B$.

The operation of vertex split can be applied to B in the same manner. Then it can be shown that $\bar{\varphi}'_A$ and $\bar{\varphi}'_B$ define the one-to-one mappings between $\bar{V}'_A$ and $\bar{V}'_B$, respectively, and $(\bar{\varphi}'_A)^{-1} = \bar{\varphi}'_B$.

On the other hand, the original edges connected with a mother vertex of v_A^0 may be reassigned to her children of $v_A^1, \ldots, v_A^n$ for better visual effects. It is done as following:

FOR (all the mother vertex $\bar{v}_A^0$ adjacent to v_A^0)
  IF $((\bar{\varphi}'_A(v_A^0)\bar{F}\bar{\varphi}'_A(\bar{v}_A^0) \wedge \bar{\varphi}'_A(v_A^0)\bar{R}\bar{\varphi}'_A(\bar{v}_A^0))\wedge$
    $(\exists v_A^i \in F(v_A^0) \Rightarrow \bar{\varphi}'_A(v_A^i)F\bar{\varphi}'_A(\bar{v}_A^0) \vee \bar{\varphi}'_A(v_A^i)R\bar{\varphi}'_A(\bar{v}_A^0)))$ THEN
    Reassign $v_A^0\bar{v}_A^0$ to v_A^i.
END FOR

4.2. Edge correspondence

The one-to-one mapping has been constructed between vertex sets of T'_A and T'_B after vertex split, but that between edge sets E'_A of T'_A and E'_B of T'_B still requires edge correspondence to be defined.

We will only discuss the correspondence of edge set E'_A in T'_A since the correspondence of the edge set E'_B in T'_B is similar to that in T'_A.

$\forall \ell \in E'_B$,with two end points v_B^1 and v_B^2, denote $v_A^1 = \bar{\varphi}'_B(v_B^1)$, $v_A^2 = \bar{\varphi}'_B(v_B^2)$. Then

(1) $v_A^1 F v_A^2$. Insert a new edge between v_A^1 and v_A^2. Of course, this operation will be ignored if v_A^1 and v_A^2 is adjacent already.

(2) $v_A^1 \bar{F} v_A^2$. If $F(v_A^1)RF(v_A^2)$ holds, a new edge is inserted between v_A^1 and v_A^2; otherwise particular measures must be taken as following (see Fig. 6):

Insert a new vertex $\bar{v}_B^1$ onto the edge $v_B^1v_B^2$ and move it onto the edge $v_B^1v_B^2$ in $\bar{G}_B$, whose location could be the midpoint of the edge $v_B^1v_B^2$ in $\bar{G}_B$ or specified by user. Then $\bar{v}_B^1$, as a mother vertex, give birth to a child vertex $\bar{v}_B^2$ which is connected with

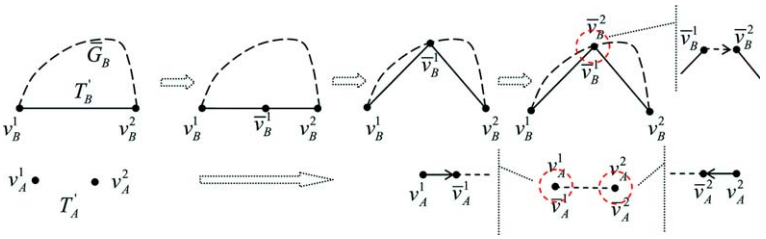

Fig. 6. Default edge correspondence when the edge $v_B^1 v_B^2$ has no counterpart in T_A'.

$\bar{v}_B^1$ by a dashed-oriented edge, and the edge $\bar{v}_B^1 v_B^2$ is reassigned to $\bar{v}_B^2$. Accordingly, v_A^1 (v_A^2) puts forth a child vertex $\bar{v}_A^1$ ($\bar{v}_A^2$). A new edge is inserted between v_A^1 and $\bar{v}_A^1$, which is a solid-oriented edge indicating that v_A^1 is a mother. So does between v_A^2 and $\bar{v}_A^2$. Finally for one-to-one mapping of edge $\bar{v}_A^1$ and $\bar{v}_A^2$ are connected by a dashed edge indicating that they are not in a family. It is easy to see that the edge correspondence process contains vertex split: $\bar{V}_A'$ is extended to $\bar{V}_A' \cup \{\bar{v}_A^1, \bar{v}_A^2\}$ and $\bar{V}_B'$ is extended to $\bar{V}_B' \cup \{\bar{v}_B^1, \bar{v}_B^2\}$. So we renew the definition of $\bar{\varphi}_A'$ and $\bar{\varphi}_B'$ on $\{\bar{v}_A^1, \bar{v}_A^2\}$ and $\{\bar{v}_B^1, \bar{v}_B^2\}$, respectively:

$$\bar{\varphi}_A'(\bar{v}_A^1) = \bar{v}_B^1, \quad \bar{\varphi}_A'(\bar{v}_A^2) = \bar{v}_B^2, \quad \bar{\varphi}_B'(\bar{v}_B^1) = \bar{v}_A^1, \quad \bar{\varphi}_B'(\bar{v}_B^2) = \bar{v}_A^2.$$

If unsatisfied with the automatic edge correspondence, the user is allowed to interfere and govern the editing process of edge correspondence by manually inserting vertices into the edited edge and specifying their correspondence counterparts. Because this process depends on the context in which editing is performed, a fully common algorithm seems impossible for the general case, but it should conform to the rules of vertex and edge correspondence stated above. For instance, the user lays his (her) account for the pattern (a) in Fig. 1. Then the user adds a new vertex $\bar{v}_A^1$ into the edge $v_A^2 v_A^3$ and defines $\bar{\varphi}_A'(\bar{v}_A^1) = v_B^1$. Then the following steps are similar to those mentioned above, as shown in Fig. 7.

It is easy to prove that a unified bijective mapping of vertex and edge can be made between T_A' and T_B' after correspondence operation of T_A and T_B. That is, there is no need for additional mappings of edge since they can be specified by the vertex mappings of $\bar{\varphi}_A'$ and $\bar{\varphi}_B'$ based on the above assumption of G_A and G_B.

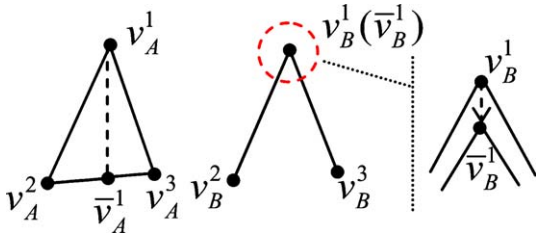

Fig. 7. User-controlled edge correspondence.

In the following sections, our discussion will focus on the extended T'_A and T'_B instead of T_A and T_B. So we will still denote, for short, $\bar{\varphi}'_A$, $\bar{\varphi}'_B$, $\bar{V}'_A$, $\bar{V}'_B$, T'_A, T'_B as $\bar{\varphi}_A$, $\bar{\varphi}_B$, $\bar{V}_A$, $\bar{V}_B$, T_A, T_B, respectively, without confusion.

5. Generation of intermediate feature graph

Feature graph reveals the topological structure of a shape in a simpler manner and dominates in our approach because the trajectories of feature vertices are determined by their corresponding feature graphs. For further process using the edge-angle blending technique, we must know the length and angle of each edge.

Suppose ℓ_A is an edge of T_A. If ℓ_A is a degenerate form, say $|\ell_A| = 0$, there is no natural angle definition for ℓ_A. In fact, such a case could come into being due to the above vertex and edge correspondence. We call such an edge as a zero edge which will disturb the interpolation using intrinsic shape parameters.

5.1. Computation of the angle for zero edges

5.1.1. Angle of zero edges

For simplicity and clarification, when Λ_A is a denotation of one certain variable about A, we denote the counterpart of Λ_A as Λ_B related to object B in case they are involved in the same context.

One can observe a simple fact: there exists no more than one zero edge in any pair of corresponding edges ℓ_A in T_A and ℓ_B in T_B because the construction process of $\bar{\varphi}_A$ and $\bar{\varphi}_B$ impliedly specify the edge correspondence and it is not allowed that two vertices coincide at one position in G_A and G_B. Without loss of generality, let $|\ell_A| = 0$ and $|\ell_B| \neq 0$. We will confirm the angle of ℓ_A according to the intrinsic angles among the adjacent and corresponding edges of ℓ_A and ℓ_B, respectively.

Let v_A be an end point of ℓ_A. Denote $NE(\ell_A, v_A)$ as the set of edges connected with ℓ_A at v_A, whose length is nonzero or whose length is zero but direction has been specified, and $NE(\ell_B, v_B)$ is defined similarly for B. Then we define

$$E(\ell_B, v_B) = \{b_B | b_B \in NE(\ell_B, v_B) \ \wedge \ b_A \in NE(\ell_A, v_A)\},$$

- $E(\ell_B, v_B) \neq \phi$ We search two edges in $E(\ell_B, v_B)$, say, ℓ^1_B and ℓ^2_B,[2] which are the first edges barged up against in the clockwise and anticlockwise directions from ℓ_B, respectively. Suppose the intrinsic angle is θ^1_B between ℓ_B and ℓ^1_B, and θ^2_B between ℓ_B and ℓ^2_B. The corresponding angles of ℓ_A, θ^1_A and θ^2_A, should satisfy the following conditions (see Fig. 8):

(1) $\ell_A, \ell^1_A, \ell^2_A$ and $\ell_B, \ell^1_B, \ell^2_B$ around v_A and v_B are in the same order, clockwise or anticlockwise, respectively.

$$\frac{\theta^1_A}{\theta^2_A} = \frac{\theta^1_B}{\theta^2_B}.$$

[2] ℓ^1_B and ℓ^2_B may be the same edge.

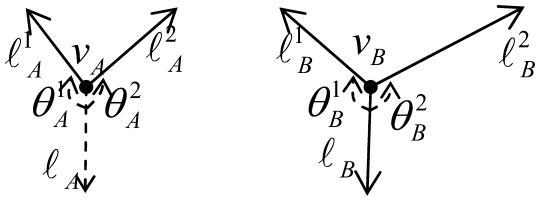

Fig. 8. Direction angle of a zero edge ℓ_A.

- $E(\ell_B, v_B) = \phi$ The direction of ℓ_A is still uncertain and any value can be set.

Once θ_A^1 and θ_A^2 are fixed, the direction of ℓ_A can be specified. We write the principle calculating the angle of zero-edge ℓ as $CA(\ell, v)$, where v is an end point of ℓ.

5.1.2. Initialization of the direction of zero edges

There may be more than one zero-edge in T_A or T_B. We take a pair of corresponding non-zero edges as the reference ones, on which the specification of angles of the other zero-edges is based. If there is no such a pair of edges, though this case seldom appears in practice, any pair could be, such as the pair in the *center* of their graphs or specified by user, denoted as ℓ_A and ℓ_B yet. On the basis of above-mentioned conclusion, only one of them is a zero-edge, say ℓ_A, and its angle can be designed to that of ℓ_B since there is no "reasonable" pair of nonzero-edge and any direction could be allowed. The algorithm is as following:

Calculate the direction of all nonzero-edge in T_A and T_B
Select a pair of edges (ℓ_A, ℓ_B) as reference edges and calculate their angles
Push(ℓ_A) Mark(ℓ_A)
WHILE (Stack is not empty)
  Pop an edge in stack to ℓ_A
  Push all the unmarked and adjacent edges of $\ell_A$to the stack and mark them
  ℓ_B =Corresponding edge of $\ell_A$in T_B
  IF (Both angles of ℓ_A and ℓ_B have been calculated) CONTINUE
  IF (The angle of ℓ_A is not calculated)
  $\ell = \ell_A$
  ELSE $\ell = \ell_B$
  Ang1 $= CA(\ell, \ell \to v_1)$
  Ang2 $= CA(\ell, \ell \to v_2)$
  $(\omega_1, \omega_2) =$ Weight(ℓ)[3]
  Set the angle of ℓ to be $\omega_1 * Ang1 + \omega_2 * Ang2$.
END WHILE.

[3] IF $E(\ell, \ell \to v_1) \neq \Phi \wedge E(\ell, \ell \to v_2) \neq \Phi$ THEN $(\omega_1, \omega_2) = (0.5, 0.5)$ ELSE IF $E(\ell, \ell \to v_1) \neq \Phi$ THEN $(\omega_1, \omega_2) = (1, 0)$ ELSE $(\omega_1, \omega_2) = (0, 1)$.

5.2. Generation of intermediate feature graph

After the initialization of zero edges, we can set about the computation of intermediate feature graphs.

5.2.1. Vertex indexing

Firstly, a pair of corresponding vertices in T_A and T_B, u_A and u_B, are chosen as the base vertices. The other vertices are then indexed starting from the base vertices in the breath-first manner. The vertices having the same index n belong to the same level n and are denoted as $v^0_{n,A}, v^1_{n,A}, \ldots, v^{l_n}_{n,A}$ for A or $v^0_{n,B}, v^1_{n,B}, \ldots, v^{l_n}_{n,B}$ for B in turn. In particular, let $v^0_{0,A} = u_A$ and $v^0_{0,B} = u_B$. Meanwhile, we chose a pair of corresponding edges connected to $v^0_{0,A}$ or $v^0_{0,B}$ as reference edges (Refer to the above section for the selection principle), whose another end points are denoted as $v^0_{1,A}$ or $v^0_{1,B}$. $v^i_{n,A}$ and $v^j_{n+1,A}$ are adjacent if we write $\ell^{ij}_{n,A} = v^i_{n,A}v^j_{n+1,A}$ and it is done for B in the same manner.

5.2.2. Generation of intermediate feature graphs

The generation of intermediate feature graphs is practically the process of tracing the travel trajectories of feature vertices, from which the morphing will benefit between the source skeleton and the target one. It is illustrated with $T_A(t)$ from A to B.

We use the method in [12] to produce the intermediate feature graph $T_A(t)$:

(1) Locate $v^0_{0,A}(t)$;
(2) Using the edge-angle blending technique, calculate the intermediate edge of the reference edge, $\ell^{00}_{0,A}(t)$, whose intrinsic definition is calculated with respect to the x axis.
(3) Interpolate the edges in level 0 to compute $\ell^{0i}_{0,A}(t)$, $(i \neq 0)$, whose intrinsic definition are designed to those with respect to the reference edge $\ell^{00}_{0,A}$.
(4) Interpolate the edges in $\ell^{ij}_{l,A}$ level $l(\geqslant 1)$, whose intrinsic definition are determined with respect to the edges in level $l-1$ adjacent to $\ell^{ij}_{l,A}$.

Some details of the algorithm for the generation of $T_A(t)$ in this paper are different from that in [10] because we must guarantee it to be symmetric with respect to t (see Fig. 14). See Section 6.3 and (Appendix B) for more details.

6. Skeleton-driven shape morphing

In this section, we describe our algorithm for the smooth interpolation between A and B via their feature graphs.

The aim of morphing is to find a continuous interpolation function $W(t, Y)$, $0 \leqslant t \leqslant 1$, where Y is a planar point set. It satisfies on domain A and B:

$$W(0, A) = A, \quad W(1, A) = B,$$

$$W(0, B) = B, \quad W(1, B) = A.$$

Let $W(t, V_A)$ be the restriction of function $W(t, A)$ on V_A and $W(t, V_B)$ be that of $W(t, B)$ on V_B. It is obvious that we have defined $W(t, V_A)$ and $W(t, V_B)$ in Section 5.

In this section we will give their definition on A and B due to $W(t, V_A)$ and $W(t, V_B)$, respectively. Similarly as above, we illustrate it by the construction of $W(t, A)$ from A to B.

6.1. Morphing of original skeleton graph

By applying the isomorphic operation of vertex and edge in T_A to $\bar{G}_A$, we will obtain the extended geometric graph S_A of $\bar{G}_A$, which processes the same geometry, say vertex set, and same topology, say connectivity, as T_A. The difference between S_A and T_A is the shape of their edges, where the edge shape in S_A relies on G_A, but not line segment simply and solely as in T_A. Similarly, we get S_B.

It is no doubt that $T_A(t)$ gives rise to the vector $X_A^i(t)$ of the two end-points of each edge $C_A^i(t)$ in $S_A(t)$, where i is the corresponding edge index. So $S_A(t)$ can be guided to shape directly driven by $T_A(t)$, each edge of which is calculated in the same way as mentioned in the Section 2.2.

Thus, we have defined $W(t, A)$ on domain G_A. That is, $W(t, G_A) = S_A(t)$.

6.2. Skeleton-driven 2D distance field metamorphosis

Given two domains of A and B, the interpolation from A to B can be transformed into that from $MAT(A)$ to $MAT(B)$ because a medial axis transformation can exactly reconstruct its corresponding shape. Thus we define

$$MAT(t, A) = \{(p, r) \mid p \in S_A(t), r = (1-t)r_A + tr_B \text{ where } (p_A, r_A) \in MAT(A)$$

$$\text{in which } W(t, p_A) = p \text{ and } (p_B, r_B) \in MAT(B) \text{in which } W(1-t, p_B) = p\}.$$

We take $MAT(t, A)$ as the intermediate shape from A to B. That is

$$W(t, A) = MAT(t, A).$$

6.3. Symmetric property

It is a desirable property that the transformation would be symmetric with respect to time step, namely, W should be symmetric with respect to t [2]:

$$W(t, A) = W(1-t, B).$$

Before proving this property, we must have a clear idea of what purpose we want the equality of two geometric graphs to be. In this paper, it means that these two graphs coincide with each other entirely. For example, $T_A(t) = T_B(1-t)$ holds if the geometric coordinates of their corresponding vertices are equal because $T_A(t)$ and $T_B(1-t)$ are isomorphic in the sense of edge-angle blending and are linear geometric graphs. That is, if $\forall v_{l,A}^{ij}(t) \in \bar{V}_A(t) \Rightarrow v_{l,A}^{ij}(t) = v_{l,B}^{ij}(1-t)$, $T_A(t) = T_B(1-t)$ holds.

Property. $W(t, A) = W(1-t, B)$.

Proof. We have shown that $T_A(t) = T_B(1-t)$ in (Appendix B) The constructive proof is self-explanatory. On the other hand, since $T_A(t)$ gives rise to the vector $X_A^i(t)$ of the two end-points of each edge $C_A^i(t)$ in $S_A(t)$ and $X_A^i(t) = X_B^i(1-t)$ holds, $C_A^i(t) = C_B^i(1-t)$ holds. Thus we have $S_A(t) = S_B(1-t)$, leading to $MAT(t,A) = MAT(1-t,B)$. That is, $W(t,A) = W(1-t,B)$.

7. Example

In this section we give a few examples that illustrate the behavior of our algorithm.

Fig. 9 shows the extended skeletal graphs of two shapes 'd' and 'e,' where the feature vertices are represented by dots, respectively. And we mark the feature vertices with relevant numbers for clarification. The vertices encircled by a red circle are regarded as a whole with a same number. Although the graphs of 'd' and 'e' have been extended, we can learn how our approach works in detail to some extent. In this example, vertices of 1, 3, and 4 could be the key ones for correspondence. They divide jointly the whole shapes of 'd' and 'e' into three subparts, respectively. The different manner of the correspondence of subparts (or key vertices) can generate different morphing effects. Fig. 10 shows two available patterns of the shape interpolation between 'd' and 'e' due to different correspondence. One pattern is realized by breaking down the circle substructures of 'd' and 'e' into the linear ones, including topological change, as shown in Fig. 10B. It is realized by specifying correspondence of subparts such that d123 to e3654, d34–e43, e4563–d3271 and the detailed correspondence of each subpart can be reflected on in Fig. 10A by readers. If, however, we specify that d123–e1723, d34–e34, d3654–e3654 (see Fig. 10C), another pattern transforms 'd' to 'e' mainly via a rotational transformation, in which orientation and position change a lot instead of topological structure as in Fig. 10D.

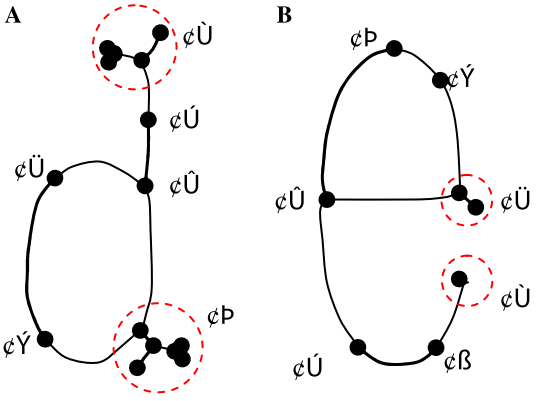

Fig. 9. (A) 'd' shape (B) 'e' shape Extended skeletal graphs.

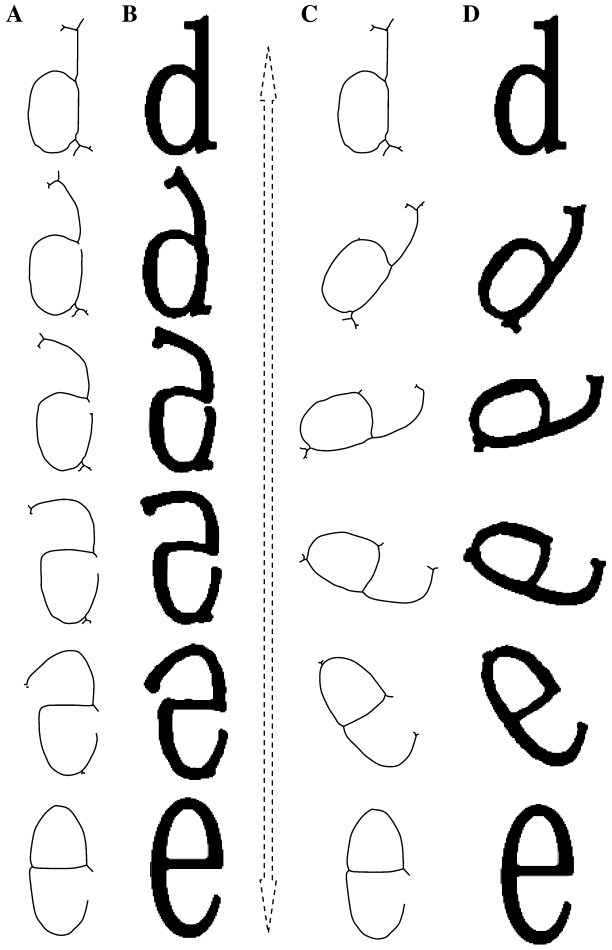

Fig. 10. Morphing between the letter 'd' and 'e'. (A) is the morphing of the skeletons in one manner and (B) is the counterpart morphing of medial axis transformation driven by (A). (C) and (D) are the corresponding pair of morphing but in another manner.

Fig. 11 is a metamorphosis of 'X' and 'O.' A normal consideration gives rise to scheme (a) and attention should be paid to its natural and smooth process in which the 'X' shape is not broken into pieces as in [21]. Another strange consideration is shown in Fig. 11B although someone may feel it unpleasant. Because the shape of 'X' is broken up into two pieces, a dashed edge located at the separate position of the centre of 'X' should exert an influence of connection on the feature graph between the two separate parts. Thus all the primitives of vertices and edges in the feature graph can be visited from the based vertex through the gangway of such a dashed edge.

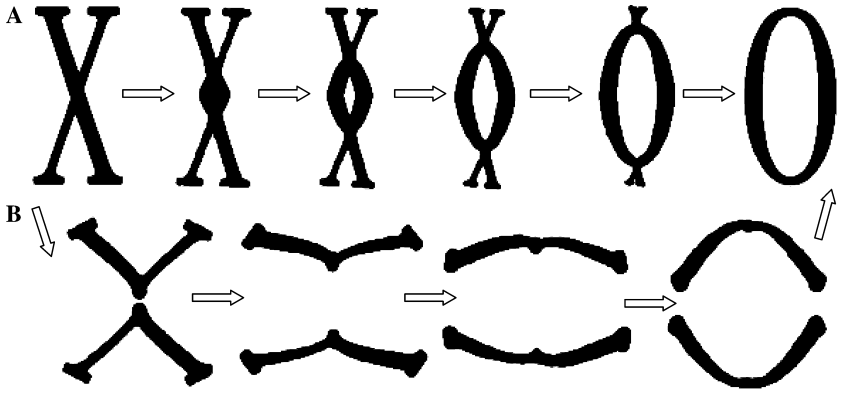

Fig. 11. Morphing between the letters 'X' and 'O.'

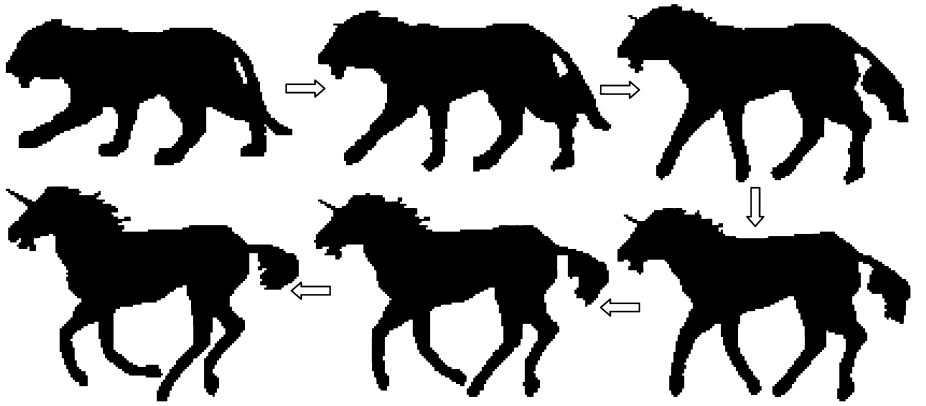

Fig. 12. Morphing between a tiger and a unicorn.

In Fig. 12, a normal interpolation process is shown, in which two shapes of a unicorn and a tiger are the source and the target objects, respectively.

8. Conclusion

In this paper, we present an algorithm for shape interpolation by applying skeletal representation to control topological structure. It combines the context of transformation with the user' subjective aesthetic criteria to the most extent, to achieve the compelling effect of controllable topologies of intermediate shapes. On the other hand, more interesting animations are possible if transition rates differs from part to part in-between images [22]. So we can set different transition control for different edges since each edge of feature graph is relatively independent.

The main limitation of our method is the difficulty of totally automatic correspondence of feature vertices in the general case. It usually requires more or less manual involvement for reasonable and different correspondences when one shape is obviously dissimilar to the other. Another important issue is the control of the texture detail. There was no mention of the problems in this paper. In fact, they partly lead to the symmetric property of our algorithm, which is proposed for further work. We are getting ready for that.

Acknowledgments

The authors acknowledge the referees for helpful comments. This work is supported by the NSF of China under Grant No. 60073023 the fund for innovative research group (60021201) and the 973 Program of China under Grant No. 2002CB312101.

Appendix A. Morphing between two curves

For simplicity and clarification, let $\{*, \bar{*}\} = \{A, B\}$. Given two curves $C_A = \{v_A^0, v_A^1, \ldots, v_A^n\}$ and $C_B = \{v_B^0, v_B^1, \ldots, v_B^n\}$, we write their intermediate curves as $C_*(t) = \{v_*^0(t), v_*^1(t), \ldots, v_*^n(t)\}$, where * shows that C_* is regarded as the source object and $C_{\bar{*}}$ as the target one, say, $C_*(0) = C_*$, $C_*(1) = C_{\bar{*}}$. And in this way, if Λ_* is a denotation of one variable about object $*$, we denote its the counterpart of Λ_* as $\Lambda_{\bar{*}}$ but related to object $\bar{*}$ in case they are involved in the same context, and its corresponding denotation of intermediate shape as $\Lambda_*(t)$from $*$ to $\bar{*}$ at time t.

The travel trajectories of $C_*(t)$ can be traced by using intrinsic shape parameters shown in Fig. 13. Let $X_* = v_*^n - v_*^0$. We desire that $X_*(t) = v_*^n(t) - v_*^0(t)$ for $C_*(t)$, where $X_*(t)$ can be obtained in advance (see Section 5). Since the desire can not be met just by interpolating intrinsic shape parameters linearly, we can adjust the edge lengths only to do so [17]

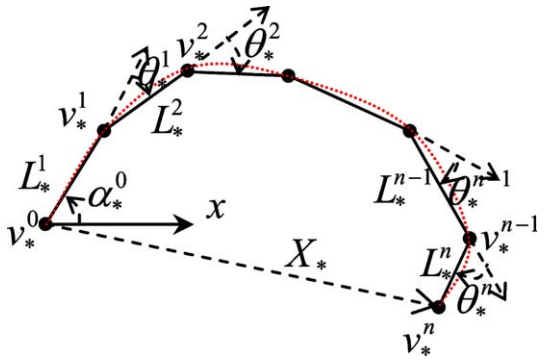

Fig. 13. Intrinsic variables.

$$(1-t)L_*^i + tL_{\bar{*}}^i + D_*^i(t) = L_*^i(t) + D_*^i(t), \quad i = 0, 1, \ldots, n.$$

Here superscripts denote index, not exponentiation.

Our goal is to find $D_*^0(t), D_*^1(t), \ldots, D_*^n(t)$ so that the objective function

$$f(D_*^0(t), D_*^1(t), \ldots, D_*^n(t)) = \sum_{i=0}^{n} \left(\frac{D_*^i(t)}{L_*^i, \bar{*}} \right)^2$$

is minimized subject to the two equality constraints enforced by $X_*(t) = (x_*(t), y_*(t)) = v_*^n(t) - v_*^0(t)$:

$$\begin{cases} \varphi\left(D_*^0(t), D_*^1(t), \ldots, D_*^n(t)\right) = \sum_{i=0}^{n} \left[L_*^i(t) + D_*^i(t)\right] \cos\alpha_*^i - x_*(t) = 0 \\ \varphi\left(D_*^0(t), D_*^1(t), \ldots, D_*^n(t)\right) = \sum_{i=0}^{n} \left[L_*^i(t) + D_*^i(t)\right] \sin\alpha_*^i - y_*(t) = 0 \end{cases}.$$

Here t is a constant parameter in $[0, 1]$ and the meaning of each variable is as following:

$$\begin{cases} L_{*,\bar{*}}^i = \left| v_*^i - v_{\bar{*}}^{i-1} \right| + SN, & i = 1, \ldots, n \\ L_*^i(t) = (1-t)L_*^i + tL_{\bar{*}}^i, & i = 1, \ldots, n \\ \alpha_*^0(t) = (1-t)\alpha_*^0 + t\alpha_{\bar{*}}^0, & \\ \theta_*^i(t) = (1-t)\theta_*^i + t\theta_{\bar{*}}^i, & i = 1, \ldots, n \\ \alpha_*^i(t) = \alpha_*^{i-1}(t) + \theta_*^i(t), & i = 1, \ldots, n, \end{cases}$$

where SN is a small number to avoid division by zero. The optimal solution to the objective function is similar to [17]. According to the objective function and its constraints we have

$$D_A^i(t) = D_B^i(1-t) \quad i = 0, 1, \ldots, n.$$

So if $v_A^0(t) = v_B^0(1-t)$ and $X_A(t) = X_B(1-t)$, $C_A(t) = C_B(1-t)$ holds.

Appendix B. Generation of intermediate feature graph

Here we present our method of how to produce the intermediate the feature graphs between T_A and T_B. And It is also a constructive proof of $T_A(t) = T_B(1-t)$.

(1) The intermediate position of the base point

The travel trajectories of the base vertices in T_A and T_B can be obtained by a function $g(t)$, $0 \leqslant t \leqslant 1$, which satisfies the boundary conditions:

$$g(0) = v_{0,A}^0, \quad g(1) = v_{0,B}^0.$$

It is common that $g(t)$ is calculated using the linear interpolation of $v_{0,A}^0$ and $v_{0,B}^0$. That is

$$g(t) = (1-t)v_{0,A}^0 + tv_{0,B}^0.$$

We cannot help believing that it is natural and optimal if the intermediate coordinates, regarding T_A and T_B as source object, respectively, can be represented as:

$$v^0_{0,A}(t) = g(t), \quad v^0_{0,B}(t) = g(1-t).$$

It is evident that $v^0_{0,A}(t) = v^0_{0,B}(1-t)$ holds.

(2) Recursive generation of $T_*(t)$

Let $\gamma(\ell^{ij}_{n,*}, \ell^{jk}_{n+1,*})$ be the intrinsic angle of $\ell^{ij}_{n,*}$ and $\ell^{jk}_{n+1,*}$. Its intermediate value at time t is $\gamma_*(\ell^{ij}_{n,*}, \ell^{jk}_{n+1,*}, t) = (1-t)\gamma(\ell^{ij}_{n,*}, \ell^{jk}_{n+1,*}) + t\gamma(\ell^{ij}_{n,\bar{*}}, \ell^{jk}_{n+1,\bar{*}})$, * being the source object.

Denote the length of $\ell^{ij}_{n,*}$ as $L^{ij}_{n,*}$ and the angle with respect to x axis as $\theta^{ij}_{n,*}$ and let $L^{ij}_{n,*}(t) = (1-t)L^{ij}_{n,*} + tL^{ij}_{n,\bar{*}}$ $\theta^{ij}_{n,*}(t) = (1-t)\theta^{ij}_{n,*} + t\theta^{ij}_{n,\bar{*}}$.

We will prove it by induction on the level number n of vertices in $\bar{V}_*(t)$.

- $n = 0$. There exits only the base point in the level 0. As mentioned above, we know $v^0_{0,A}(t) = v^0_{0,B}(1-t)$.
- $n = 1$. First, we take $v^{00}_{1,*}$ into account. Since its immediate coordinate is fixed by the reference edges of $\ell^{00}_{0,*}$, whose intrinsic angle, $\theta^{00}_{0,*}$, is calculated with respect to x axis, we have

$$v^0_{1,A}(t) = L^{00}_{1,A}(t)\mathrm{e}^{\mathrm{i}\theta^{00}_{0,A}(t)} + v^0_{0,A}(t), \quad v^0_{1,B}(t) = L^{00}_{1,B}(t)\mathrm{e}^{\mathrm{i}\theta^{00}_{0,B}(t)} + v^0_{0,B}(t),$$

where $\mathrm{i}^2 = -1$.

Taking notice of $L^{00}_{1,A}(t) = L^{00}_{1,B}(1-t)$, $\theta^{00}_{0,A}(t) = \theta^{00}_{0,B}(1-t)$, $v^0_{0,A}(t) = v^0_{0,B}(1-t)$, we know

$$v^0_{1,A}(t) = v^0_{1,B}(1-t).$$

Then we continue calculating $v^i_{1,*}(t)(i \neq 0)$. The intrinsic definition of $\ell^{0i}_{0,*}$ is defined with respect to $\ell^{00}_{0,*}$, so

$$\theta^{0i}_{0,*}(t) = \gamma_*(\ell^{00}_{0,*}, \ell^{0i}_{0,*}, t) + \theta^{00}_{0,*}(t).$$

According to $\gamma_A(\ell^{00}_{0,A}, \ell^{0i}_{0,A}, t) = \gamma_B(\ell^{00}_{0,B}, \ell^{0i}_{0,B}, 1-t)$ and $\theta^{00}_{0,A}(t) = \theta^{00}_{0,B}(1-t)$, $\theta^{0i}_{0,A}(t) = \theta^{0i}_{0,B}(1-t)$ holds. Thus $v^i_{1,A}(t) = v^i_{1,B}(1-t)$ holds.

Suppose $v^i_{n,A}(t) = v^i_{n,B}(1-t)$ if $v^i_{n,A}(t) = v^i_{n,B}(1-t)$ holds, where $n \leqslant l$.

- $n = l+1$.We need to prove $v^i_{l+1,A}(t) = v^i_{l+1,B}(1-t)$.

Denote the vertices in level l adjacent to $v^i_{l+1,*}$ as $v^{j_0}_{l,*}, v^{j_1}_{l,*}, \ldots, v^{j_p}_{l,*}$ in turn. The position of $v^i_{l+1,*}(t)$ is computed as follows.

$$v^i_{l+1,*}(t) = \sum_{a=0}^{p} \alpha^{j_a i}_l \bar{v}^{j_a i}_{l+1,*}(t) / \sum_{a=0}^{p} \alpha^{j_a i}_l$$

Here, $\alpha^{j_a i}_l = 1/(SN + |L^{j_a i}_{l,*} - L^{j_a i}_{l,\bar{*}}|)$ and $\bar{v}^{j_a i}_{l+1,*}(t)$ is the new position of $v^i_{l+1,*}$ calculated by linear interpolation of intrinsic parameters of $\ell^{j_a i}_{l,*}$ solely. $\ell^{j_a i}_{l,*}$ may possess more than one intrinsic parameter, for it can be adjacent to more than one edge, as shown in Fig. 14.

Denote the vertices adjacent to $\ell^{j_a i}_{l,*}$ in level $l-1$ as $v^{k_b}_{l-1,*}$ $b = 0, 1, \ldots, q$. The direction of $\ell^{j_a i}_{l,*}(t)$ is concerned with $\ell^{k_0 j_a}_{l-1,*}(t), \ell^{k_1 j_a}_{l-1,*}(t), \ldots, \ell^{k_q j_a}_{l-1,*}(t)$ in the sense of edge-angle blending, for $\ell^{j_a i}_{l,*}$ shapes an intrinsic angle $\bar{\theta}^{j_a i}_{l,*}(t)$ with respect to each one of $\ell^{k_b j_a}_{l-1,*}(b = 0, 1, \ldots, q)$. We write the formulae as $F_*(\ell^{k_0 j_a}_{l-1,*}, \ell^{k_1 j_a}_{l-1,*}, \ldots, \ell^{k_q j_a}_{l-1,*}, t, \ell^{j_a i}_{l,*})$. Here it is defined as

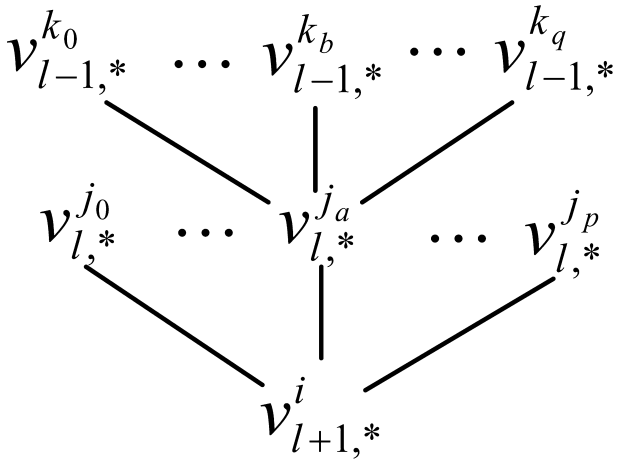

Fig. 14. Connection relation of $v_{l+1,*}^{i}$ to other vertices in higher level.

$$\bar{\theta}_{l,*}^{j_a i}(t) = F_*\left(\ell_{l-1,*}^{k_0 j_a}, \ell_{l-1,*}^{k_1 j_a}, \ldots, \ell_{l-1,*}^{k_q j_a}, t, \ell_{l,*}^{j_a i}\right) = \sum_{b=0}^{q} \beta_l^{k_b j_a} \gamma_*\left(\ell_{l-1,*}^{k_b j_a}, \ell_{l,*}^{j_a i}, t\right) \Big/ \sum_{b=0}^{q} \beta_l^{k_b j_a}.$$

Here $\beta_l^{k_b j_a} = \frac{1}{SN + |\gamma_*(\ell_{l-1,*}^{k_b j_a}, \ell_{l,*}^{j_a i}) - \gamma_*(\ell_{l-1,\bar{*}}^{k_b j_a}, \ell_{l,\bar{*}}^{j_a i})|}$.

Obviously, we have $\bar{\theta}_{l,*}^{j_a i}(t) = \bar{\theta}_{l,\bar{*}}^{j_a i}(1-t)$. Thus

$$\bar{v}_{l+1,A}^{j_a i}(t) = L_{l,A}^{j_a i}(t) e^{i\bar{\theta}_{l,A}^{j_a i}(t)} = L_{l,B}^{j_a i}(1-t) e^{i\bar{\theta}_{l,B}^{j_a i}(1-t)} = \bar{v}_{l+1,B}^{j_a i}(1-t)$$

Since $\alpha_l^{j_a i}$ is independent of t, we get $v_{l+1,A}^{i}(t) = v_{l+1,B}^{i}(1-t)$.

According to mathematical induction principle, we know that the corresponding vertices in $T_A(t)$ and $T_B(1-t)$ have the same coordinates, leading to $T_A(t) = T_B(1-t)$.

References

[1] M. Alexa, Recent advances in mesh morphing, Computer Graphics Forum 21 (2) (2002) 173–197.
[2] M. Alexa, D. Cohen-Or, D. Levin, As-rigid-as-possible shape morphing, in: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New Orleans, Louisiana, USA, 2000, pp. 157–164.
[3] G. Barequet, M. Sharir, Piecewise-linear interpolation between polygonal slices, Computer Vision and Image Understanding 63 (2) (1996) 251–272.
[4] T. Beier, S. Neely, Feature-based image metamorphosis, in: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Chicago, 1992, pp. 35–42.
[5] R.L. Blanding, G.M. Turkiyyah, D.W. Storti, M.A. Ganter, Skeleton-based three-dimensional geometric morphing, Computational Geometry 15 (1–3) (2000) 129–148.
[6] W.J. Che, X.N. Yang, G.Z. Wang, A dynamic approach to skeletonization, Journal of Software 14 (4) (2003) 818–823 (in Chinese).
[7] S. Cohen, G. Elber, R. Bar-Yehuda, Matching of freeform curves, Computer Aided Design 19 (5) (1997) 369–378.

[8] D. Cohen-Or, D. Levin, A. Solomovici, Three-dimensional distance field metamorphosis, ACM Transactions on Graphics 17 (2) (1998) 116–141.
[9] Y.R. Ge, J.M. Fitzpatrick, On the generation of skeletons from discrete euclidean distance maps, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (11) (1996) 1055–1066.
[10] P. Kanonchayos, T. Nishita, Y. Shinagawa, T.L. Kunii, Topological morphing using reeb graphs, in: Proceedings of the First International Symposium on Cyber Worlds, Tokyo, Japan, November 2002, pp. 465–471.
[11] R. Kimmel, D. Shaked, N. Kiryati, A.M. Bruckstein, Skeleton via distance maps and level sets, Computer Vision and Image Understanding 62 (3) (1995) 382–391.
[12] H. Johan, Y. Koiso, T. Nishita, Morphing using curves and shape interpolation techniques, in: Proceedings of Pacific Graphics, HongKong, China, 2000, pp. 348–358.
[13] A. Lerios, C.D. Garfinkle, M. Levoy, Feature-based volume metamorphosis, in: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, CA, 1995, pp. 449–456.
[14] F. Leymarie, M.D. Levine, Simulating the grassfire transform using an active contour model, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1) (1992) 56–75.
[15] M. Mortara, M. Spagnuolo, Similarity measures for blending polygonal shapes, Computers and Graphics 25 (1) (2001) 13–27.
[16] V. Ranjan, A. Fournier, Matching and interpolation of shapes using unions of circles, Computer Graphics Forum (Eurographics'96) 15 (3) (1996), C129–C142.
[17] T.W. Sederberg, P.S. Gao, G.J. Wang, H. Mu, 2-D shape blending: An intrinsic solution to the vertex path problem, in: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Anaheim, 1993, pp. 15–18.
[18] M. Shapira, A. Rappoport, Shape blending using the star-skeleton representation, IEEE Transactions on Computer Graphics and applications 15 (2) (1995) 44–50.
[19] T. Surazhsky, V. Surazhsky, G. Barequet, A. Tal, Blending polygonal shapes with different topologies, Computers and Graphics 25 (1) (2001) 29–39.
[20] S.Takahashi, Y. Kokojima, R. Ohbuchi, Explicit Control of Topological Transitions in Morphing Shapes of 3D Meshes, in: Proceedings of the Ninth Pacific Conference on Computer Graphics and Applications, Tokyo, Japan, October 2001, pp. 70–79.
[21] G. Turk, J.F. O'Brien, Shape transformation using variational implicit functions, in: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, California, USA, 1999, pp. 335–342.
[22] G. Wolberg, Image morphing: a survey, Visual computer 14 (1998) 360–372.