Metamorphosis of Polyhedral Models Using Intrinsic Shape Parameters

by

Sun Yue Man

A thesis
presented to the University of Hong Kong
in fulfillment of the
thesis requirement for the degree of
Master of Philosophy
in
Computer Science
August 1995

Abstract of thesis entitled

Metamorphosis of Polyhedral Models Using Intrinsic Shape Parameters

submitted by

Sun Yue Man

for the degree of Master of Philosophy

at the University of Hong Kong

in August 1995

Metamorphosis, also known as morphing, is the gradual transformation of one shape into another. This thesis deals with metamorphosis of 3-D genus zero polyhedral models. Metamorphosis of polyhedral models generally consists of two subproblems: the correspondence problem and the interpolation problem. A correspondence specifies which face, edge, or vertex of one model is mapped to which face, edge, or vertex of the other model. A correspondence algorithm determines a suitable correspondence between the two models. The interpolation algorithm determines how the faces, edges and vertices are transformed during the animation according to a particular correspondence.

This thesis introduces intrinsic shape parameters into 3-D metamorphosis, and also proposes a set of criteria that a metamorphosis algorithm should satisfy – identity preserving, rotation invariant, translation invariant, and feature preserving. The application of the intrinsic parameters to metamorphosis of polyhedral models is realized by (1) using graph-based representations for polyhedral models, and (2) selecting intrinsic shape parameters to describe the interrelation between the nodes

in the graphs representations. A polyhedral model is considered as a planar graph representing the interrelations between vertices, called the *vertex adjacency graph.* The dual of this planar graph, called the *face adjacency graph*, is also used, representing the interrelations between faces. Interrelating the vertices and faces in the two graphs, intrinsic shape parameters, such as dihedral angles and edge lengths, are used for interpolation. Shape metrics based on intrinsic shape parameters and the two intrinsic representations for polyhedral models are defined.

A solution to the interpolation problem using intrinsic shape parameters is presented and discussed. The interpolation algorithm works for a general one-to-one correspondence, and experience shows that it preserves the features during interpolation, avoids unnatural shrinkage and flipping inside out in the morphing process, and produces more satisfactory results than other existing techniques. Finally, an approach to the correspondence problem is presented.

KEYWORDS: Metamorphosis, Computer Animation, Interpolation, Shape Transformation, Object Representation, Planar Graph, Shape Metric, Correspondence

To My Family

Contents

1 Introduction **1**

1.1 Background 1

1.2 Difficulties 3

1.3 Contributions 5

1.4 Overview of This Thesis 6

2 Related Work **8**

2.1 Problem Definition 9

2.1.1 Correspondence and Interpolation 9

2.2 Previous Work on Correspondence 10

2.2.1 Using Metrics on Shape Difference 11

2.2.2 Merging Topology by Heuristics 14

2.2.3 Using Geometric Information Only 18

2.2.4 Miscellaneous Techniques 20

2.3 Previous Work on Interpolation 21

2.3.1 Existing Solutions to Interpolation Problems 21

3 Basic Criteria **24**

3.1 Direction of Research 24

3.2 Basic Criteria 25

3.3 General Ideas – Intrinsic Shape Parameters 27

4 Representations **30**

4.1 Polyhedral Model Representation 30

4.2 General Correspondence 35

4.2.1 Vertex Correspondence Mapping 35

4.2.2 Super-Graph 36

5 Interpolation of Face Adjacency Graphs **37**

5.1 Notation 37

5.2 Interpolation of Face Adjacency Graphs 38

5.2.1 Computing the First Two Normals 38

5.2.2 Propagation 40

5.3 Interpolation of Connected EDI 42

5.3.1 Results of Using Connected EDI 42

5.4 Interpolation of Connected EGI 43

6 Two-Phase Intrinsic Interpolation **44**

6.1 The Two-Phases 44
6.2 Interpolation of Vertex Adjacency Graphs 45
6.2.1 Computing the First Two Vertices 45
6.2.2 Propagation 45
6.3 Stability of Algorithm 47
6.4 Pseudo Code 48
6.5 Interpolation under General Correspondence 50
6.5.1 Applying the Method for One-to-One Correspondence 50
6.5.2 Adapting Intrinsic Shape Information 51
6.6 Pseudo Code of Two-Phase Intrinsic Interpolation Algorithm 52
6.7 Results and Discussion 53
6.7.1 Properties of Two-Phase Intrinsic Interpolation Algorithm . . 53
6.7.2 Experimental Results 55

7 Establishment of Correspondence **61**

7.1 Shape Metric 61
7.1.1 Deforming Polyhedral Models 61
7.1.2 Metric Based on Connected EGI 63
7.1.3 Metric Based on Connected EDI 67
7.2 The Searching Algorithm 68

8 Conclusions and Future Research **69**

8.1 Future Research 70

Chapter 1

Introduction

1.1 Background

Recently, *metamorphosis* has emerged as a hot topic in computer graphics. It is the process of smoothly transforming one object to another, and is also known as *morphing* [1]. The concept is best illustrated by figures. In Fig. 1.1, the leftmost object is morphed into the rightmost. There is no restriction on how one object should be morphed into another, and the upper and lower sequences show the metamorphosis in two possible manners.

Metamorphosis of 3-D objects is useful in achieving special effects in computer animation [27], animation of biological evolution [21], and can also be used to create new models combining features of existing designs in industrial design [7]. Although metamorphosis of 2-D projected images of 3-D objects [3, 47] may be able to produce visually similar effects, morphing of the 3-D models, instead of their projected images, allows the animation to be independent of projection transformation, and thus the view point used, and it also gives the shape of the morphed objects. This is

[1] Morphing originally refers to the metamorphosis of images rather than objects. However, people nowadays use these two terms interchangeably.

In this thesis, a correspondence specification is formulated. An approach to the correspondence problem and preliminary solutions to the interpolation problem using intrinsic shape parameters are presented. Finally, a more sophisticated interpolation algorithm is presented. Unlike [41], our interpolation algorithm is not restricted to one-to-one correspondence, and experience shows that it preserves object features during interpolation, and avoids unnatural shrinkage and flipping inside out in the morphing process.

1.4 Overview of This Thesis

This thesis is organized as a progressive study of our approach using intrinsic shape parameters. Each chapter shows a stage in the research and is based on the foundation provided by the previous chapters.

Chapter 2 surveys the related previous work on the problem of metamorphosis. This survey is not exhaustive, but serves to set the context for the contributions of this thesis and also summarizes the problems and difficulties of this topic.

The focus in Chapter 3 is on establishing the direction of the research that takes into account the problems uncovered by the analysis in Chapter 2. In this chapter, criteria for the solutions to the problem are also proposed as the base of our research. It is hoped that these criteria will be useful in future in developing new metamorphosis algorithms.

Chapter 4 discusses the new representations for polyhedral models used in our morphing algorithm. These representations are also part of our contribution, and they form underlying structures that enable the use of intrinsic shape parameters.

The aim of Chapter 5 is to present the interpolation algorithms using intrinsic shape parameters. The use of two different sets of intrinsic shape parameters is discussed. Some preliminary results and problems encountered are also discussed.

A more sophisticated interpolation algorithm is discussed in detail in Chapter 6. We will call it the *two-phase intrinsic interpolation* [44].

Chapter 7 develops shape metrics for correspondence algorithms based on the intrinsic parameters. Some preliminary results and problems encountered are also discussed.

Chapter 8 concludes this thesis with a summary of the main results of the thesis and comments on future research.

Chapter 2

Related Work

Dynamic and kinematic animation, and simulation of evolution, like metamorphosis, can be used to change the shape of an object but their methods rely on information that is not used by metamorphosis methods, such as physical laws [2, 37], key frames, reference skeletons [45], moving point constraints [38], and evolution behaviors [8]. The first metamorphosis algorithm appeared in 1988 in [21]. Subsequently, many techniques of metamorphosis [5, 7, 6, 10, 12, 18, 24, 27, 28, 29, 30, 40, 41] have been proposed. It is interesting to note that nearly all techniques are different in nature and bear very little similarity among them. The basic assumptions, criteria for producing the in-between objects, and the shape parameters to be interpolated in these techniques are completely different.

Most existing morphing algorithms solve the correspondence and the interpolation problems in separate steps. The only exceptions are the Minkowski sum technique and the alpha shape geometric morphing, in which the correspondence and interpolation problems are coupled and solved together. However, theoretically, we can recognize that they use linear interpolation carried out in E^3 or E^4. So we will separate the discussion in two parts: the correspondence and the interpolation. We will first focus on the approaches to the correspondence problem and then those to the interpolation problem. In the discussion, we will concentrate on techniques

that produce an in-between object that is topologically well-defined, rather than just a set of points in space. So techniques that are based on voxels [30], volume models [32], or Fourier transform [24], will not be considered.

2.1 Problem Definition

Throughout this thesis, A and B will denote the two input objects. Unless otherwise specified, the objects are polyhedral models. In the metamorphosis between A and B, we transform A to B continuously from $t = 0$ to $t = 1$ under a prescribed correspondence between the vertices of A and B.

Definition 2.1 *A polyhedral model is said to be topologically valid or topologically well-defined if the model satisfies the following properties:*

1. *each edge is incident to two and only two vertices;*
2. *each edge is incident to two and only two faces;*
3. *there is at most one edge incident to both of two given vertices;*
4. *each vertex is incident to at least three faces;*
5. *each face is incident to at least three vertices.*

2.1.1 Correspondence and Interpolation

This section introduces the reader to the concept of correspondence and interpolation. For illustration, consider examples of metamorphosis of 2-D polygons. In Fig. 2.1 and 2.2. two different correspondences are used and shown in two *correspondence matrices* [40]. The rows and columns represent vertices of the input polygons

$A = [A_0, A_1, \ldots, A_{m-1}]$ and $B = [B_0, B_1, \ldots, B_{n-1}]$, respectively. The whole correspondence is specified by a *correspondence path*, which consists of consecutive line segments going through the correspondence matrix as shown in Fig. 2.1 and 2.2. Each grid point $[i, j]$ represents a vertex correspondence between A_i and B_j. A line segment $([i, j], [i, j + 1])$ represents a correspondence between vertex A_i and edge B_jB_{j+1}; a line segment $([i, j], [i + 1, j])$ represents a correspondence between edge A_iA_{i+1} and vertex B_j; and a segment $([i, j], [i+1, j+1])$ represents a correspondence between edge A_iA_{i+1} and edge B_jB_{j+1}. In Fig. 2.1, the linear vertex path is used for interpolation; in Fig. 2.2, the animation is done by a rotation.

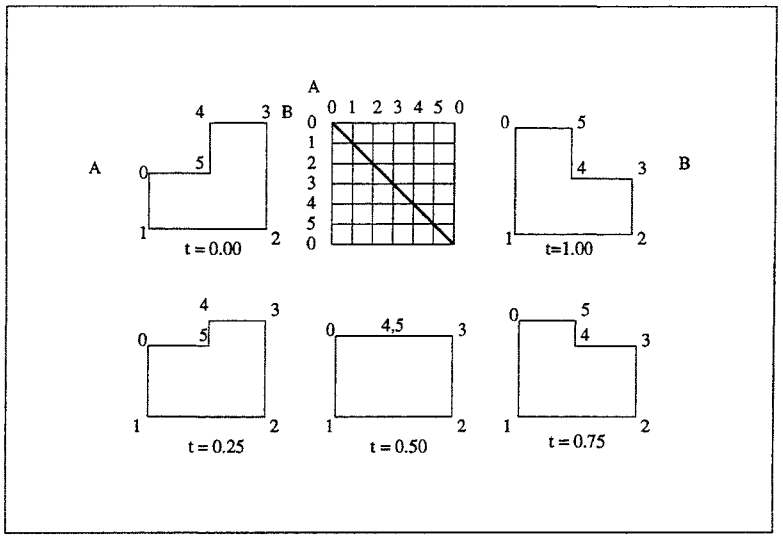

Figure 2.1: *Correspondence Example 1*

While the correspondence matrix can be used to represent 2-D correspondence, unfortunately, there is so far no effective representation for correspondence of 3-D polyhedral models.

2.2 Previous Work on Correspondence

The existing approaches to the correspondence problem can be generally classified into the following categories: (a) using a quantitative approach to measure the shape

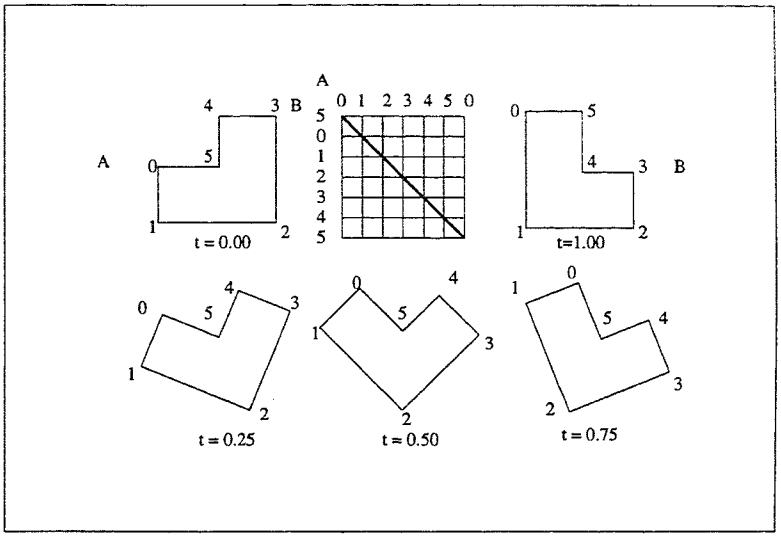

Figure 2.2: *Correspondence Example 2*

difference and finding the correspondence with the minimum measure; (b) using a heuristic to find two models with the same topology but each having the same shape as one of the input objects; (c) using geometric information only; and (d) others.

2.2.1 Using Metrics on Shape Difference

A correspondence specifies which vertex, edge, and face of one object will also be transformed into which vertices, edges, or faces of the other object. Usually, when the correspondence is different, the transformation and the in-between shapes will also be different. It would be very ideal if we could find a correspondence such that the in-between object's *shape difference* between A and B is minimized. In the approach of using metrics on shape difference, there is a tentative shape metric associated with each correspondence. The correspondence with the minimal shape difference is the one we will use for interpolation. Although shape difference has been extensively investigated [14, 15, 16, 22, 26, 36, 39, 42, 48, 49], no universal shape metric exists. That is because how much one shape is different from another is very subjective.

The techniques of minimizing the distances between corresponding face centroids and vertex pairs [21] and intrinsic shape interpolation [40] use different shape metrics and are discussed in the following sections.

(a) Minimizing distances of corresponding face centroids and vertex pairs

It was proposed in [21] to use the distances between corresponding face centroids and vertex pairs as the measure of shape difference. The correspondence is established first at the face level and then at the vertex level. In establishing correspondence at the face level, A and B are first normalized: A and B are scaled so that their farthest face centroids from the origin are both 1 and their centers of gravity of the respective set of the face centroids are both at the origin. The faces are matched by establishing a bijection g from the set of face centroids of A to the set of face centroids of B such that $\sum \|a_i - g(a_i)\|^\mu$ is minimized, where a_i is a face centroid of A, $g(a_i)$ is a face centroid of B, and μ is some constant. When one object, A say, has more faces than B, the extra faces of A are matched to the face of B that corresponds to the extra faces' closest neighbours. Similarly, the vertices of corresponding faces are matched by minimizing the distances of the corresponding vertices.

Remarks. Based on the face centroid coordinates and vertex coordinates, this metric depends on the relative orientation of the two inputs. Thus, the correspondence algorithm is not rotation invariant. Besides, if we change one vertex of an input polyhedral model and keep all other vertices unchanged, the normalized vertices or centroids will be changed as the center of gravity is changed. This small change can result in a big change in the minimization. Moreover, as mentioned in [28], the faces of the in-between model may consist of isolated faces, because this metric ignores the topologies of the objects in establishing the correspondence of the faces.

(b) Physically based shape blending

The physically based shape blending [40] works only for 2-D polygons. We include it here as it provides some insights into the morphing of 3-D polyhedral models.

The physically based shape blending [40] measures shape difference by the work done required to transform one object into the object. Two kinds of work done are considered: (1) work for stretching each pair of corresponding line segments and, (2) work for bending each pair of corresponding interior angles. Stretching an edge from length L_A to L_B requires the work

$$W_s = k_s \frac{|L_A - L_B|^{e_s}}{(1 - c_s)\min(L_A, L_B) + c_s \max(L_A, L_B)}$$

where k_s, c_s, and e_s are user-defined constants. The work to bend an angle $\theta(t)$ from $t = 0$ to $t = 1$, where $\theta(0) = \theta_A$ and $\theta(1) = \theta_B$, is similarly defined by

$$W_b = \begin{cases} k_b(\delta\theta + m_b\delta\theta^*)^{e_b} & \text{if } \theta(t) \text{ never goes to zero} \\ k_b(\delta\theta + m_b\delta\theta^*)^{e_b} + p_b & \text{otherwise} \end{cases}$$

where k_b, e_b, m_b, p_b are user-defined constants, $\delta\theta = |\theta_A - \theta_B|$, $\delta\theta^*$ measures how far $\theta(t)$ deviates from montonicity, m_b penalizes non-monotonic $\theta(t)$, and p_b penalizes if $\theta(t)$ ever goes to zero.

Remarks. The metric of shape difference in the physically based shape blending [40] is based on edge lengths and internal angles. Since the edge lengths and internal angles between adjacent edges are invariant under translation and rotation, the metric is invariant under translation and rotation. This resolves the situation in Fig. 1.2 as the correspondence is independent of the relative orientation. Moreover, each parameter is locally defined. The disadvantage of the dependence on absolute coordinates in the technique of minimizing the distances of corresponding face centroids and vertex pairs [21] is thus overcome. However, this method works only for 2-D polygons.

(c) Discussion

The technique of minimizing distances of corresponding face centroids and vertex pairs [21] and physically based shape blending [40] use two different metrics to measure shape difference. The design of a good shape metric is crucial to using the approach of shape metric. However, it is hard to design a good metric. Besides, this approach has the disadvantage that the minimization computation is expensive. This makes real-time computation and real-time animation impossible.

2.2.2 Merging Topology by Heuristics

The approach of merging topology aims to solve a major difficulty in metamorphosis – the topologies of the two input polyhedral models are different in general. The difficulty is overcome by finding two auxiliary objects with the same topologies but each having the same shape as one of the input objects.

This approach includes the projection methods [28, 29], the super-object technique [5], and the technique using contours [7].

(a) Projection methods

Several morphing techniques making use of projection are proposed in [28, 29]. In these techniques, the topologies of the two input polyhedral models are merged as follows: (1) obtain projected models of the input models by projecting the input models' vertex/edge/face network onto the surface of unit sphere; (2) the two projected models are merged by clipping the projected faces of one object against the projected faces of the other; (3) a common network model is then obtained by mapping the merged model back to the original surfaces of the input models.

When the method of projection is first introduced [28], it applies to star-shaped polyhedra only and the central projection is used. Using different projection map-

pings for different classes of objects is proposed in a later work [29]. For example, in dealing with extruded objects, the cross section can be mapped to its convex hull by a method that recursively reduces concavities of a 2-D polygon [13]. The resulting convex model is projected to the unit sphere using a central projection. Another physically based approach is also introduced in [29] as an alternative to the projection methods above when they cannot apply. It is designed to deal with polyhedra without specific characteristics, such as being star-shaped or extruded. Treating an object as having flexible surfaces, it simulates the process of inflating the model until it becomes convex. Vertices and edges are treated as masses and springs respectively, and spring force is applied along an edge together with internal air pressure applied to the centroid of each face in the direction of its outward normal.

Remarks. In establishing a correspondence between objects with similar features, the features may not be matched correctly by the projection methods [28, 29]. Moreover, the projection method [28] is limited to a special class of objects, star-shaped polyhedra. Although some solutions have been proposed for the class of extruded objects in [29], we do not believe that making use of more and more heuristics and model knowledges will lead to good solutions to the correspondence problem. Moreover, the techniques using projection are difficult to be extended to general polyhedral models, such as the one as simple as in Fig. 2.3. Although the simulation method in [29] is proposed to solve this problem, it is time-consuming to run the simulation and still unknown what class of objects this simulation can apply to; moreover, it is not always possible to inflate a polyhedral model to become convex. In some cases, whether this method works becomes known only after the simulation has been carried out.

(b) Super-object

The super-object technique [5] aims to find two auxiliary models, called *super-objects*, such that their topologies are the same, each of them is in the same shape

2.2.3 Using Geometric Information Only

In the approach of using geometric information, the shape of the object is considered as a set of points. The in-between object depends on the geometry of the input objects. The Minkowski sum technique [27] and the alpha shape geometric morphing [12] use this approach.

(a) Using the Minkowski sum

In the technique using the Minkowski sum, the correspondence and interpolation problems are coupled and solved simultaneously. For any A and $B \subset \Re^n$, the Minkowski sum, [1] denoted $A \oplus B$, is defined to be $\{a + b | a \in A, b \in B\}$. The in-between model at time t is defined to be $(1-t) * A \oplus t * B$, denoted as $A \oplus_t B$, where the "$*$" operator denotes the scaling of a set.

Assume A and B to be polyhedral models. The algorithm to compute $A \oplus_t B$ is based on the propositions that any face of $A \oplus B$ can be expressed as $V_A \oplus F_B$, $F_A \oplus V_B$ or $E_A \oplus E_B$, for some vertex V_A, edge E_A, face F_A in A and some vertex V_B, edge E_B, face F_B in B. These faces are said to be of type VF, FV, and EE, respectively. That is, we can compute a superset of the faces of $A \oplus_t B$ by considering all the VF, FV, and EE faces. It can also be proven that the boundary of the Minkowski sum of two polyhedra is a subset of the boundary of the Minkowski sum of their boundaries. Therefore, the in-between object can be displayed by rendering all the faces in the superset. By considering the orientation of the faces, vertices, and edges of the polyhedra, we can improve the efficiency of the algorithm but we will not cover the details here.

Remarks. From the theoretical point of view, the correspondence in the Minkowski sum technique [27] is such that each point contained in the 3-D model of one object corresponds to every point contained in the other one. Thus, we have a many-to-

[1] Properties of Minkowski sum can be found in [20].

many correspondence and this results in a lack of locality. This is why the Minkowski sum technique [27] fails to produce the identity correspondence in Fig. 1.3.

(b) The geometric morphing of alpha shapes

In the geometric morphing of alpha shapes [12], the two input models given to the morphing algorithm are in the form of alpha shapes [2]. The alpha complexes of A and B are projected to two complexes on a paraboloid in $\Re^4$, and these are interpolated in $\Re^4$ using Minkowski sum and the resulting complex is projected back in $\Re^3$ to form the mixed complex. Finally, the in-between object is computed as the alpha shape of the mixed complex.

Remarks. It should be noted that although for every polyhedron there exists a finite set of points and a value of α so that the polyhedron is the α-shape of the point set [12], there is no known algorithm to find this set of points and the α for a given polyhedron.

Unlike the technique of Minkowski sum [27], the alpha shape morphing [12] is identity preserving and this is achieved by the use of standard methods from convex geometry [17] in dealing with non-convexity. However, as the topological information is also considered, the resulted in-between model may be disconnected.

(c) Discussion

The approaches using geometry information only are based on well understood concepts, Minkowski sum and alpha shapes, and can be formulated rigorously. The approach can also apply to all 3-D objects. However, as mentioned in the sections above, these techniques suffer from topological problems.

[2] Refer to [11, 12] for the definition and properties of alpha shapes and alpha complexes.

2.2.4 Miscellaneous Techniques

Simplex-based animation

Animating transformation of models in [10] is based on the *simplex mesh representation* of objects. A simplex mesh can be considered as a network of independent particles with fixed connectivities. The simplex mesh, like triangulations, can represent all orientable surfaces.

Transformation of a simplex mesh involves (1) the dynamics of each vertex governed by some internal and external constraint forces; and (2) mesh transformation operations that alter the structure of the mesh. The dynamics of a vertex P_i of the simplex mesh is given by

$$m\frac{d^2P_i}{dt^2} = -\gamma\frac{dP_i}{dt} + F_{int} + F_{ext}$$

where t is the time, m is the mass of the node, γ is the damping factor, F_{int} is the internal constraint force, and F_{ext} is the external force set by the user or some other constraints. The internal constraint force is defined by some shape parameters of the mesh and it brings the mesh to its rest shape when no external constraint is applied. Simplex mesh structures can be locally modified without exhibiting any irregularity in the mesh connectivity. There are four basic mesh transformation operations to transform one mesh to another: edge removal, face splitting, handle creation, and handle removal.

The simplex mesh representing A is first fit on object B by setting an attractive external force that drags the surface model of A close to the 3-D data of B, and then the mesh is modified interactively by a sequence of the basic mesh transformation operations for it to have the precise shape of B. To animate the metamorphosis, the sequence of basic transformations obtained by the fitting process above is applied to the simplex mesh of A. Then the shape parameters defining internal constraint force of B are then assigned to the resulting mesh. The internal constraint force F_{int} then brings the mesh to the shape of B.

The problem with the simplex-based animation is that it is unknown how to automatically find out the sequence of the basic mesh transformation operations to have the precise shape of B.

2.3 Previous Work on Interpolation

2.3.1 Existing Solutions to Interpolation Problems

Although the approaches to correspondence problem are very diversed, most existing techniques for solving the interpolation problem fall into one of the following three categories: (a) linear vertex path; (b) independent interpolation of each vertex pair using a curved path; (c) intrinsic shape interpolation.

(a) Linear vertex path

In the linear vertex path, each corresponding vertex pair is linearly interpolated independently. It is the most widely used technique in practice. The super-object technique [5], and the method of minimizing the distances of corresponding face centroids and vertex pairs [21] all use the linear vertex path. It is simple, computationally inexpensive, suitable for real-time animation, and works well for morphing highly dissimilar objects.

In the super-object technique [5] and the technique of minimizing the distances of corresponding centroids and vertices [21], the interpolation algorithms consider an object as a collection of independent points. Shrinkage usually occurs for two objects that differ by a rotation. For example, consider the metamorphosis of two congruent tetrahedra, one of which is 180° opposite to the other. In this case, an in-between tetrahedron can even flip totally inside out. Figure 2.4 shows morphing one tetrahedron into another similar one at time $t = 0, 0.2, 0.4, 0.6, 0.8, 1$, from right to left. The instance at $t = 0.4$ flips totally inside out.

Chapter 3

Basic Criteria

3.1 Direction of Research

Intuitively, an animation of metamorphosis should be smooth. Although human beings can easily tell whether a morphing process is visually smooth or not, translating the human perception into qualitative criteria is difficult. Different ways of animation may please people with different tastes. For example, it is hard to say any of the two ways of morphing in Fig. 1.1 is better than the other. However, we believe that there are some basic criteria that a morphing algorithm should satisfy. We propose four such criteria in this chapter to serve as the starting point of our research.

In analyzing the correspondence problem of 3-D objects, some observations stand out. First, features of the two objects are ignored in all the existing metamorphosis algorithms when setting up the correspondence. All the existing 3-D metamorphosis algorithms are not designed to match the features. Second, just like the morphing techniques in [12, 21, 27], using only geometry of the objects suffers from topological problems: probably topologically disconnected in-between objects or failure to preserve local properties. Last, using only topologically information also has its

problems. Established using the topologies of input polyhedral models only, the correspondence used by the super-object method[5] often causes distortion in the animation.

As for the interpolation problem, only very simple techniques for interpolating 3-D objects – the linear vertex path and the curved path – are available. These interpolation techniques assume that the polyhedral models to be interpolated are just a set of independent points, ignoring the topological structure and the inter-relations between the points, and thus usually result in distorted and unnaturally shrunk in-between objects. An example of morphing using the linear vertex path is shown in Fig. 2.4, where polyhedral models of the similar shapes but of different orientations are transformed.

In this thesis, the success of the 2-D physically-based shape blending [40] and intrinsic shape interpolation [41] is examined and extended. Although this extension leads to a very expensive algorithm, it has the capability of solving a lot of problems that are not solved by other approaches. Following this framework, we use the approach of minimizing the metric defined in terms of intrinsic shape parameters and the approach of topological merging in establishing correspondence.

3.2 Basic Criteria

Although there is no consensus on how to morph objects, we believe that there are some basic criteria a metamorphosis algorithm must satisfy. For instance, when the two input objects are identical, the in-between model should always be the same as the input objects. We call this property *identity preserving*. The shape of the in-between model should also be independent of the displacements and orientations of the two objects. The in-between model should also preserve the features common to both input objects. For example, in morphing a cat to a rabbit, a leg as a prominent feature should always remain as a leg.

We propose four basic criteria a metamorphosis algorithm should satisfy. Let $A_{\mathbf{x}}$ be the translation of object A by vector $\mathbf{x}$. Let $A^{\theta,\mathbf{a}}$ be the rotation of object A by angle θ about the axis of rotation $\mathbf{a}$. Let $A \otimes_t B$ be the in-between polyhedral model of A and B at time t. Let $\varphi_{A,B}$ be the correspondence between A and B. If the in-between model is computed under a specific correspondence φ, then the in-between model is denoted as $A \otimes_t^{\varphi} B$.

1. **Identity preserving:**
   For any $t \in [0,1]$, $A \otimes_t A = A$. When the two input objects are identical, the in-between object should always be the same as the input objects.

2. **Translation invariant:**
   For any vectors $\mathbf{x}, \mathbf{y}$ and $t \in [0,1]$, $A_{\mathbf{x}} \otimes_t B_{\mathbf{y}} = (A \otimes_t B)_{\mathbf{z}}$ for some $\mathbf{z}$. That is, the morphing should be independent of the relative position of A and B. Note that $\mathbf{z} = (1-t)\mathbf{x} + t\mathbf{y}$ if the animation speed is uniform.

3. **Rotation invariant:**
   For any axes of rotation, $\mathbf{a}, \mathbf{b}$, angles $\theta, \beta \in [0, 2\pi]$, and $t \in [0,1]$, $A^{\theta,\mathbf{a}} \otimes_t B^{\beta,\mathbf{b}} = (A \otimes_t B)^{\gamma,\mathbf{c}}$ for some γ and $\mathbf{c}$. The morphing should be independent of the relative orientation of A and B.

4. **Feature preserving:**
   If there are features common to both input objects, the features should be preserved during the metamorphosis.

The last criteria is informal as there is no general agreement on which part of an object is a feature and how to tell whether a feature is preserved or not in a morphing. But in some applications, this criterion is meaningful to most people. One example is preserving the legs or the head of two different animals during morphing.

A metamorphosis algorithm is often described by its correspondence algorithm and interpolation algorithm, so it is desirable to define also the following criteria for them. We say a correspondence algorithm is

1. identity preserving if $\varphi_{A,A} = I$, where I is the identity correspondence;
2. translation invariant if for any vectors $\mathbf{x}, \mathbf{y}, \varphi_{A_\mathbf{x},B_\mathbf{y}} = \varphi_{A,B}$;
3. rotation invariant if for any axes of rotation $\mathbf{a}, \mathbf{b}$ and $\theta, \beta \in [0, 2\pi], \varphi_{A^{\theta,\mathbf{a}},B^{\beta,\mathbf{b}}} = \varphi_{A,B}$;
4. feature preserving if the features common to both input objects are matched;

and an interpolation algorithm is

1. identity preserving if for any $t \in [0,1], A \otimes_t^I A = A$, where I is the identity correspondence;
2. translation invariant if for any vectors $\mathbf{x}, \mathbf{y}$, $t \in [0,1]$, and correspondence φ, $A_\mathbf{x} \otimes_t^\varphi B_\mathbf{y} = (A \otimes_t^\varphi B)_\mathbf{z}$ for some $\mathbf{z}$;
3. rotation invariant if for any axes of rotation $\mathbf{a}, \mathbf{b}, \theta, \beta \in [0, 2\pi], t \in [0,1]$, and correspondence $\varphi, A^{\theta,\mathbf{a}} \otimes_t^\varphi B^{\beta,\mathbf{b}} = (A \otimes_t^\varphi B)^{\gamma,\mathbf{c}}$ for some γ and $\mathbf{c}$;
4. feature preserving if features common to both input objects, which are matched under a feature preserving correspondence, are preserved during the metamorphosis.

It should be noted that a metamorphosis algorithm is identity preserving/translation invariant/rotation invariant/ feature preserving if both its correspondence and interpolation algorithms are identity preserving/translation invariant/rotation invariant/ feature preserving.

3.3 General Ideas – Intrinsic Shape Parameters

No existing 3-D metamorphosis algorithm satisfies all the above criteria. In particular, none of them is rotation invariant and feature preserving.

In 2-D morphing, if we use the physically-based shape blending [40] to establish the correspondence and use the intrinsic shape interpolation [41], the first three

criteria will be satisfied. Moreover, it is shown in [41] that given a proper correspondence, features of the objects are well preserved in the animation. In establishing the correspondence, some parameters can be set by the user to adjust the relative contributions by the stretching work done and the bending work done. The intrinsic shape interpolation has demonstrated its effectiveness in feature preserving [40]. The success of these algorithms can be explained by the fact that intrinsic parameters are invariant under translation and rotation. Consequently, the use of intrinsic parameters becomes a natural choice in devising 3-D metamorphosis algorithms satisfying the four criteria.

The first problem encountered in generalizing intrinsic parameters to 3-D is the representation of the objects. We represent a polyhedral model as an *oriented plane graph* [44], called the *vertex adjacency graph (VAG)*. The vertex adjacency graph is composed of the vertices and edges of the polyhedron [5]. The *face adjacency graph (FAG)*, which can be thought of as the dual of the vertex adjacency graph, is used to represent the interrelations between faces. The nodes of the face adjacency graph and vertex adjacency graph are interrelated by intrinsic parameters, such as edge lengths, $\tilde{l}$, dihedral angles, l, and interior angles, $\tilde{\theta}$ (See Fig. 3.1). We will use the intrinsic parameters to interpolate the face adjacency graph and the vertex adjacency graph.

The second problem of using intrinsic parameters is which intrinsic shape parameters are to be used. There are a lot of intrinsic parameters for polyhedral models, such as edge lengths, interior angles, dihedral angles, outface angles [44], etc. In the

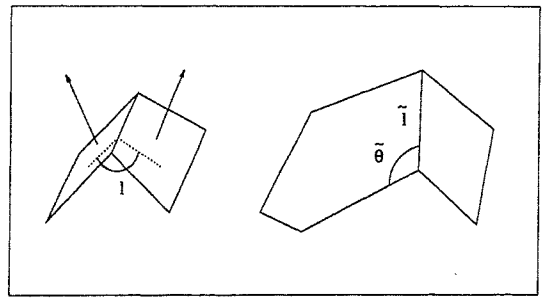

Figure 3.1: *Examples of intrinsic parameters.*

forthcoming chapters, these representations and the parameters will be defined and discussed for their application in metamorphosis.

Chapter 4

Representations

This chapter presents the representations of the polyhedral models used in correspondence and interpolation, which makes use of the intrinsic parameters. A representation for correspondence is also defined.

4.1 Polyhedral Model Representation

A polyhedron can be represented as a planar graph. Since a planar graph can have more than one embedding on the plane, we will restrict ourselves to the particular embedding that represents the topology of the polyhedron. An embedded planar graph is called a *plane graph* [19]. In order to distinguish a plane graph from its mirror-image graph, we assign an orientation to the plane graph, as done in [46]. This results in an *oriented plane graph.*

Definition 4.1 *An oriented plane graph is a plane graph in which each circuit bounding a region is a cyclically ordered set of links that bound the region in a counterclockwise order.*

Definition 4.2 *The vertex adjacency graph (VAG) is an oriented plane graph that is associated with a polyhedron. Each node of the VAG represents a vertex of the polyhedron and there is a link between two nodes if the corresponding vertices are linked by an edge of the polyhedron. Each region of VAG corresponds to a face of the polyhedron. The order of the links around a region of VAG is the same as that of the corresponding edges around the corresponding face in the polyhedron. Each node contains the vertex coordinates.*

The VAG represents a polyhedron uniquely, and we will use the VAG as the primary representation of a polyhedral model.

Definition 4.3 *The face adjacency graph (FAG) is an oriented plane graph that is associated with a polyhedron. A node contains the outward unit normal vector of the corresponding face. The link connecting two nodes contains a flag of value +(positive), −(negative), or =(coplanar), indicating that the corresponding faces form a convex, concave dihedral angle, or that the two faces are coplanar, respectively.*

For simplicity, we assume that the flag in FAG only takes the value + or −. Each region in the FAG corresponds to a vertex of the polyhedron. Note that an FAG cannot define a polyhedron uniquely. As we will see later, additional geometric information is needed to make the FAG a complete representation of a polyhedron. It should be clear that as plane graphs, FAG and VAG are dual to each other, and the FAG of a polyhedron can be constructed if the VAG of the polyhedron is given. Since each node of VAG corresponds to a vertex of the polyhedron, we will use the terms "node of VAG" and "vertex" interchangeably. Similarly, we will use the terms "node of FAG" and "face" interchangeably.

Definition 4.4 *Two VAGs or two FAGs are* topologically equivalent *if the vertices, edges, and faces of the underlying graphs can be put in a one-to-one correspondence that keeps the ordering of the bounding edges surrounding every pair of corresponding regions.*

We now introduce two FAG-based representations for polyhedral models.

Definition 4.5 *When the nodes of the FAG contain the areas of the corresponding facets, the FAG will be called a connected Extended Gaussian Image, or a connected EGI for short.*

The connected EGI is an extension of the Extended Gaussian Image (EGI) [23], which is an object representation in which an object is represented as a set of independent weighted points on the Gaussian sphere with the weight being the area of the facet. Any convex polyhedron can be uniquely represented by an EGI representation [23, 34]. We also have a similar property for the connected EGI.

Theorem 4.1 *Uniqueness Theorem for the connected EGI*

For any given two polyhedra, no matter convex or not, with convex faces only, if they have the same connected EGI then they are the same polyhedron up to translation.

The proof is similar to that for the theorem of Alexandrov [34]. Before giving the proof of Theorem 4.1, we state the following lemma whose proof can be found in [34].

Lemma 4.2 *Given two convex polygons X and Y, if they cannot be strictly embedded one in the other by a translation, we assign +/− sign to each of edge of X and Y, such that if the edge of X is longer than the corresponding parallel edge of Y with the same direction of outward normal* [1] *(if there is no corresponding edge, we use the vertex of Y such that the edge of Y is supporting to this vertex and the outward normal vector of X is pointing outside from the vertex) then, we mark a + sign at the edge of X. If it is shorter, we mark a − sign there. If they are equal, we do not*

[1] An outward normal of an edge of a 2-D polygon is a vector perpendicular to the edge and pointing towards the exterior of the polygon.

mark it. Then, on a circuit of polygon X or Y, we have $\geq$ *4 transitions of + to −, or − to + sign.*

Proof of Theorem 4.1

Let A and B be two polyhedra such that they are topologically equivalent and there is a one-to-one correspondence between their faces. Suppose they have convex faces only. Corresponding faces have the same outwards normal and area. Since they have the same adjacency information, we have the one-to-one correspondence between edges.

Suppose that there is some pairs of their edges that are not equal in length. We mark each of the edge of polyhedron A as explained in the lemma. Since each edge will correspond only to a single edge or vertex, edges will be marked consistently. Then, there are two cases for the corresponding faces to have the same area:

> *Case 1:* they are congruent; then all their edges are unmarked.
> *Case 2:* they cannot be strictly embedded into each other by a translation; then around a circuit of the face of A, there are more than or equal to 4 transitions.

Let H be the planar graph such that it is obtained by deleting all faces with no mark from the dual graph of the polyhedron surface. Each vertex in H has more than or equal to 4 transitions of sign changes.

Let F be the number of faces of H. Let V be the number of vertices of H. Let E be the number of edges of H. Let M be the sum of transitions of sign changes over all faces in H. So, M is also the sum of transitions of sign changes over all vertices in H. Then, $M \geq 4V$. Let a_i be the number of i-sided regions. Then, $2E = \sum_{i \geq 3}(i * a_i)$ (where $i \geq 3$ since all the faces in the original graph before edge and vertex deletion are at least 3-sided, after deletion, they are still at least 3-sided). For a face of sides a_i, the maximum number of transitions of sign changes is less than or equal to $2 * \lfloor i/2 \rfloor$.

So,

$$\begin{aligned} M &\leq \sum_{i\geq 3}(2 * \lfloor i/2 \rfloor * a_i) \\ 4V &\leq \sum_{i\geq 3}(2 * \lfloor i/2 \rfloor * a_i) \end{aligned}$$

and

$$\begin{aligned} V + F &= E + 2 \\ V &= E - F + 2 \\ &= \sum(i/2 * a_i) - \sum(a_i) + 2 \\ &= \sum((i/2 - 1) * a_i) + 2 \end{aligned}$$

So,

$$\begin{aligned} \sum(1/2 * \lfloor i/2 \rfloor * a_i) &\geq \sum((i/2 - 1) * a_i) + 2 \\ \sum(\lfloor i/2 \rfloor * a_i) &\geq \sum((i - 2) * a_i) + 4 \\ -4 &\geq \sum((i - 2 - \lfloor i/2 \rfloor) * a_i) \\ &= 2a_5 + 2a_6 + 4a_7 + 4a_8 + ... \\ &\geq 0 \end{aligned}$$

So, we get a contradiction. Therefore, all the corresponding edge lengths of A and B are equal, and thus A is equal to B up to translation. *Q.E.D.*

Definition 4.6 *If each node of the FAG contains the perpendicular distance from the corresponding facet to its center of gravity, the FAG is called a connected Extended Distance Image, or a connected EDI for short.*

Theorem 4.3 *Uniqueness Theorem for the connected EDI*

For any given two polyhedra, if they have the same connected EDI, then they are the same polyhedron up to translation.

Proof of Theorem 4.3

Let A and B be two polyhedra such that they are topologically equivalent and there is a one-to-one correspondence between their faces. First consider the case that the centers of gravity of A and B are at the origin. If their corresponding perpendicular distances are the same, each pair of the corresponding faces have the same equation. And since they are topologically equivalent, all corresponding edges and vertices are the same. Thus, A and B are the same.

When their centers of gravity are not coincident at the origin, they are congruent up to translation. *Q.E.D.*

4.2 General Correspondence

In general, for two input polyhedra A and B, their vertex adjacency graphs are not topologically equivalent. Then a general correspondence has to be used, in which one vertex of A may correspond to many vertices, edges or faces of B, and vice versa. In the following, the representation of a valid general correspondence for polyhedral models is formulated. Two concepts are needed to define a valid general correspondence: the *vertex correspondence mapping* and the *super-graph.*

4.2.1 Vertex Correspondence Mapping

Let M and A be polyhedral models. Let VAG^M and VAG^A be the vertex adjacency graphs of M and A. Let $M.V$ and $A.V$ be the set of nodes of VAG^M and VAG^A,

respectively. A vertex correspondence mapping $hv^A : M.V \rightarrow A.V$ is defined as $hv^A(x) = y$ if the vertex x of VAG^M corresponds to the vertex y of VAG^A. We use $hv^A(VAG^M)$ to denote the graph with the set of nodes being the range of hv^A and the set of links being all the links mapped from the links of VAG^M.

Let $M.E$ and $A.E$ be the set of links of VAG^M and VAG^A, respectively. Let $M.F$ and $A.F$ be the set of faces of VAG^M and VAG^A, respectively. We call $hv^A : M.V \rightarrow A.V$ a *valid vertex correspondence mapping* if it satisfies the following conditions:

1. hv^A is onto.
2. If $(m_1, m_2) \in M.E$, then either $(hv^A(m_1), hv^A(m_2)) \in A.E$ or $hv^A(m_1) = hv^A(m_2) \in A.V$.
3. For any $e \in A.E$, there exists $(m_1, m_2) \in M.E$ such that $(hv^A(m_1), hv^A(m_2)) = e$.
4. For any face $r \in M.F$, r is mapped to either a face with the same orientation, an edge, or a vertex of A.

4.2.2 Super-Graph

If hv^A satisfies conditions 1, 2, and 3, then $hv^A(VAG^M)$ is isomorphic to VAG^A. In this case, the only difference between $hv^A(VAG^M)$ and VAG^A, as oriented plane graphs, may be the embeddings and orientation. If hv^A further satisfies condition 4, each face of M is mapped to either a face, an edge or a vertex of A. Then $hv^A(VAG^M)$ is topologically equivalent to VAG^A. In this case, we call VAG^M a *super-graph* of VAG^A.

A valid general correspondence between A and B is defined by (1) a super-graph VAG^M of VAG^A and VAG^B; and (2) two valid vertex correspondence mappings $hv^A : M.V \rightarrow A.V$ and $hv^B : M.V \rightarrow B.V$.

Chapter 5

Interpolation of Face Adjacency Graphs

5.1 Notation

Let FAG^A, FAG^B, and $FAG^{M(t)}$ be the face adjacency graphs of A, B, and $M(t)$, respectively, and let VAG^A, VAG^B, and $VAG^{M(t)}$ be their respective vertex adjacency graphs.

We use P to refer to a genus zero polyhedron, and let its number of faces and vertices be n and m, respectively. The face adjacency graph and vertex adjacency graph of P will be denoted by FAG and VAG, respectively. By definition, n and m are also the the number of the regions and nodes of VAG, respectively. Throughout this thesis, a notation with the name of a polyhedron as superscript, such as FAG^A, FAG^B, and $FAG^{M(t)}$, will refer to the elements for that polyhedron. When it is without superscript, the notation is used for a general genus zero polyhedron.

The above notation will be used from Chapter 5 onwards.

5.2 Interpolation of Face Adjacency Graphs

In this section, the one-to-one correspondence between VAG^A and VAG^B is first assumed for simplicity; the interpolation under a general correspondence will be discussed in Section 6.5. Under a one-to-one correspondence, VAG^A, VAG^B, and $VAG^{M(t)}$ are assumed to be topologically equivalent, so are FAG^A, FAG^B, and $FAG^{M(t)}$. In this case, $n^A = n^B = n^{M(t)}$ and $m^A = m^B = m^{M(t)}$. For simplicity, we denote $\overline{n} = n^A = n^B = n^{M(t)}$ and $\overline{m} = m^A = m^B = m^{M(t)}$.

The interpolation of FAG^A and FAG^B produces $FAG^{M(t)}$. Let $f_i^{M(t)}$ be the i^{th} face of the in-between polyhedral model $M(t)$, and $N_i^{M(t)}$ be the outward unit normal of $f_i^{M(t)}$. When interpolating FAG^A and FAG^B, our goal is to compute the normal vector $N_i^{M(t)}$ at time t, for $i = 0, 1, \ldots, \overline{n} - 1$, We will start with two initial faces, f_0 and f_1, and then use the intrinsic relation between faces to compute $N_i^{M(t)}$ for $i = 2, 3, \ldots, n - 1$, by propagation.

5.2.1 Computing the First Two Normals

To proceed, we introduce the *outface normal* of an edge. Let f_b and f_a be two adjacent faces sharing an edge in a polyhedron. Let $T_{b,a}$ be the unit vector parallel to the edge and oriented counterclockwise with respect to the face f_b. The *outface normal* $e_{b,a}$ to the edge with respect to f_b is defined by the unit vector $T_{b,a} \times N_b$. That is, $e_{b,a}$ is parallel to face f_b and points towards the exterior of f_b. Fig. 5.1 and 5.2 illustrate the definition in two different situations and their meanings are also illustrated on the *Gaussian sphere* [23], denoted **S**. The Gaussian sphere can be thought of as a sphere on which every point corresponds to a unit direction vector. Thus, all the unit normal vectors can be represented by points on the Gaussian sphere. Let $l_{b,a}$ be the dihedral angle between the faces f_b and f_a.

The normal vector $N_0^{M(t)}$ is interpolated using the shortest arc on **S** connecting N_0^A and N_0^B. $N_1^{M(t)}$ can be computed if the dihedral angle $l_{0,1}^{M(t)}$ and the outface

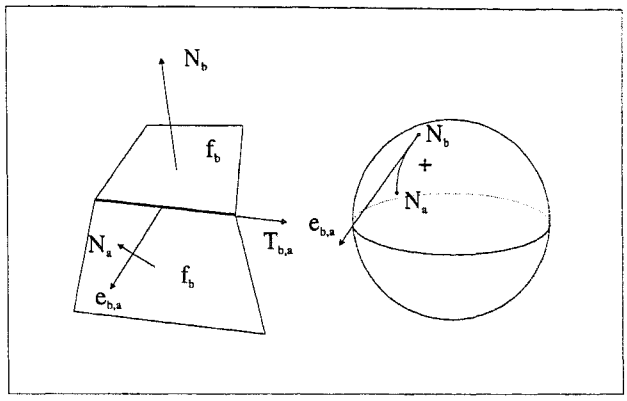

Figure 5.1: *Definition of the outface normal $e_{b,a}$ of a convex edge.*

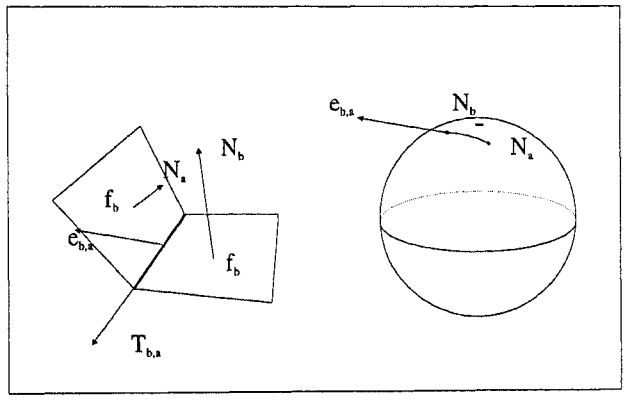

Figure 5.2: *Definition of the outface normal $e_{b,a}$ of a reflexive edge.*

normal $e_{0,1}^{M(t)}$ are known. The outface normal $e_{0,1}^{M(t)}$ is computed as follows. Since N_0^A and $e_{0,1}^A$ are orthogonal, $\mathbf{F}^A = \{N_0^A, e_{0,1}^A, N_0^A \times e_{0,1}^A\}$ is an orthogonal frame. Similarly, $\mathbf{F}^B = \{N_0^B, e_{0,1}^B, N_0^B \times e_{0,1}^B\}$ is also an orthogonal frame. Let $\mathbf{K}(t)$, $t \in [0,1]$, be the shortest rotational motion about a fixed axis that rotates $\mathbf{F}^A$ into $\mathbf{F}^B$, i.e., $\mathbf{K}(0) = \mathbf{I}$ and $\mathbf{K}(1)\mathbf{F}^A = \mathbf{F}^B$. Then set $e_{0,1}^{M(t)} = \mathbf{K}(t)e_{0,1}^A$. See Fig. 5.3. The dihedral angle $l_{0,1}^{M(t)}$ is computed as $(1-t)l_{0,1}^A + tl_{0,1}^B$. Thus, the normal $N_1^{M(t)}$ is computed by rotating $N_0^{M(t)}$ about the axis $N_0^{M(t)} \times e_{0,1}^{M(t)}$ for an angle $l_{0,1}^{M(t)}$. To make the computation of $N_1^{M(t)}$ stable, we choose the first two faces $f_0^{M(t)}$ and $f_1^{M(t)}$ such that $\min\{l_{0,1}^A, l_{0,1}^B\}$ is maximized.

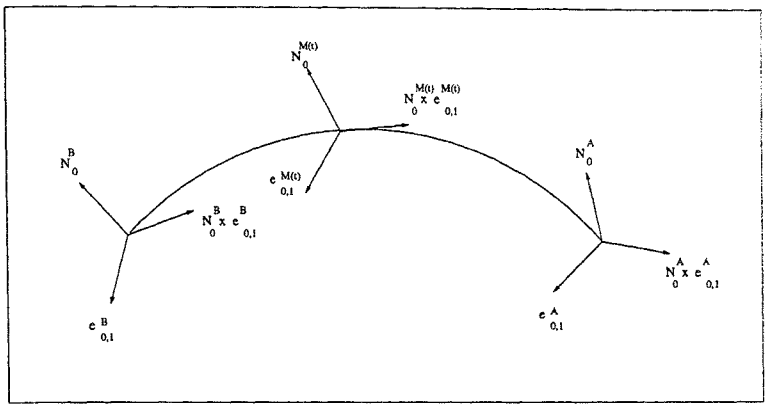

Figure 5.3: Rotation of the first two normals

5.2.2 Propagation

Suppose f_c, f_b, and f_a are three consecutive nodes around a region of the FAG. Let $\theta_{c,b,a}$ be the angle of rotation needed to rotate $e_{b,c}$ to $e_{b,a}$ about the rotation axis N_b. It is called the *outface angle* from $e_{b,c}$ into $e_{b,a}$ about N_b. When there is no ambiguity about the meaning of $\theta_{c,b,a}$, we would simply write it as θ. Let $\mathbf{R}_N(\alpha)$ denote the rotation about axis N by an angle α. Then, as $e_{b,c}$ is determined by N_c and N_b, the vectors N_c, N_b, and N_a are related by the following relations (see Fig. 5.4 and 5.5):

$$e_{b,a} = \mathbf{R}_{N_b}(\theta) e_{b,c} \tag{5.1}$$

$$N_a = \mathbf{R}_{N_b \times e_{b,a}}(l_{b,a}) N_b \tag{5.2}$$

Since the in-between polyhedron $M(t)$ should also satisfy these two relations, the equations:

$$e_{b,a}^{M(t)} = \mathbf{R}_{N_b^{M(t)}}(\theta^{M(t)}) e_{b,c}^{M(t)} \tag{5.3}$$

$$N_a^{M(t)} = \mathbf{R}_{N_b^{M(t)} \times e_{b,a}^{M(t)}}(l_{b,a}^{M(t)}) N_b^{M(t)} \tag{5.4}$$

will be used to compute $N_a^{M(t)}$ when $N_b^{M(t)}$ and $N_c^{M(t)}$ are known. The terms $\theta^{M(t)}$ and $l_{b,a}^{M(t)}$ used in Eq. (5.3) and (5.4) are computed by interpolation

$$\theta^{M(t)} = (1-t)\theta^A + t\theta^B \tag{5.5}$$

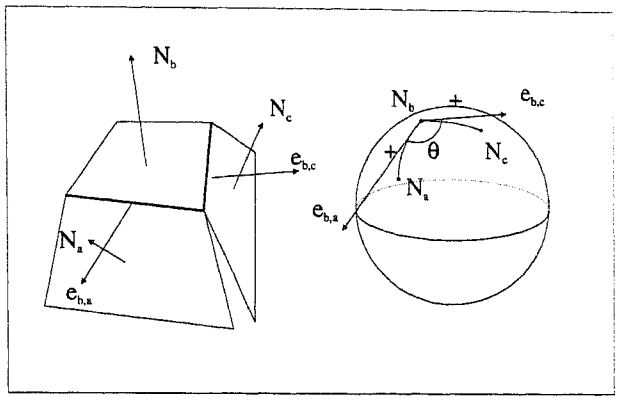

Figure 5.4: *Relation of N_a, N_b, and N_c when $f_a^{M(t)}$ and $f_b^{M(t)}$ share a convex edge.*

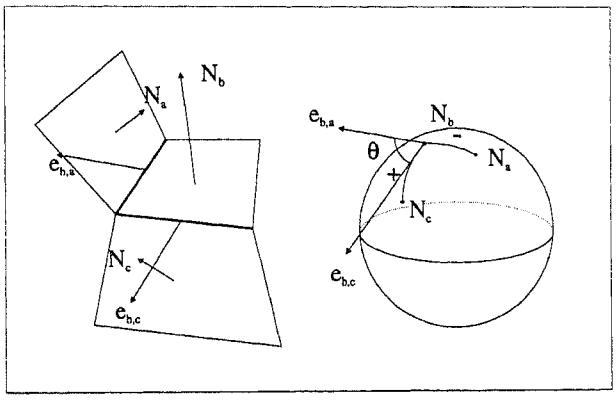

Figure 5.5: *Relation of N_a, N_b, and N_c when $f_a^{M(t)}$ and $f_b^{M(t)}$ share a reflexive edge.*

$$l_{b,a}^{M(t)} = (1-t)l_{b,a}^{A} + tl_{b,a}^{B} \tag{5.6}$$

where θ^A, θ^B, $l_{b,a}^A$, and $l_{b,a}^B$ are directly computed from FAG^A and FAG^B. Thus, $e_{b,a}^{M(t)}$ is determined by Eq. (5.3) and $N_a^{M(t)}$ is then determined by Eq. (5.4). In other words, given FAG^A and FAG^B, $N_a^{M(t)}$ can be determined if $N_b^{M(t)}$ and $N_c^{M(t)}$ are known. Consequently, $N_i^{M(t)}$, $i = 2, 3, \ldots, \overline{n} - 1$, can be computed once $N_0^{M(t)}$ and $N_1^{M(t)}$ are known. There are numerous choices of the order of computing the $N_b^{M(t)}$, and breadth-first search is used in our implementation.

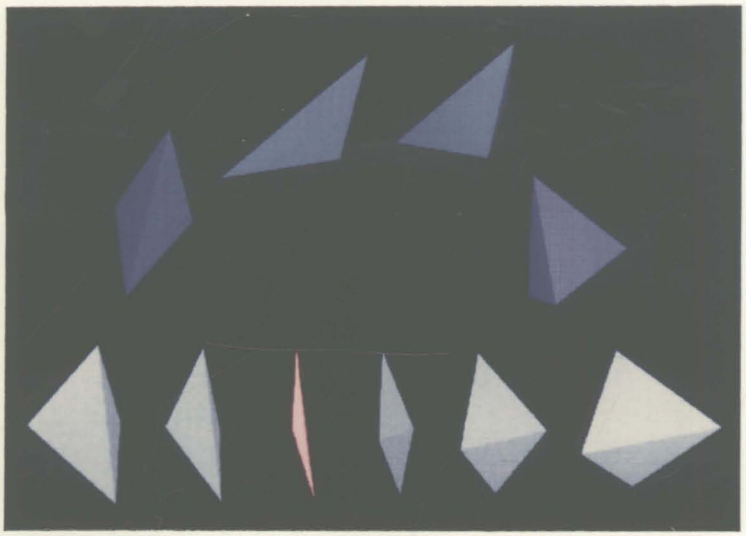

Figure 5.6: *Interpolation of the tetrahedra in Fig. 2.3*

5.3 Interpolation of Connected EDI

In this and the next subsections, we will consider applying the interpolation schemes of FAG above to the connected EDI and the connected EGI. Let μ_i be the perpendicular distance from the polyhedron's center of gravity to the face f_i. The distance component of the connected EDI of A and B can be interpolated independently by

$$\mu_i^{M(t)} = (1-t)\mu_i^A + t\mu_i^B \tag{5.7}$$

5.3.1 Results of Using Connected EDI

Two simple examples are shown in Fig. 5.6 and 5.7 to illustrate the interpolation of the connected EDI. Fig. 5.6 shows that the interpolation of face adjacency graphs eliminates the total inversion of interior surfaces that results from the linear vertex interpolation. In Fig. 5.7, although the general shape is changing, there are defects at the vertices of the in-between object: more than three faces do not intersect at a common vertex when they are supposed to.

When interpolating the connected EDI, the interpolated connected EDI may not

Before interpolating the vertex adjacency graphs, we need to interpolate the face adjacency graphs as done in Chapter 5. We will not repeat the details here.

6.2 Interpolation of Vertex Adjacency Graphs

This section discusses the interpolation of VAG^A and VAG^B. Let $V_j^{M(t)}$ be the j^{th} vertex of $M(t)$, and let $v_j^{M(t)}$ be the vertex coordinates of $V_j^{M(t)}$. Our goal is to compute the vertex coordinates $v_j^{M(t)}$ for $j = 0, 1, \ldots, \overline{m} - 1$. We first compute two starting vertices which are determined by the two starting faces in interpolation of FAG^A and FAG^B. Then using the intrinsic relation between vertices, we compute $v_j^{M(t)}$ by propagation for $j = 2, 3, \ldots, \overline{m} - 1$.

6.2.1 Computing the First Two Vertices

We denote the two vertices of the edge shared by the faces f_0 and f_1 by V_0 and V_1, oriented such that V_1 follows V_0 counterclockwise around f_0. We will compute $v_0^{M(t)}$ and $v_1^{M(t)}$ first. Let $\tilde{l}_{a,b}$ be the length of the edge $\overline{v_a v_b}$. First, set $v_0^{M(t)} = (1-t)v_0^A + tv_0^B$. Then $v_1^{M(t)}$ is computed such that the direction of $v_1^{M(t)} - v_0^{M(t)}$ is parallel to $N_0^{M(t)} \times e_{0,1}^{M(t)}$ and $\|v_1^{M(t)} - v_0^{M(t)}\| = \tilde{l}_{0,1}^{M(t)} = (1-t)\tilde{l}_{0,1}^A + t\tilde{l}_{0,1}^B$.

6.2.2 Propagation

Suppose V_c, V_b, and V_a are three consecutive vertices of a face f_i. Let $\tilde{e}_{b,a}$ be the unit edge tangent vector parallel to the direction of $v_a - v_b$ and let $\tilde{\theta}_{c,b,a}$ be the interior angle $\angle V_c V_b V_a$. When V_c, V_b, and V_a take the counterclockwise order around f_i, $\tilde{\theta}_{c,b,a}$ is negative; otherwise, it is positive. When there is no ambiguity about the defining vertices for the angle, we will simply write it as $\tilde{\theta}$.

We need to find a relation that involves intrinsic parameters only and links v_c,

v_b, and v_a. See Fig. 6.1.

$$v_a = v_b + \tilde{l}_{b,a}\mathbf{R}_{N_i}(\tilde{\theta})\tilde{e}_{b,c} \tag{6.1}$$

This relation gives the equation

$$v_a^{M(t)} = v_b^{M(t)} + \tilde{l}_{b,a}^{M(t)}\mathbf{R}_{N_i^{M(t)}}(\tilde{\theta}^{M(t)})\tilde{e}_{b,c}^{M(t)} \tag{6.2}$$

to compute $v_a^{M(t)}$ when $v_b^{M(t)}$ and $v_c^{M(t)}$ are known. The terms $\tilde{\theta}^{M(t)}$ and $\tilde{l}_{b,a}^{M(t)}$ used

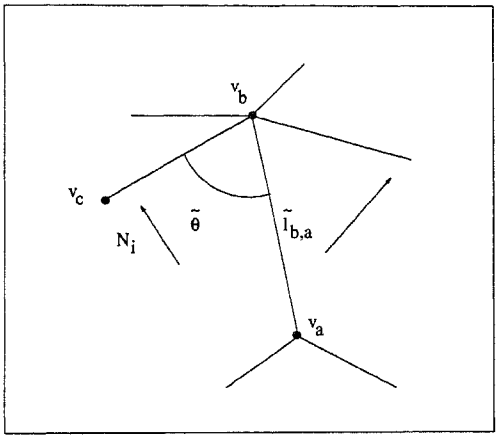

Figure 6.1: *Relation of v_a, v_b, and v_c.*

in Eq. (6.2) are computed by the interpolation

$$\tilde{\theta}^{M(t)} = (1-t)\tilde{\theta}^A + t\tilde{\theta}^B \tag{6.3}$$

$$\tilde{l}_{b,a}^{M(t)} = (1-t)\tilde{l}_{b,a}^A + t\tilde{l}_{b,a}^B \tag{6.4}$$

Note that in Eq. (6.2), the N_i is unknown if we only interpolate the vertex adjacency graphs. That is why we need to introduce a two-phase interpolation algorithm.

Thus, when $v_0^{M(t)}$ and $v_1^{M(t)}$ are known, $v_j^{M(t)}$, $j = 2, 3, \ldots, \overline{m} - 1$ can be determined by propagation. Again, breadth-first search is used in our implementation for propagation.

6.3 Stability of Algorithm

The interpolation of FAG or VAG is a one-pass algorithm. The result of the interpolation depends on the initial values and the order of computation. A small perturbation of initial values can introduce a big difference in later stages of computation. In Eq. (6.2), when $\|v_c^{M(t)} - v_b^{M(t)}\| \ll \tilde{l}_{b,a}^{M(t)}$, a perturbation $\delta v_c^{M(t)}$ to $v_c^{M(t)}$ can make a big change to $v_a^{M(t)}$. The same is true of Eq. (5.4). Thus, if the initial values are not set properly, the in-between object can be highly distorted. We circumvent this problem by using a heuristic to make the algorithm numerically more stable.

In the intermediate stage of interpolating VAG^A and VAG^B, suppose we are to compute the vector $v_a^{M(t)}$. To compute $v_a^{M(t)}$, we first have to find a pair of vertices, $V_b^{M(t)}$ and $V_c^{M(t)}$, such that the coordinates of $V_b^{M(t)}$ and $V_c^{M(t)}$ are already computed, and $V_a^{M(t)}$, $V_b^{M(t)}$, and $V_c^{M(t)}$ are consecutive around a face of $VAG^{M(t)}$. Let there be $\tilde{d}_a$ pairs of such vertices for $V_a^{M(t)}$, denoted by $V_{b_k}^{M(t)}$ and $V_{c_k}^{M(t)}$, $k = 0, 1, .., \tilde{d}_a - 1$. Fig. 6.2 illustrates the situation. Each of these $\tilde{d}_a$ pairs is used to compute a $v_{a_k}^{M(t)}$

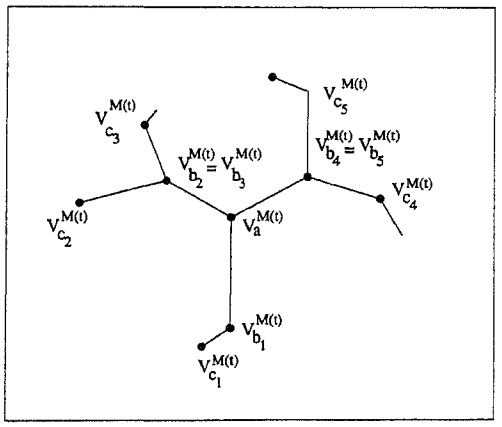

Figure 6.2: *Computing* $v_a^{M(t)}$.

with a weight

$$\tilde{w}_{a_k} = \frac{\tilde{l}_{b_k,c_k}^{M(t)}}{\tilde{l}_{b_k,a_k}^{M(t)}} = \frac{\|v_{b_k}^{M(t)} - v_{c_k}^{M(t)}\|}{\|v_{b_k}^{M(t)} - v_{a_k}^{M(t)}\|}. \tag{6.5}$$

The final $v_a^{M(t)}$ will be computed as

$$\frac{\sum_{k=0}^{\tilde{d}_a-1} \tilde{w}_{a_k} v_{a_k}^{M(t)}}{\sum_{k=0}^{\tilde{d}_a-1} \tilde{w}_{a_k}}$$

The choice of the weight by Eq. (6.5) is made so that a pair $v_{c_k}^{M(t)}$, $v_{b_k}^{M(t)}$ with shorter length $\|v_{c_k}^{M(t)} - v_{b_k}^{M(t)}\|$ contributes less to the final $v_a^{M(t)}$.

The same principle applies to the interpolation of FAG^A and FAG^B. That is, $N_a^{M(t)}$ is assigned the direction of $\sum w_{a_k} N_{a_k}^{M(t)}$ with the weight w_{a_k} given by:

$$w_{a_k} = \frac{l_{b_k,c_k}^{M(t)}}{l_{b_k,a_k}^{M(t)}} \tag{6.6}$$

Experiments show that the above remedy greatly alleviates the numerical instability of $v_a^{M(t)}$ and $N_a^{M(t)}$, and leads to a more robust algorithm.

6.4 Pseudo Code

The algorithms for interpolating the face adjacency graphs and the vertex adjacency graphs are given below in pseudo code:

Algorithm *FAGInterpolation*(VAG^A, VAG^B, t)
Input: Vertex Adjacency Graphs VAG^A and VAG^B of polyhedra A and B.
t is the time in interval [0,1].
Output: Face Adjacency Graph $FAG^{M(t)}$ of polyhedron $M(t)$.
1. construct FAG^A from VAG^A;
   construct FAG^B from VAG^B;
2. choose two initial adjacent nodes of $FAG^{M(t)}$ and assign them two unit normal vectors; //section 5.2
3. **while** there is still an uncomputed node of $FAG^{M(t)}$

do

3.1 select next node $f_a^{M(t)}$ by breadth-first search

3.2 **for all** pairs $(f_{c_k}^{M(t)}, f_{b_k}^{M(t)})$ such that
$N_{c_k}^{M(t)}$ and $N_{b_k}^{M(t)}$ have been computed and
$f_{c_k}^{M(t)}$, $f_{b_k}^{M(t)}$ and $f_a^{M(t)}$ are
consecutive around a region of $FAG^{M(t)}$

do

compute $N_{a_k}^{M(t)}$ by Eq. (5.4);

compute w_{a_k} by Eq. (6.6);

3.3 compute $N_a^{M(t)} = \frac{\sum_k (w_{a_k} N_{a_k}^{M(t)})}{\| \sum_k (w_{a_k} N_{a_k}^{M(t)}) \|}$;

4. **return** $FAG^{M(t)}$;

Algorithm *VAGInterpolation*(VAG^A, VAG^B, $FAG^{M(t)}$, t)

Input: Vertex Adjacency Graphs VAG^A and VAG^B of
polyhedra A and B.
Face Adjacency Graph $FAG^{M(t)}$ of $M(t)$.
t is the time in interval [0,1].

Output: Vertex Adjacency Graph $VAG^{M(t)}$ of the in-between object.

1. assign initial vertex coordinates to $V_0^{M(t)}$ and $V_1^{M(t)}$; //section 6.2
2. **while** there is still an uncomputed node of $VAG^{M(t)}$

do

3.1 select next vertex $V_a^{M(t)}$ by breadth-first search;

3.2 **for all** pairs $(V_{c_k}^{M(t)}, V_{b_k}^{M(t)})$ such that
$v_{c_k}^{M(t)}$ and $v_{b_k}^{M(t)}$ have been computed and
$V_{c_k}^{M(t)}$, $V_{b_k}^{M(t)}$, and $V_a^{M(t)}$ are
consecutive around a region of $VAG^{M(t)}$

do

compute $v_{a_k}^{M(t)}$ by Eq. (6.2);

compute $\tilde{w}_{a_k}$ by Eq. (6.5);

3.3 compute $v_a^{M(t)} = \frac{\sum_k (\tilde{w}_{a_k} v_{a_k}^{M(t)})}{\sum_k (\tilde{w}_{a_k})}$;

4. return $VAG^{M(t)}$;

6.5 Interpolation under General Correspondence

This section presents how to interpolate VAG^A and VAG^B under a general correspondence. The correspondence is assumed to be valid throughout the discussion. We first replace VAG^A and VAG^B by two vertex adjacency graphs which are topologically equivalent to $VAG^{M(t)}$, so that the method used for the case of the one-to-one correspondence can be directly applied. However, this may cause degeneracy problems in interpolating the intrinsic parameters and using the relations Eq. (5.3), (5.4), (5.5), (6.2), and (6.3). We will resolve it by using the intrinsic shape information in one object to approximate the degenerate part of another object.

6.5.1 Applying the Method for One-to-One Correspondence

VAG^A is replaced by a vertex adjacency graph, denoted VAG^{M_A}, which is topologically equivalent to $VAG^{M(t)}$. The node values are determined as follows: if $hv^A(V_a^M) = V_b^A$, $v_a^{M_A} = v_b^A$. From VAG^{M_A}, we construct FAG^{M_A}. We renumber the vertices of VAG^{M_A} and faces of FAG^{M_A} in such a way that $V_j^{M_A}$ of VAG^{M_A} corresponds to $V_j^{M(t)}$ of $VAG^{M(t)}$, and $f_i^{M_A}$ of FAG^{M_A} corresponds to $f_i^{M(t)}$ of $FAG^{M(t)}$. VAG^B is replaced by VAG^{M_B} that is similarly defined. FAG^{M_B} is constructed similarly.

Unfortunately, the edges and faces of VAG^{M_A}, VAG^{M_B}, FAG^{M_A}, and FAG^{M_B} may be degenerate. If a face $f_i^{M_A}$ is mapped to an edge or a vertex of A, the normal vector $N_i^{M_A}$ is not well defined; consequently, the related outface angles and outface normals are not defined. If an edge $(V_a^{M_A}, V_b^{M_A})$ of VAG^{M_A} is mapped to a vertex of VAG^A, the edge tangent $\tilde{e}_{a,b}^{M_A}$ is not defined, neither are the related

interior angles. The same problem occurs in VAG^{M_B} and FAG^{M_B}. Thus, Eq. (5.3), (5.4), and (5.5) cannot be used for the interpolation of FAG^{M_A} and FAG^{M_B}, neither can Eq. (6.2) and (6.3) for the interpolation of VAG^{M_A} and VAG^{M_B}. To overcome the problem, we need to approximate the values of the undefined outward normals, outface normals, outface angles, edge tangents, and interior angles for degenerate elements of VAG^{M_A} or VAG^{M_B}. The approximation is made by adapting the intrinsic shape information in one object to the corresponding degenerate part of another object. We first compute the undefined outward normals and then by definition, the undefined outface normals, outface angles, edge tangents, and interior angles will follow.

6.5.2 Adapting Intrinsic Shape Information

We adapt the intrinsic shape information from the other object to approximate the undefined outward normals. The method of approximating the undefined outward normals $N_a^{M_A}$ and $N_a^{M_B}$ is similar to the idea of interpolating FAG^A and FAG^B in Section 5.2. We assume that there is at least one pair of well-defined corresponding outface normals $e_{b,a}^{M_A}$ and $e_{b,a}^{M_B}$ to serve as the start point of the approximation. This condition is satisfied in most practical situations.

Suppose $f_a^{M_A}$ is mapped to a vertex or an edge of VAG^A, thus $N_a^{M_A}$ is not defined. We assume that $f_a^{M_B}$ is defined, for otherwise, the super-graph can always be simplified so that the assumption holds. To approximate $N_a^{M_A}$, we first have to find two pairs of corresponding faces, $f_c^{M_A}$ and $f_c^{M_B}$, and $f_b^{M_A}$ and $f_b^{M_B}$, with well-defined outward normals such that $f_c^{M_A}$, $f_b^{M_A}$ and $f_a^{M_A}$, and $f_c^{M_B}$, $f_b^{M_B}$, and $f_a^{M_B}$ are consecutive around a pair of corresponding regions in FAG^{M_A} and FAG^{M_B}, respectively. Let the total number of such pairs be κ_a. Denote these pairs of faces by $f_{b_k}^{M_A}$ and $f_{c_k}^{M_A}$, $k = 0, 1, \ldots, \kappa_a - 1$. Here we simply use θ^{M_A} to denote $\theta_{c_k,b_k,a_k}^{M_A}$. To simulate the nondegenerate case, the normal $N_{a_k}^{M_A}$ could be computed by

$$e_{b_k,a_k}^{M_A} = \mathbf{R}_{N_{b_k}^{M_A}}(\theta^{M_A}) e_{b_k,c_k}^{M_A}$$

$$N_{a_k}^{M_A} = \mathbf{R}_{N_{b_k}^{M_A} \times e_{b_k,a_k}^{M_A}}(l_{b_k,a_k}^{M_A}) N_{b_k}^{M_A}$$

However, as the outface angle θ^{M_A} in the first equation is not defined, we replace θ^{M_A} by its counterpart in VAG^{M_B}, i.e., set $\theta^{M_A} = \theta^{M_B}$. Similarly, we set $l_{b_k,a_k}^{M_A} = l_{b_k,a_k}^{M_B}$. As a result, $e_{b_k,a_k}^{M_A} = \mathbf{R}_{N_{b_k}^{M_A}}(\theta^{M_B}) e_{b_k,c_k}^{M_A}$, and $N_{a_k}^{M_A} = \mathbf{R}_{N_{b_k}^{M_A} \times e_{b_k,a_k}^{M_A}}(l_{b_k,a_k}^{M_B}) N_{b_k}^{M_A}$, $k = 0, 1, \ldots, \kappa_a - 1$. The outward normal $N_a^{M_A}$ is then approximated as the unit vector of $\sum_{k=0}^{\kappa_a - 1} w_{a_k} N_{a_k}^{M_A}$. Now, the outface normals and edge tangents related to $f_a^{M_A}$ can be determined from $N_a^{M_A}$ and the related outface angles and interior angles can also easily be determined. The undefined quantities in VAG^{M_B} and FAG^{M_B} are approximated similarly.

6.6 Pseudo Code of Two-Phase Intrinsic Interpolation Algorithm

Algorithm *IntrinsicInterpolation*(VAG^A, VAG^B, $VAG^{M(t)}$, t)

Input: Vertex Adjacency Graphs VAG^A and VAG^B of polyhedra A and B.
$VAG^{M(t)}$ is the given in-between Vertex Adjacency Graph whose node values are yet to be computed.
t is the time in interval [0,1]

Output: Vertex Adjacency Graph $VAG^{M(t)}$ of the in-between object

1. Replace VAG^A and VAG^B by VAG^{M_A} and VAG^{M_B}, respectively;
2. Construct FAG^{M_A} and FAG^{M_B} from VAG^A and VAG^B, respectively;
3. Approximate the undefined normals $N_a^{M_A}$, $N_a^{M_B}$ // section 6.5
4. Approximate the undefined outface normals, edge tangents, outface angles and interior angles // section 6.5
5. Compute $FAG^{M(t)}$ at time t by *FAGInterpolation*(VAG^{M_A}, VAG^{M_B}, t)
6. Compute $VAG^{M(t)}$ at time t by *VAGInterpolation*(VAG^{M_A}, VAG^{M_B}, $FAG^{M(t)}$, t);
7. **return** $VAG^{M(t)}$;

6.7 Results and Discussion

In this section, some properties of the two-phase intrinsic interpolation algorithm and some experimental results are discussed.

6.7.1 Properties of Two-Phase Intrinsic Interpolation Algorithm

Property 6.1 *If A and B are congruent polyhedral models and the given correspondence is the identity correspondence, i.e., all vertices, edges, and faces of A correspond to their counterparts of B, then, $M(t) = A = B$ for all $t \in [0, 1]$.*

The proof is trivial and therefore omitted. By this property, the two-phase intrinsic interpolation algorithm is identity preserving when the given correspondence is the identity correspondence.

As we can see, all operations – approximating normals, computing initial normals and vertex coordinates, stabilized propagation of computing normals and vertex coordinates – involve only the interpolation of intrinsic parameters, which are invariant under rotation and translation. So, we have the following properties:

Property 6.2 *The intrinsic interpolation is rotation invariant.*

Property 6.3 *The intrinsic interpolation is translation invariant*

Lemma 6.1 *To produce an in-between model by the intrinsic interpolation requires computing $O(E)$ normals and $O(E)$ vertices, where E is the number of edges of the super-graph $VAG^{M(t)}$.*

Proof: In the interpolation of the FAG^A and FAG^B, computing a normal in $FAG^{M(t)}$ by propagation requires a certain number of applications of the Eq. (5.3)

and (5.4). Each application of Eq. (5.3) and (5.4) produces a normal. It is noted that each application of Eq. (5.3) and (5.4) involves an outface angle, and each outface angle is involved at most once in the whole algorithm. So, we have:

total number of computed normals

$\leq$ total number of outface angles of $FAG^{M(t)}$

$= 2 \times$ total number of edges of $FAG^{M(t)}$

$= 2 \times$ total number of edges of $VAG^{M(t)}$

$= 2E$ where E is the total number of edges of $VAG^{M(t)}$.

Similarly, in the interpolation of the VAG^A and VAG^B, we have:

total number of computed vertices

$\leq$ total number of interior angles of $VAG^{M(t)}$

$= 2 \times$ total number of edges of $VAG^{M(t)}$

$= 2E$

Since we need to compute two initial normals and two initial vertices, the intrinsic algorithm computes $O(E)$ normals and $O(E)$ vertices to produce an in-between model. *Q.E.D.*

Lemma 6.2 *We say a super-graph is of minimal number of edges if there exists no edge (m_1, m_2) of the graph such that $hv^A(m_1) = hv^A(m_2)$ and $hv^B(m_1) = hv^B(m_2)$, where hv^A and hv^B are the vertex correspondence mappings of the super-graph. Then, for a super-graph of minimal number of edges, $E \leq E^A + E^B$, where E^A and E^B are the number of edges of VAG^A and VAG^B.*

Proof: For a super-graph of minimal number of edges, each edge is mapped to (1) an edge in VAG^A and an edge in VAG^B; (2) an edge in VAG^A and a vertex in VAG^B; or (3) a vertex in VAG^A and an edge in VAG^B. Therefore, $E \leq E^A + E^B$ is directly followed. *Q.E.D.*

By Lemma 6.1 and 6.2, and that computing each normal by Eq. (5.3) and (5.4) and each vertice by Eq. (6.2) take constant time, we have the following property.

Property 6.4 *The intrinsic algorithm computes an in-between model in* $O(E^A + E^B)$ *time for a super-graph of minimal number of edges, where* E^A *and* E^B *are the number of edges of* VAG^A *and* VAG^B.

As a comparison, we recall that an in-between model is computed in $O(V')$ time, where $V' = \max(n^A, n^B)$, by the linear or curved vertex path.

6.7.2 Experimental Results

The algorithm has been implemented and tested on SGI Indigo/XZ graphics workstations with CPU R4000. Experiments show that the algorithm works well for many cases where the two morphed objects have similar features, and if the correspondence is properly specified, the features are preserved during morphing. Fig. 1.1 is an example of showing different ways of morphing between two objects, in which the only difference is the input correspondence. Fig. 6.3 shows the interpolation of two tetrahedra that are used in Fig. 2.4. The objects are placed along a curved path of motion for better illustration. Notice that the in-between tetrahedron rotates and deforms at the same time, thus avoiding the flipping inside out problem that occurs when the objects are morphed by using the linear vertex path.

Fig. 6.4 shows the interpolation of two human figures facing us. The figure is rotating his body, while at the same time, turning his right arm. Each in-between figure needs about 0.2 seconds to compute. Fig. 6.5 shows the interpolation of the same figure as in Fig. 6.4 but using the linear vertex path. Notice that the in-between human figures are unnaturally thin and the right arm is severely distorted. The most severe distortion occurs at $t = 0.6$ when the right arm disappears completely. Fig. 6.7 shows the comparison of the animation in Fig. 6.4 and 6.5 at $t = 0.6$.

Fig. 6.6 shows another example of morphing. Two objects with opposite "horns" are placed at the rightmost and leftmost positions. The lower sequence, which is

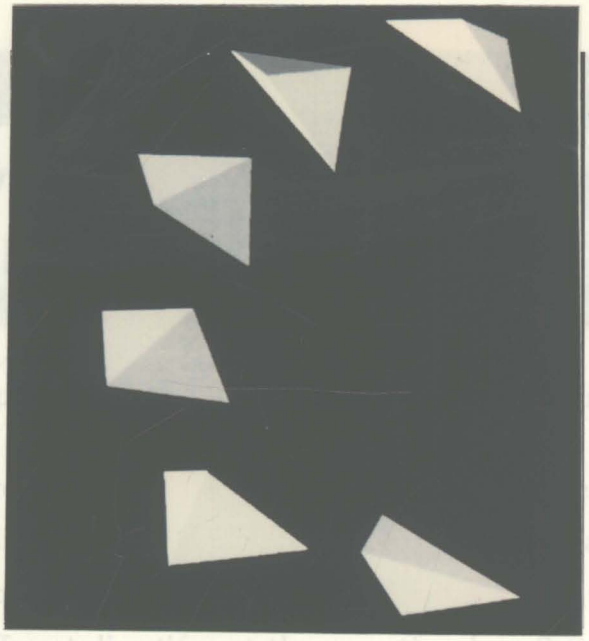

Figure 6.3: *Interpolation between two tetrahedra.*

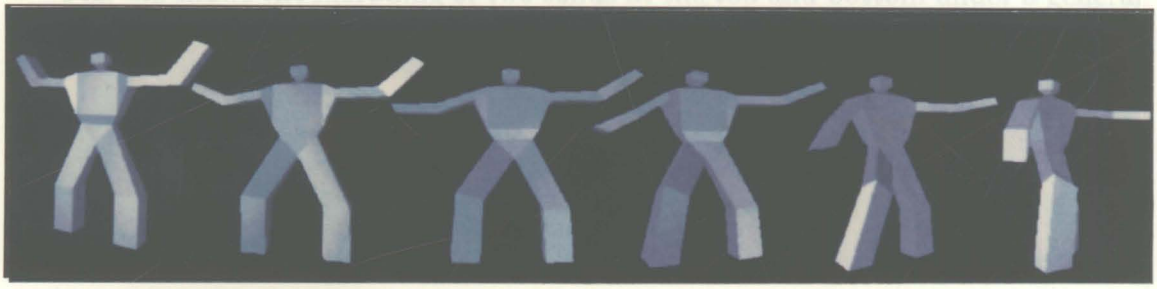

Figure 6.4: *Interpolation between two human figures.*

arranged in a curved path, shows the morphing process using the two-phase intrinsic interpolation, while the upper sequence, which is arranged in a linear path, shows the morphing process using the linear vertex path. The two-phase intrinsic interpolation algorithm avoids the inversion of interior surfaces in the in-between models by rotating and distorting the models at the same time.

As suggested in [27], one is often tempted to think that properly tuning the orientation of one object and simply using the linear vertex path or Minkowski sum can produce a desired in-between model. However, the morphing like that in Fig. 6.4 may involve a lot of prominent features, e.g., the trunk and arms, needed to

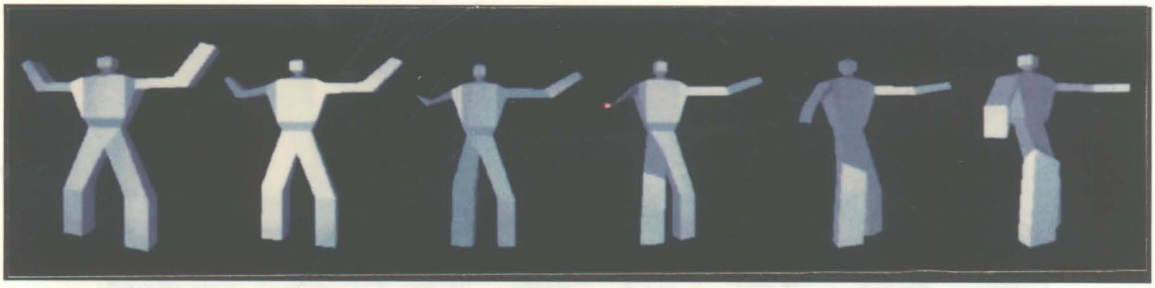

Figure 6.5: *Interpolation between two human figures using linear vertex path.*

turn in different directions at the same time. In this case, the approach of tuning the relative orientation of the input objects can never produce a desired in-between object. Thus, that our intrinsic interpolation algorithm is able to turn the arm and the trunk in two different directions at the same time is a distinct advantage.

Fig. 6.8 shows the morphing of two solids at the top and bottom under a general correspondence with the linear interpolation shown on the left and the intrinsic interpolation on the right. The approximated normal is shown as the white arrow. The flipping inside out in the linear vertex paths at $t = 0.2$ and $t = 0.4$ is avoided by the intrinsic interpolation. Fig. 6.9 and 6.10 with similar layout to Fig. 6.8 show how the degenerated normal is approximated at a convex edge and a concave edge, respectively. Shrinkage and flip-around are avoided in the interpolation and the computation is done nearly instantaneously.

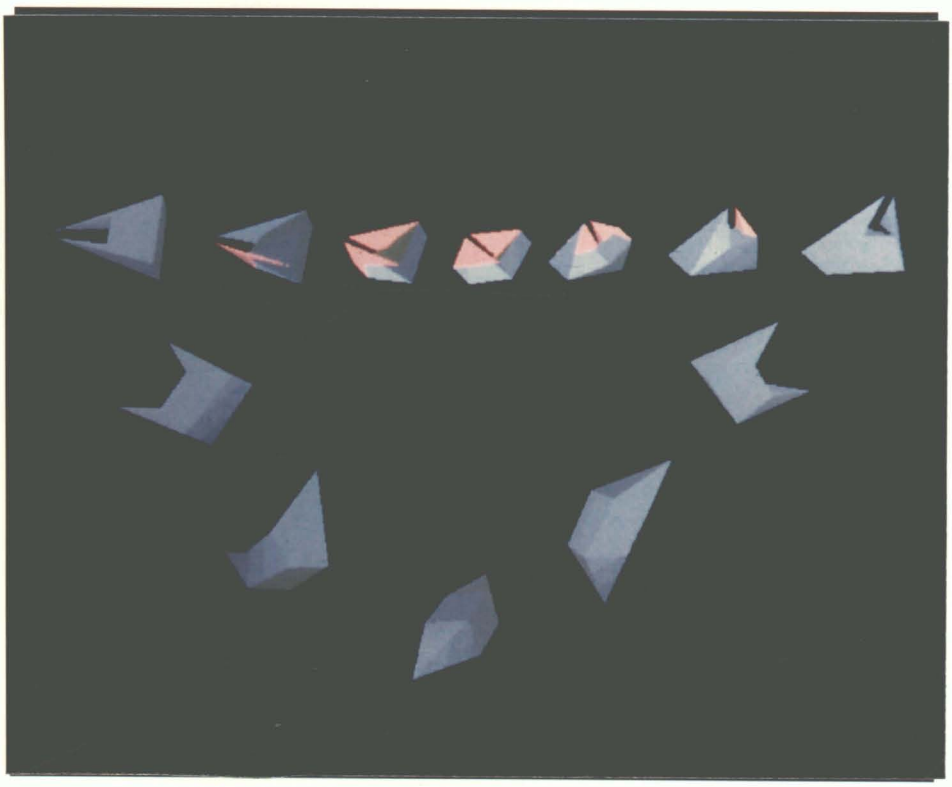

Figure 6.6: *Morphing between two general polyhedral models*

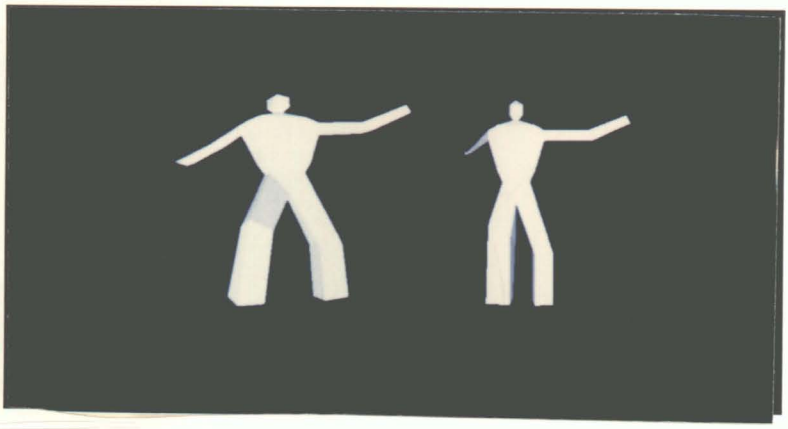

Figure 6.7: *Comparison of the in-between models by our method and the linear vertex path.*

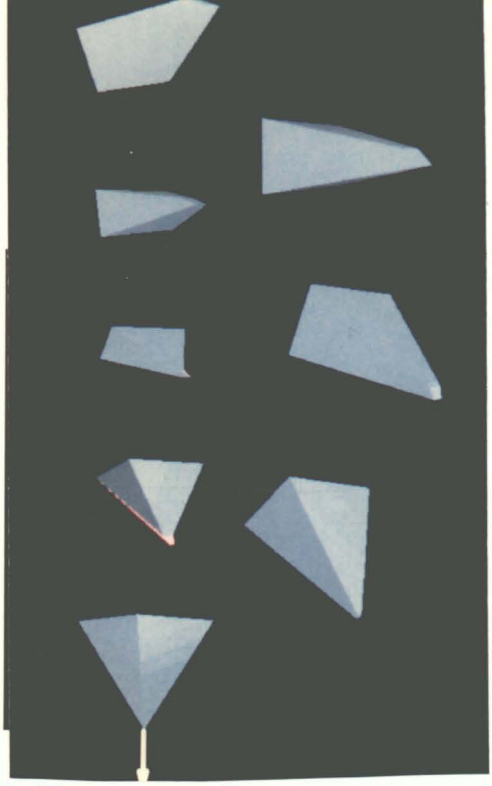

Figure 6.8: *Morphing under a general correspondence.*

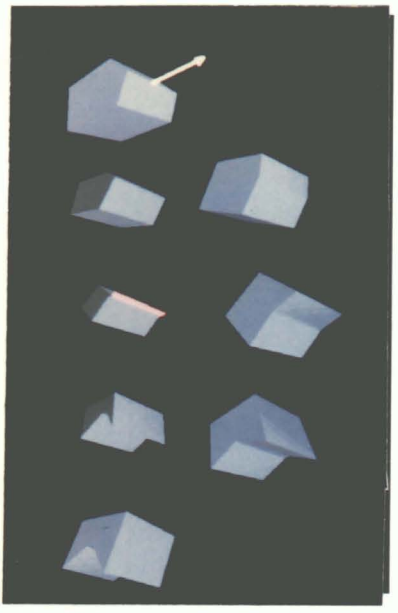

Figure 6.9: Approximating normal at a convex edge.

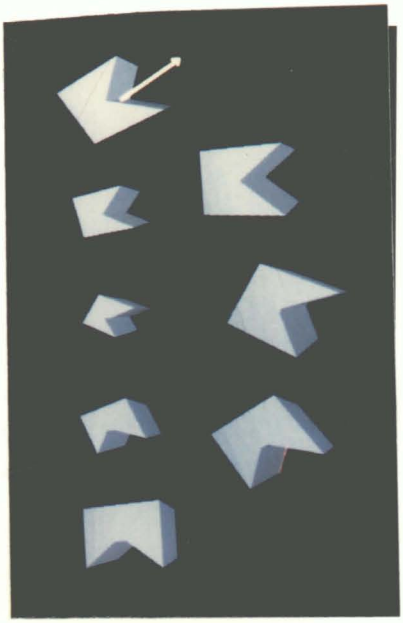

Figure 6.10: Approximating normal at a concave edge.

Chapter 7

Establishment of Correspondence

This chapter discusses how to apply intrinsic parameters to solving the correspondence problem by deriving shape metrics that are based on intrinsic parameters.

7.1 Shape Metric

7.1.1 Deforming Polyhedral Models

For simplicity, we first assume that A and B are topologically equivalent polyhedra and their faces, edges, and vertices are numbered according to the one-to-one correspondence between A and B. When deforming a polyhedral model, we could deform its faces by size, shape, position, and orientation. The four cases above do not exclude each other and often happen simultaneously. Fig. 7.1 shows the deformation of a face by size. Fig. 7.2 shows that the shape of a face is deformed. In Fig. 7.3, the relative orientations of two adjacent faces are changed. In Fig. 7.4, we slide one face by a translation. We can define work required to make the transformations in the four cases above. Deforming face f_i^A of area ω_i^A to face f_i^B of area ω_i^B requires

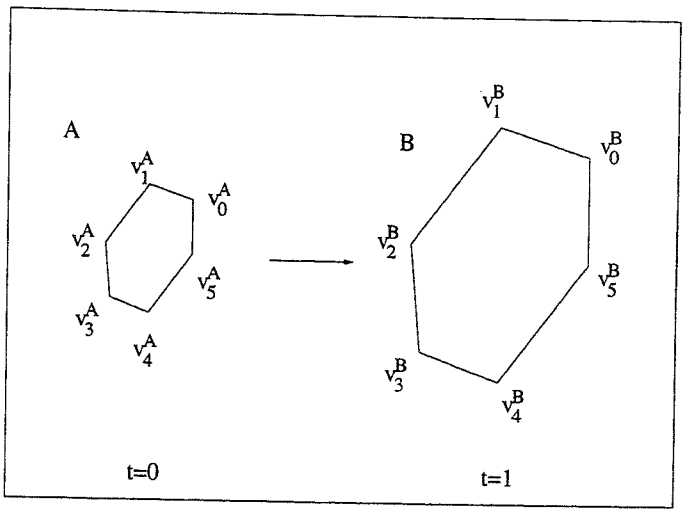

Figure 7.1: *A face changes in size.*

the work

$$W_\omega(i) = k_\omega|\omega_i^A - \omega_i^B| \tag{7.1}$$

where ω_i^A and ω_i^B are areas of the i^{th} faces of A and B, respectively, and k_ω represents the stiffness to stretch an area. As in the physically-based shape blending for 2-D polygons, the work to transform the shapes of the faces can be defined in terms of the bending work and the stretching work. Let $\tilde{\theta}_{c,b,a}$ be the interior angle $\angle V_a V_b V_c$. The work to bend interior angle $\tilde{\theta}^A_{c,b,a}$ of value $\theta_A(i,j)$ to $\tilde{\theta}^B_{c,b,a}$ of value $\theta_B(i,j)$ is defined as

$$W_\zeta([c,b,a]) = k_b|\tilde{\theta}^A_{c,b,a} - \tilde{\theta}^B_{c,b,a}|$$

where k_b is a user-defined constant representing the stiffness to bending. The work to stretch the i^{th} edge from length $\tilde{l}_i^A$ to $\tilde{l}_i^B$ is defined as

$$W_S(i) = k_s|\tilde{l}_i^A - \tilde{l}_i^B| \tag{7.2}$$

where k_s is a user-defined constant representing the stiffness to stretch an edge, and $\tilde{l}_i^A$ and $\tilde{l}_i^B$ are the lengths of the i^{th} edges of A and B respectively. Similarly, bending adjacent faces f_a^A and f_b^A forming dihedral angle $l^A_{a,b}$ to adjacent faces f_a^B and f_b^B forming dihedral angle $l^B_{a,b}$ requires the work

$$W_D([a,b]) = k_d|l^A_{a,b} - l^B_{a,b}| \tag{7.3}$$

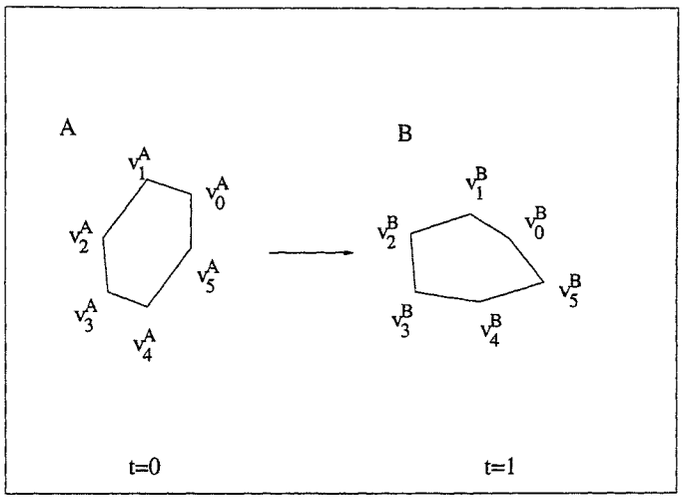

Figure 7.2: *A face changes in shape.*

for some constant k_d which controls the stiffness to fold or unfold two adjacent faces. Sliding one face involves a change in the perpendicular distance from the center of gravity of the polyhedral model to the sliding face. The deformation work done for transforming face f_i^A of perpendicular distance μ_i^A to face f_i^B of perpendicular distance μ_i^B to the center of gravity is defined by

$$W_P(i) = k_p|\mu_i^A - \mu_i^B| \tag{7.4}$$

for some constant k_p which is related to the resistance to inflate or deflate the object by changing the relative position of a face.

7.1.2 Metric Based on Connected EGI

(a) Definition

We develop a shape metric based on the connected EGI for polyhedral models. Let $W_{EGI}^{\varphi}(A, B)$ be the shape metric between A and B under correspondence φ. The connected EGI is composed of the face adjacency graph, which contains the unit outward face normals of the polyhedron, and the areas of the faces. For simplicity, we define $W_\zeta([a, b, c]) = 0$ when v_a^A, v_b^A, and v_c^A do not proceed around a face of A

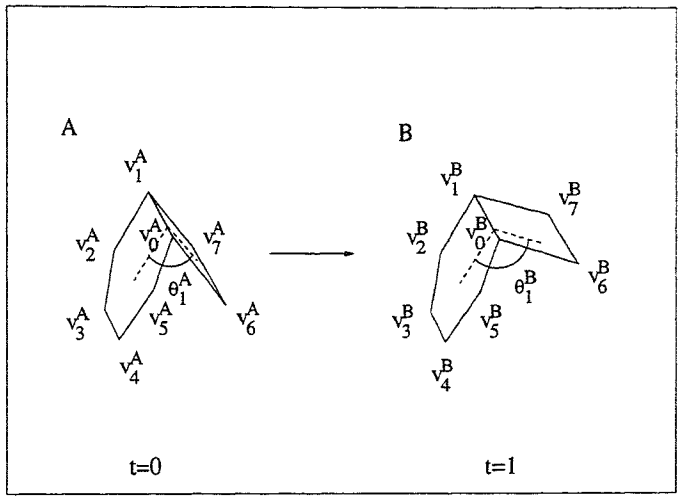

Figure 7.3: *Change of relative orientations of two adjacent faces.*

counterclockwise and neither do v_a^B, v_b^B, and v_c^B, and $W_D([i,j],[i,j]) = 0$ when f_i^A and f_j^A, and f_i^B and f_j^B are not adjacent. Then $W_{EGI}^{\varphi}(A,B)$ is defined by

$$W_{EGI}^{\varphi}(A,B) = \sum_{i=0}^{\overline{n}-1} W_{\omega}(i) + \sum_{a=0}^{\overline{n}-1}\sum_{b=0}^{\overline{n}-1}\sum_{c=0}^{\overline{n}-1} W_{\zeta}([a,b,c]) + \sum_{i=0}^{\overline{n}-1}\sum_{j=i}^{\overline{n}-1} W_D([i,j])$$

The shape metric $W_{EGI}^{\varphi}(A,B)$ above is defined only for two topologically equivalent objects. Topologically equivalent to $VAG^{M(t)}$, the two vertex adjacency graphs VAG^{M_A} and VAG^{M_B} that are defined by φ, A, and B are used to replace VAG^A and VAG^B. We then approximate the undefined interior angles, outward face normals, dihedral angles, outface normals, edge tangents, and outface angles by the same method of interpolating polyhedral models under general correspondence in Section 6.5. In this case, we define

$$W_{EGI}^{\varphi}(A,B) = W_{EGI}^{\varphi'}(M^A, M^B) \tag{7.5}$$

where M^A and M^B are the polyhedral models represented by VAG^{M_A} and VAG^{M_B} respectively and φ' is the one-to-one correspondence between VAG^{M_A} and VAG^{M_B}.

The parameters k_ω, k_b, and k_d are user-controllable and are used to adjust the relative contributions of the different types of work done.

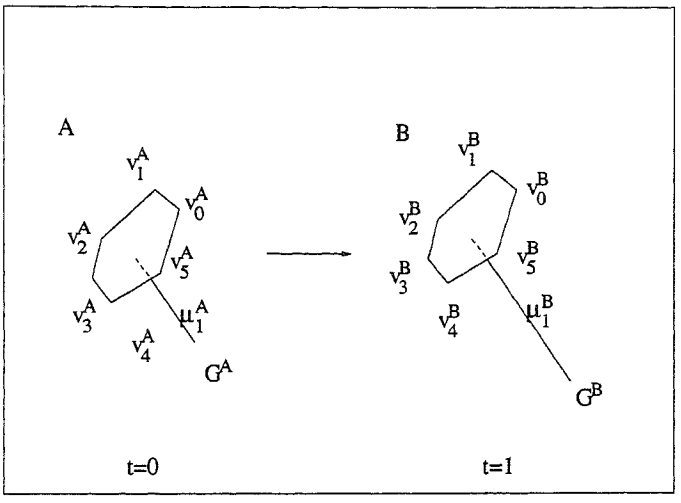

Figure 7.4: *Change of relative positions of a face.*

(b) Properties

Property 7.1 *For any* $\theta, \beta \in [0, 2\pi]$, *and vectors* $\mathbf{a}, \mathbf{b}$, $W^{\varphi}_{EGI}(A^{\theta,\mathbf{a}}, B^{\beta,\mathbf{b}}) = W^{\varphi}_{EGI}(A, B)$

Property 7.2 *For any vectors* $\mathbf{x}, \mathbf{y}$, $W^{\varphi}_{EGI}(A_{\mathbf{x}}, B_{\mathbf{y}}) = W^{\varphi}_{EGI}(A, B)$

Property 7.3 *When* k_ω, k_b, *and* k_d *are not equal to zero, and* A *and* B *have convex faces only,* $W^{C}_{EGI}(A, B) = 0$ *if and only if* A *and* B *are congruent.*

Proof: When A and B are congruent, $W^{\varphi}_{EGI}(A, B)$ is obviously equal to zero.

If $W^{\varphi}_{EGI}(A, B) = 0$, and k_ω, k_b, and k_d are not equal to zero, then

$$\begin{aligned} W^{\varphi}_{\omega}(A, B) &= 0 \\ W^{\varphi}_{\zeta}(A, B) &= 0, \text{ and} \\ W^{\varphi}_{D}(A, B) &= 0 \end{aligned}$$

$W^{\varphi}_{\omega}(A, B) = 0$ means that all the corresponding faces have the same area; $W^{\varphi}_{D}(A, B) = 0$ means that all the corresponding dihedral angles are the same; and

$W_P^{\varphi}(A, B) = 0$ means that all the corresponding interior angles are the same. Suppose that the connected EGI of A cannot coincide with that of B by applying any rotation transformation to A and B. For simplicity, we assume that VAG^A and VAG^B are topologically equivalent and all their boundary elements, such as faces and edges, are named in one-to-one corresponding fashion. We align the arc N_0^A, N_1^A with the arc N_0^B, N_1^B such that they coincide on the Gaussian sphere. Suppose a connected subgraph of VAG^A coincides with a corresponding connected subgraph of VAG^B of s nodes, $s < \overline{n}$. Let f_c^A, f_b^A, and f_a^A be three consecutive nodes around a region of the FAG^A such that f_b^A and f_c^A belong to the subgraph while f_a^A does not. f_c^B, f_b^B, and f_a^B are similarly defined. Then the relation among N_c^A, N_b^A, and N_a^A, and that among N_c^B, N_b^B, and N_a^B are described by Eq. (5.2) and Eq. (5.2) [1]:

$$e_{b,a} = \mathbf{R}_{N_b}(\theta_{c,b,a})e_{b,c} \tag{7.6}$$

$$N_a = \mathbf{R}_{N_b \times e_{b,a}}(l_{b,a})N_b \tag{7.7}$$

Owing to the hypothesis that the two subgraphs coincide on the Guassian sphere, $e_{b,c}^A = e_{b,c}^B$, $N_b^A = N_b^B$, and $\theta_{c,b,a}^A = \theta_{c,b,a}^B$. Thus $e_{b,a}^A = e_{b,a}^B$. And since $l_{b,a}^A = l_{b,a}^B$, $N_a^A = N_a^B$. Consequently, with f_a^A and f_a^B added, the two subgraphs coincide on the Gaussian sphere. By mathematical induction, the connected EGI's of A and B are the same up to rotation.

Thus, by the Uniqueness Theorem 4.1, A and B are congruent. *Q.E.D.*

By selecting different coefficients k_ω, k_b, k_d, we can make either kind of deformation to have a smaller or greater work. Therefore, we can obtain different correspondences, which may result in morphing more involved in a certain kind of deformation.

[1] As a reminder, $e_{b,a}$ is the outface normal parallel to face f_b and points towards the exterior of f_b; $\theta_{c,b,a}$ is the outface angle from $e_{b,c}$ into $e_{b,a}$ about N_b; $l_{b,a}$ is the dihedral angle between the faces f_b and f_a. See page 38 for details.

7.1.3 Metric Based on Connected EDI

(a) Definition

A shape metric based on the connected EDI for polyhedral models is also developed. Let $W^{\varphi}_{EDI}(A,B)$ be the shape metric between A and B under correspondence φ. The connected EDI is composed of the face adjacency graph, which contains the unit outward face normals of the polyhedron, and the perpendicular distances from the centers of gravity to the faces. $W^{\varphi}_{EDI}(A,B)$ is defined by

$$W^{\varphi}_{EDI}(A,B) = \sum_{i=0}^{\overline{n}-1} W_P(i) + \sum_{a=0}^{\overline{n}-1}\sum_{b=0}^{\overline{n}-1}\sum_{c=0}^{\overline{n}-1} W_\varsigma([a,b,c]) + \sum_{i=0}^{\overline{n}-1}\sum_{j=i}^{\overline{n}-1} W_D([i,j])$$

When VAG^A and VAG^B are not topologically equivalent, we define

$$W^{\varphi}_{EDI}(A,B) = W^{\varphi'}_{EDI}(M^A, M^B) \tag{7.8}$$

where M^A, M^B, φ' are defined similarly in the definition of $W^{\varphi}_{EGI}(A,B)$. Similarly, the parameters k_p, k_b, and k_d are user-controllable.

(b) Properties

Property 7.4 *For any* $\theta, \beta \in [0, 2\pi]$, *and vectors* $\mathbf{a}, \mathbf{b}$, $W^{\varphi}_{EDI}(A^{\theta,\mathbf{a}}, B^{\beta,\mathbf{b}}) = W^{\varphi}_{EDI}(A,B)$

Property 7.5 *For any vectors* $\mathbf{x}, \mathbf{y}$, $W^{\varphi}_{EDI}(A_{\mathbf{x}}, B_{\mathbf{y}}) = W^{\varphi}_{EDI}(A,B)$

Property 7.6 *When* k_ω, k_b, *and* k_d *are not equal to zero,* $W^{C}_{EDI}(A,B) = 0$ *if and only if* A *and* B *are congruent.*

Proof: When A and B are congruent, $W^{\varphi}_{EDI}(A,B)$ is obviously equal to zero.

If $W^{\varphi}_{EDI}(A,B) = 0$, and k_p, k_b, and k_d are not equal to zero, then

$$W^{\varphi}_{P}(A,B) \;=\; 0 \tag{7.9}$$

$$W_\zeta^\varphi(A,B) = 0 \text{ and} \tag{7.10}$$

$$W_D^\varphi(A,B) = 0 \tag{7.11}$$

$W_P^\varphi(A,B) = 0$ means that all the corresponding faces have the same perpendicular distance from the centers of gravity of A and B respectively. $W_D^\varphi(A,B) = 0$ means that all the corresponding dihedral angles are the same; and $W_\zeta^\varphi(A,B) = 0$ means that all the corresponding interior angles are the same. By the same argument in the proof of Property 7.3, the connected EDI of A are the same as that of B up to rotation. Thus, by the Uniqueness Theorem 4.3, A and B are congruent. *Q.E.D.*

Similarly, by selecting different coefficients k_p, k_b, k_d, we obtain different correspondences.

7.2 The Searching Algorithm

The problem of establishing a correspondence involves searching for all possible correspondences between two given input polyhedral models such that the correspondence is valid and the the work done metric is minimized. One possible way to do so is to carry out parallel breadth-first search on the the vertex adjacency graphs of A and B as in the super-object approach [5]. Unfortunately, it is still unknown how to do the searching efficiently; and the search for all possible correspondences has not been implemented.

Chapter 8

Conclusions and Future Research

While the criteria for morphing can be artistic, this thesis proposes the basic criteria that a morphing algorithm should satisfy – identity preserving, rotation invariant, translation invariant, and feature preserving, and introduces the intrinsic parameters into metamorphosis algorithms. This work also applies the the intrinsic shape parameters by presenting the two-phase intrinsic interpolation algorithm to 3-D morphing of polyhedral models. The algorithm is based on two graph-based representations of polyhedral objects – the vertex adjacency graphs, and the face adjacency graphs – and the intrinsic shape parameters to describe the interrelation between nodes in the graphs. The two-phase interpolation algorithm is translation and rotation invariant, and is identity preserving if the correspondence is the identity correspondence. It has also been demonstrated in practice that the two-phase intrinsic interpolation algorithm preserves the features common to both input objects and avoids the shrinkage that usually occurs in morphing two identical objects with different orientations by the linear vertex path.

Valid general correspondences for polyhedral models is formulated. Based on the proposed criteria and the intrinsic representation for the polyhedral models, shape difference metrics for the correspondence problem are also proposed.

8.1 Future Research

The interpolation of FAG or VAG in the two-phase interpolation algorithm is one-pass and depends on the order of computation, or the order of searching uncomputed nodes. To eliminate the computation order dependence, methods based on global optimization may be worth investigating. There is no guarantee that the in-between object produced by the two-phase intrinsic interpolation algorithm does not have self-intersection. Up to now, no known published work on the interpolation problem has effectively solved this problem. This still remains an open problem for future research in metamorphosis.

Making an exhaustive search to solve the correspondence problem is an impractical solution. The future research includes finding an efficient searching algorithm for the minimization of shape difference in the correspondence problem.

Bibliography

[1] Harold Abelson and Andrea A. diSessa. *Turtle Geometry*, The MIT Press, 1980.

[2] Barzel R, Barr A. A modeling system based on dynamic constraints. *ACM Computer Graphics SIGGRAPH*, 22(4), 1988.

[3] Thaddeus Beir and Shawn Neely. Feature-Based Image Metamorphosis. *Computer Graphics (Proc. SIGGRAPH '92)*, pp. 35-42, 1992.

[4] Bryan P. Bergeron, Luke Stao and Ronald L. Rouse. Morphing as a Means of Generating Variation in Visual Medical Teaching Materials. *Comput. Biol. Med.*, 24(1), pp.11-18, 1994.

[5] E. Wesley Bethel and Samuel P. Uselton. Shape Distortion in Computer-Assisted Keyframe Animation. *State of the Art in Computer Animation.* Magnenat-Thalmann, N. and Thalmann, D., eds., Springer-Verlag, New York, 1989, pp.215-224.

[6] David T. Chen, Andrei State, and David Banks. Interactive Shape Metamorphosis, *1995 Symposium on Interactive 3D Graphics*, pp. 43-44, 1995.

[7] Shenchang Eric Chen and Richard Parent. Shape Averaging and its Applications to Industrial Design. *IEEE CG&A*. 9(1): 47-54, 1989.

[8] X. Chen and P. Lienhardt. Modelling and Programming Evolutions of Surfaces. *Computer Graphics Forum*, 11(5), pp.323-341, 1992.

[9] Shenchang Eric Chen and Lance Williams. View Interpolation for Image Synthesis. *Computer Graphics (Proc. SIGGRAPH)*, pp. 279-288, 1993.

[10] Herve Delingette, Yasuhiko Watanabe, and Yasuhito Suenaga. Simplex Based Animation. *Models and Techniques in Computer Animation*, Nadia Magnenat Thalmann, Daniel Thalmann (Eds), 1993, Springer Verlag, pp. 13-18.

[11] Herbert Edelsbrunner. *Weighted Alpha Shapes.* Technical Report #1760, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, 1992.

[12] Herbert Edelsbrunner and Ping Fu. Geometric Morphing with Alpha Shapes. *Preprint.*

[13] Ekoule, A., Parent, F., and Oet, C. A Triangulation Algorithm from Arbitrary Shaped Multiple Planar Contours, *ACM Transactions on Graphics*, (10)2, pp.182-192, April 1991.

[14] M.A. Eshera and K.S. Fu. A Graph Distance Measure for Image Analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-14, No. 3, May/June, pp. 398-408, 1984.

[15] T.J. Fan, G. Medion, and R. Nevatia. Recognizing 3-D Objects Using Surface Descriptions. *Second International Conference on Computer Vision*, pp. 474-480, 1988.

[16] T.J. Fan. *Describing and Recognizing 3D Objects Using Surface Properties.* New York, Springer-Verlag, 1990.

[17] P.M. Gruber and J. Wills, EDS. *Handbook of Convex Geometry, volumes A and B.* North-Holland, Amsterdam, 1993.

[18] Leonida Guibas and John Hershberger. Morphing Simple Polygons. *Preprint*

[19] Frank Harray. *Graph Theory*, Addision-Wesley, 1969.

[20] Robert M. Haralick, Stanley R. Sternberg, and Xinhua Zhuang. Image Analysis Using Mathematical Morphology. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, July, 1987.

[21] Tong Minh Hong, N. Magnenat-Thalmann and D. Thalmann. A General Algorithm for 3-D Shape Interpolation in a Facet-Based Representation. *Proceedings of Graphics Interface '88*, pp. 229-235.

[22] Hong, Jiawei and Tan, Xiaonan. Recognize the Similarity Between Shapes under Affine Transformation, *Second International Conference on Computer Vision*, pp. 489-492, 1988.

[23] B.K.P. Horn. Extended Gaussian Images. *Proceedings of IEEE*, vol.72, no.12, pp.1671-1686, Dec. 1984.

[24] John F. Huges. Scheduled Fourier Volume Morphing, *Computer Graphics (Proc. SIGGRAPH '92)*, 26(2): 43-46, 1992.

[25] K. Ikeuchi. Recognition of 3-D Objects Using the Extended Gaussian Image. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, 24-28 August, 1981, UBC, Vancouver, B.C Canada, pp. 595-600.

[26] S.B. Kang and K. Ikeuchi. The Complex EGI: A New Representation for 3-D Pose Determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 7, July, pp. 707-720, 1993.

[27] Anil Kaul and Jarek Rossignac. Solid-Interpolating Deformations: Construction and animation of PIPs. In F.H. Post and W. Barth, editors, *Proc. Eurographics '91*, pp. 493-505. Elsevier Science Publishers B.V, 1991.

[28] James R. Kent, Richard E. Parent, and Wayne E. Carlson. Establishing Correspondences by Topological Merging: A New Approach to 3-D Shape Transformation. *Proceedings of Graphics Interface '91*, Calgary, Alberta, June 1991, pp. 271-278.

[29] James R. Kent, Wayne E. Carlson, and Richard E. Parent. Shape Transformation for Polyhedral Objects. *Computer Graphics (Proc. SIGGRAPH '92)*, 26(2): 47-54, 1992.

[30] Asish Law and Roni Yangel. *Voxel-Based Morphing*, Work in Progress Report, Department of Computer and Information Science, The Ohio State University, 1994.

[31] C. W. Lee. Regular Triangulations of Convex Polytopes, *Applied Geometry and Discrete Mathematics: the Victor Klee Festschrift*, ed. P. Gritzmann and B. Sturmfels, ACM and AMS, 1991.

[32] Apostolos Lerios, Chase D. Garfinkle, and Marc Levoy. Feature-Based Volume Metamorphosis, to be appeared in *Computer Graphics (Proc. SIGGRAPH '95)*, 1995.

[33] James J. Little. An Iterative Method for Reconstructing Convex Polyhedra from Extended Gaussian Polyhedra. *Proc. Nat. Conf. on AI*, Washington, D.C., Aug, 1983, pp. 247-254.

[34] L. A. Liusternik. *Convex Figures and Polyhedra.* Dover, 1963.

[35] M. Mortenson. *Geometric Modeling*, J. Wiley and Sons, New York, 1985.

[36] Shinji Mukai, Susumu Furukawa, Makoto Obi and Fumihiko Kimura. An Algorithm for Deciding Similarities of Convex Polyhedra. *Comput. & Graphics* , vol. 18, No. 2, pp. 171-176, 1994.

[37] Alan Norton, Greg Turk, Bob Bacon, John Gerth, and Paula Sweeney. Animation of Fracture by Physical Modeling, *The Visual Computer* (7), pp. 210-219, 1991.

[38] Reeves W. In-betweening for Computer Animation Utilizing Moving Point Constraints, *Proc. SIGGRAPH '81*, pp. 33-41.

[39] A. Sanfliu and K.S. Fu. A Distance Measure Between Attributed Relational Graphs for Pattern Recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No. 3, May/June, pp. 353-361, 1983.

[40] Thomas W. Sederberg and Eugene Greenwood. A Physically Based Approach to 2-D Shape Blending. *Computer Graphics (Proc. SIGGRAPH '92)*, 26(2): 47-54, 1992.

[41] Thomas W. Sederberg, Peisheng Gao, Guojin Wang, and Hong Mu. 2-D Shape Blending: An Intrinsic Solution to the Vertex Path Problem. *Computer Graphics (Proc. SIGGRAPH '93)*, pp. 15-18, 1993.

[42] L.G. Shapiro and R.M. Haralick. Structural Descriptions and Inexact Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-3, No.5, Sep., pp. 504-518, 1981.

[43] Geoffrey Slinker. *Inbetweening Using a Physically Based Model and Nonlinear Path Interpolation*, Master's thesis, Brigham Young University, Department of Computer Science, 1992.

[44] Yue Man Sun, Wenping Wang, and Francis Y.L. Chin. Interpolating Polyhedral Models using Intrinsic Shape Parameters, *Pacific Graphics* '95, to be published.

[45] Burtnyk N. and Wein M. Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation, *Comm. ACM*, Vol. 19, No. 10, pp. 564-569.

[46] Louis Weinberg. A Simple and Efficient Algorithm for Determining Isomorphism of Planar Triply Connected Graphs, *IEEE Transactions on Circuit Theory*, 13(2), pp.142-148, 1966.

[47] G. Wolberg. *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, CA, 1990.

[48] A.K.C. Wong and M. You. Entropy and Distance of Random Graphs with Application to Structural Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 5, September, pp. 599-608, 1985.

[49] A.K.C. Wong, Si W. Lu and M. Rioux. Recognition and Shape Synthesis of 3-D Objects Based on Attributed Hypergraphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 3, March, pp. 279-289, 1989.