3D Morphing Using Strain Field Interpolation*

Han-Bing Yan[1] (严寒冰), Shi-Min Hu[2]** (胡事民), and Ralph R Martin[3]

[1]*National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China*

[2]*Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*

[3]*School of Computer Science, Cardiff University, Cardiff, U.K.*

E-mail: yanhb02@mails.tsinghua.edu.cn; shimin@tsinghua.edu.cn; ralph@cs.cf.ac.uk

Received February 20, 2006; revised October 16, 2006.

Abstract In this paper, we present a new technique based on strain fields to carry out 3D shape morphing for applications in computer graphics and related areas. Strain is an important geometric quantity used in mechanics to describe the deformation of objects. We apply it in a novel way to analyze and control deformation in morphing. Using position vector fields, the strain field relating source and target shapes can be obtained. By interpolating this strain field between zero and a final desired value we can obtain the position field for intermediate shapes. This method ensures that the 3D morphing process is smooth. Locally, volumes suffer minimal distortion, and no shape jittering or wobbling happens: other methods do not necessarily have these desirable properties. We also show how to control the method so that changes of shape (in particular, size changes) vary linearly with time.

Keywords morphing, strain field, interpolation

1 Introduction

Morphing, or metamorphosis, aims to generate a smooth shape sequence which transforms a source shape into a target shape. This technique has become increasingly important in computer graphics for animation and entertainment, and is commonly employed by the special effects industry. Many morphing techniques have been developed for the 2D cases[1]. With the introduction to 3D games and cartoons, 3D morphing has increased in importance.

Methods for 3D morphing typically take one of two approaches. The first blends simple volumes in which the initial and final shapes are embedded[2−4]; the initial and final shapes may be embedded in a higher-dimensional volume[5]. The other approach is based on manipulating meshes, which may be 3D surface meshes or volumetric meshes. The first approach has the advantage of being able to deal objects with different topologies. However, in most cases, the mesh method exhibits better results — often, shape boundaries produced by the volume based method are not smooth enough. The method in this paper is based on meshes.

Usually, mesh morphing techniques involve two steps. The first is to find a mapping between source and target objects, which requires that both should be meshed in an equivalent way — they should be meshed *isomorphically* with *consistent* meshes, i.e., having a one-to-one correspondence. Having done this, the second step is to choose a suitable path for each vertex from its position in the original mesh to its position in the final mesh, while keeping the connectivity of the mesh the same. The simplest (but generally unsuitable) method for finding vertex paths is to use linear interpolation.

Two kinds of meshes may be used in 3D morphing, surface (triangle) meshes[6,7] and volumetric (tetrahedron) meshes[8]. Surface meshes have fewer elements and so methods based on them generally take less computational time. Solid meshes fill the interior of the shape instead of covering its boundary, and have the advantage that methods based on them can more easily avoid volume shrinkage, and generally result in less distortion.

In this paper, we propose a novel physically-based morphing approach for 3D meshes, which is an extension of a 2D physically based morphing method we reported earlier[9]. Note that the 3D case is much more complex than the 2D case; it also has more potential applications. We focus in this paper on the particular issue of finding paths for the vertices of consistent meshes—finding consistent meshes can be done using existing methods[8].

The main contribution of our work is the use of an appropriate mathematical tool to quantitatively describe shape deformation in 3D morphing. This tool is *strain*, which has been used in mechanics for hundreds of years. Our new morphing method is based on this mathematical tool, and as a result can theoretically guarantee that the morphing process is smooth and uniform, no matter how large the difference between the source and target shapes. The deformation occurring at each point of the shape uniformly changes with time. Our results verify that our method does not have the displeasing visual effects that often arise when using linear interpolation, and indeed many other existing methods, such as *squeezing*, *shrinking* and *local self-intersection*.

Regular Paper

*This work was done while Han-Bing Yan was a Ph.D. candidate in Tsinghua University.

**Corresponding Author.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 60673004 and 60333010), the National Grand Fundamental Research 973 Program of China (Grant No. 2006CB303106) and the Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList).

2 Related Work

Mesh morphing is an active research area in computer graphics. Some work[10−13] considers how to create consistent meshes for pairs of shapes of genus zero using a topological merging method. Much work[6,14−17] is based on dissecting the source and target shape into several pieces, and constructing a local parameterization for each piece, then using merging or remeshing methods to create a consistent mesh. Praun[18] gives a tracing method which can dissect the source and target shapes automatically.

The above papers focus on the problem of creating consistent meshes, and most of them then use simple linear interpolation methods to find the vertex paths during the morphing process. For objects having very similar shapes, linear interpolation is good enough for simple visual effects. However, for objects undergoing large deformation, especially when bending occurs, linear interpolation always leads to shrinkage of intermediate shapes, which is visually unacceptable. In [19, 20], Floater and Surazhsky proposed the use of barycentric coordinate and mean value interpolation to find suitable paths. Blanding *et al.*[21] represented the interiors of 3D shapes using compatible skeletons and applied blending to parametric descriptions of the skeletons.

Alexa[22] suggested using interpolation of Laplacian coordinates for the morphing path and discussed how to control morphing locally. Such differential methods have also been used to study deformation, a similar problem to morphing[23,24]. Laplacian coordinates are invariant under translation but are not invariant to rotation and scaling, so the interpolated Laplacian coordinates need to be modified to obtain good morphing results[25]. Sheffer and Kraevoy[26] introduced so called *pyramid coordinates* into mesh editing and morphing. Pyramid coordinates are rotation-invariant, although their use requires the solution of a non-linear optimization problem. Lipman[27] proposed rotation invariant differential coordinates by defining tangential and normal components of the surface, and used them for morphing; this approach needs to solve two linear equations for each intermediate frame. Xu *et al.*[28] calculate intermediate surface gradients by quaternion interpolation, then reconstruct surfaces by solving a Poisson equation. This method produces good results in many cases even when differences between initial and final shapes are large.

Hu[29] *et al.* presented a method based on minimization of deformation energy, which is novel in that it does not use interpolation. However, it is a global optimization method. Bao *et al.*[30] used a physically based method for morphing, like the present paper, but based on point sampled geometry while we use meshes.

A method which *can* handle source and target shapes with large differences is presented in [8]. This method first converts both surface polygon meshes into a tetrahedral mesh. To perform morphing, it then finds a transformation which is locally as similar as possible to the optimal transformation between each pair of corresponding tetrahedra. Optimization is used to minimize the difference between the desired transformation, and the actual transformation which is applied, taking into account the connectivity constraints on adjacent tetrahedra. This paper also considers how to create consistent tetrahedral meshes using a topological merging method. Sumner[31] used similar simplex transformation ideas to learn deformations from examples.

An alternative approach is proposed in [32, 33]. In these methods, consistent meshes are not required. They dynamically and adaptively change the connectivity of intermediate meshes, gradually transforming the connectivity from that of the source model to that of the target. It would seem difficult for this method to get good results for shapes that differ greatly.

By using concepts of positive and negative surfaces, Lee[34] gave a method to parameterize non-zero genus surfaces and applied it to morphing models of different genus. Liu and Che also studied morphing of different genus objects[35,36].

For further discussions of previous work, the reader is referred to two excellent 3D morphing surveys by Alexa, and Lazarus[37,38].

3 Framework of 3D Strain Field Morphing

The purpose of morphing is to create a smooth deformation process. The path of each vertex in linear interpolation is smooth, so why does not it produce a visually acceptable result in many cases? The reason is that used by itself, it does not ensure that *shape change* during this process is smooth. A smooth deformation requires not only that each vertex path should be smooth, but also that *shape* should change in a *monotonic* manner.

It would seem that previous 3D morphing methods have not used an effective means of analyzing and controlling shape deformation. As we know, deformation is a local infinitesimal quantity. In some cases, points may have very large displacement but small deformation. For example, strain should remain at zero under rigid body motion. In other cases, there can be large deformation at a point but small displacement. For example, if we stretch a bar by pulling its left and right ends outwards with equal force, the center is not displaced, but there is still a large deformation at the center (as there is at all other points of the bar). All previous methods, such as barycentric coordinates, Laplacian coordinates, or surface gradients, cannot satisfy simultaneously the requirements for describing shape deformation and displacement. Thus, previous methods do not provide the correct theoretical basis to ensure the morphing process is uniform as desired.

In fact, an ideal quantity has been used in mechanics to describe shape deformation for hundreds of years: *strain*. In this paper, we use the concept of *strain* to describe shape deformation for morphing, and use it as the basis of a new approach to 3D morphing.

Here, to apply the concepts of strain, we use a solid tetrahedral mesh to represent shape. The inputs to our method are a source mesh and a target mesh, which are *consistent* tetrahedral meshes. Consistent tetrahedral meshes can be created using existing methods[8]. The output is a sequence of intermediate meshes forming a morphing sequence. The main steps of our method are as follows:

- compute the strain field between the source shape and target shape;
- interpolate strain to determine the strain field for each intermediate shape;
- calculate the intermediate shapes from the intermediate strain fields.

In Section 4, we outline the basic concepts necessary. Then we explain our 3D strain field interpolation morphing method in detail in Section 5. Results and future work are discussed in Sections 6 and 7.

4 Preliminaries

Our aim is to make shape deformation during 3D morphing seem natural and visually appealing. We want the deformation to occur in a uniform manner. But what is deformation? Note carefully that local deformation at a point is not the same as the displacement of this point. Points in an object can have very small deformation although they have moved a long distance, or conversely, they can have large deformation but small displacement. The key idea is that *deformation* concerns how much a point is displaced *relative to neighboring points*. Thus, deformation is an infinitesimal quantity. To analyze deformation, we use a tool from mechanics, *strain*. We start by giving definitions of the *position field* and the *strain field* associated with a shape.

4.1 Definition of 3D Strain Fields

The positions of all points in a shape comprise a field, which we call *position field*. In 3D space, the position field has three components: $[x, y, z]$. When a shape moves or deforms, each point on the shape ends up in a new place $[x', y', z']$.

Using ideas from mechanics, the difference between a source shape and a target, or any intermediate shape, can be decomposed into two parts: a *rigid body motion*, and a *deformation*. The rigid body motion can be further decomposed into a translation and rotation. In 3D, there are six degrees of freedom for rigid body motion: three for translation and three for rotation. The deformation is separately captured by a *strain field*, which is independent of rigid body motion. If only rigid body motion happens, without deformation, although each point has displacement, the strain field is zero.

In morphing, we wish to handle large deformations, whereas in mechanics and mechanical engineering the deformations are often small. Thus, here we must use large deformation formulae instead of the small deformation (approximate, linearised) formulae most often used in mechanical engineering. In 3D, the strain field is a second-order tensor field, with 6 independent components (for homogeneous materials): ϵ_x, ϵ_y, ϵ_z are tension components, and γ_{yz}, γ_{zx}, γ_{xy} are shear components. In detail, ϵ_x ,ϵ_y, ϵ_z measure in relative terms how much local expansion or shrinkage at each point along each coordinate direction. Positive values represent expansion and negative values represent shrinkage. γ_{xy} represents the relative change in angle between lines initially in the x and y directions at a given point, and similarly for γ_{yz} and γ_{zx}. Engineers are often only interested in small deformations, which can be approximately expressed by a linear relationship between strain and position, for example: $\epsilon_x = \partial x'/\partial x$, etc. However, in this work, we need to deal with large deformations, so we must use large deformation strain field, or Lagrange strain field, formulae. The relationship between the Lagrange strain field and the position field in 3D is given by:

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left[\left(\frac{\partial x'}{\partial x}\right)^2 + \left(\frac{\partial y'}{\partial x}\right)^2 + \left(\frac{\partial z'}{\partial x}\right)^2 - 1\right] \\ \frac{1}{2}\left[\left(\frac{\partial x'}{\partial y}\right)^2 + \left(\frac{\partial y'}{\partial y}\right)^2 + \left(\frac{\partial z'}{\partial y}\right)^2 - 1\right] \\ \frac{1}{2}\left[\left(\frac{\partial x'}{\partial z}\right)^2 + \left(\frac{\partial y'}{\partial z}\right)^2 + \left(\frac{\partial z'}{\partial z}\right)^2 - 1\right] \\ \frac{\partial x'}{\partial y}\frac{\partial x'}{\partial z} + \frac{\partial y'}{\partial y}\frac{\partial y'}{\partial z} + \frac{\partial z'}{\partial y}\frac{\partial z'}{\partial z} \\ \frac{\partial x'}{\partial z}\frac{\partial x'}{\partial x} + \frac{\partial y'}{\partial z}\frac{\partial y'}{\partial x} + \frac{\partial z'}{\partial z}\frac{\partial z'}{\partial x} \\ \frac{\partial x'}{\partial x}\frac{\partial x'}{\partial y} + \frac{\partial y'}{\partial x}\frac{\partial y'}{\partial y} + \frac{\partial z'}{\partial x}\frac{\partial z'}{\partial y} \end{bmatrix} \quad (1)$$

where x, y and z are the position field components before deformation, and x', y' and z' are components of the position field after deformation. The *displacement field* has components u, v, w, and is related to the position field by $u = x' - x$, $v = y' - y$, $w = z' - z$.

In the following, the strain field is represented in vector form, and we write $\boldsymbol{\varepsilon}$ for $[\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{yz}, \gamma_{zx}, \gamma_{xy}]^{\mathrm{T}}$. We assume the shape varies from source to target over the time interval $t \in [0, 1]$ and we use a superscript to denote time. For example, $\boldsymbol{\varepsilon}^t$ means the strain field of the target shape at time t, so $\boldsymbol{\varepsilon}^1$ means the final strain field.

4.2 Quantitative Analysis of Strain Fields

Figs.1 and 2 are two results produced using linear interpolation, and strain field interpolation respectively. It is clear that the result of strain field interpolation is more natural and uniform: observe the elephant's trunk. The result in Fig.1 is poor because the trunk shrinks before expanding again. Linear mesh interpolation results in significant size changes manifested as shrinkage followed by re-expansion in this case. This is shown in

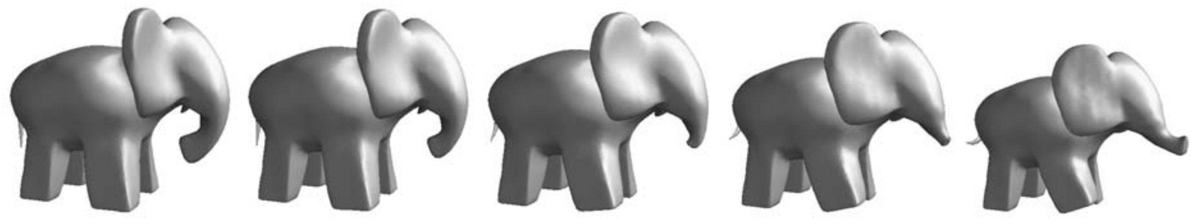


Fig.1. Linear interpolation.

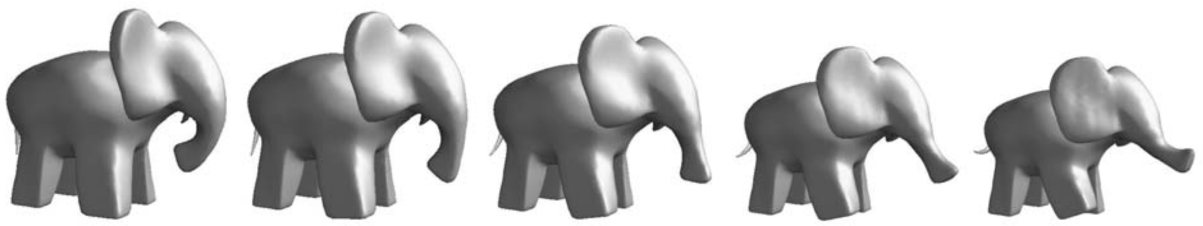
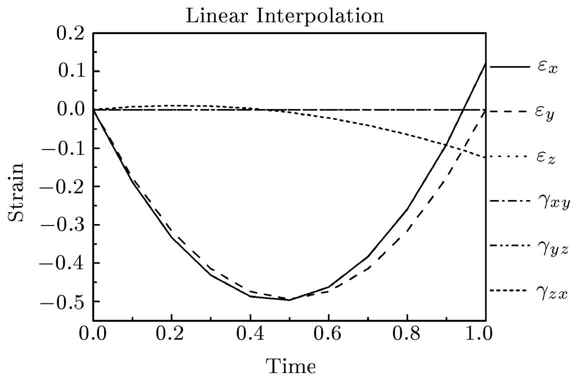


Fig.2. Strain field interpolation.



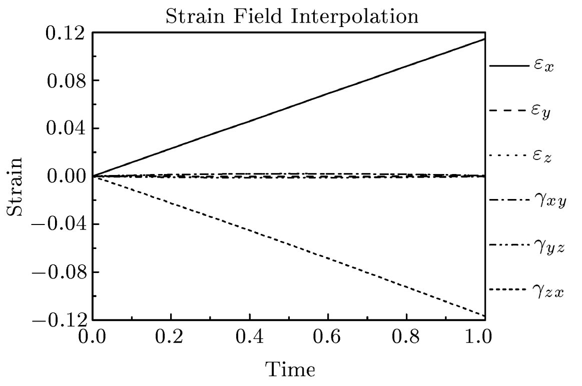


Fig.3. Strain-time curve.

Fig.3, which presents strain curves calculated for a tetrahedron near the tip of the elephant's trunk, when using linear interpolation, and strain field interpolation, respectively. Each figure shows six curves for the different components of strain. In the case of linear interpolation, the curves for ϵ_z, γ_{yz} and γ_{zx} almost overlap; in strain field interpolation, ϵ_y, ϵ_z, γ_{yz} and γ_{zx} are nearly identical to zero. In Fig.1, the tip of the elephant's trunk becomes smaller in the x-y plane at first, then becomes bigger again. The corresponding linear interpolation strain curves show ϵ_x and ϵ_y become negative at first, meaning that the trunk's tip is highly compressed. Then the curves rise again, as the tip restores its size. This effect is visually undesirable. In contrast, in Fig.2, the elephant's trunk deforms uniformly, and the corresponding strain curves change in a monotonic way, leading to a much more visually desirable result.

Considering what happens in the case of linear interpolation shows that strain fields are a powerful quantitative tool to analyze shape deformation. Going further, if we want a shape to deform gradually, the strain field should change monotonically. Otherwise *squeezing* or *wobbling* (oscillations in deformation) may happen, as in Fig.1. Our approach uses strain fields not only to analyze deformation, but to *control* it.

5 Strain Field Morphing

We now describe the steps of our new method of carrying out deformation using strain fields.

5.1 Removing the Rigid Body Motion

Strain is independent of rigid body motion. The overall metamorphosis comprises a deformation plus a rigid body motion, and we firstly separate out the latter so that we can consider the strain by itself.

A rigid body has 6 degrees of freedom in 3D space. To factor out the rigid body motion, we select three corresponding points on source and target shapes, A^0, B^0, C^0, and A^1, B^1, C^1. The points chosen to determine the rigid body motion can be selected by the computer automatically, or interactively by the user. A^0 and A^1 should be selected near the center of the source and target shapes, while B^0, B^1, C^0 and C^1 should lie along the shape's "natural" axes.

We now place the source and target shapes in a canonical position and orientation, using these reference points. The source shape is moved until A^0 coincides with the origin, the translation vector necessary being

$\boldsymbol{T}^0$. Next, the source shape is rotated around the origin until B^0 lies on the x axis. Let the rotation axis be $\boldsymbol{R}^0$, and rotation angle be α^0. Finally, the source shape is rotated around the x axis until C^0 is in the x-y plane with positive y component; the angle of rotation is β^0. The target shape treated is then similarly, with corresponding parameters $\boldsymbol{T}^1$, $\boldsymbol{R}^1$, α^1 and β^1.

The x, y, z coordinates of A, the y, z coordinates of B, and the z coordinate of C are then fixed during the entire deformation process, and the rigid body component of metamorphosis is added back as a final computation after the deformation has been determined, as will be explained in Subsection 5.4.

5.2 Intermediate Strain Field Calculation

The strain field for intermediate shapes between the source and target shapes can be calculated using the definitions in (1). We use tetrahedral elements, as often used in the *finite element method* (FEM), to calculate the strain field. Because, unlike engineering analysis, morphing does not require high precision results, it is sufficient to use linear tetrahedral elements rather than higher order elements. In FEM, the displacement of each point inside an element can be expressed as a convex combination of the displacements of the element's nodes. The vertex coordinates of a tetrahedron in the source mesh are $v_i = (x_i, y_i, z_i)$, $i = 1 \dots 4$; in the target mesh, its coordinates are $v'_i = (x'_i, y'_i, z'_i)$. Any point inside some source or target tetrahedron can be expressed in terms of its vertices and *shape functions*; for example, for a target tetrahedron:

$$\begin{cases} x' = \sum_{k=1}^{4} N_k x'_k, \\ y' = \sum_{k=1}^{4} N_k y'_k, \\ z' = \sum_{k=1}^{4} N_k z'_k. \end{cases} \tag{2}$$

The N_k are the shape functions of the element, and depend on x, y and z. For *linear* tetrahedral elements, the N_k are barycentric coordinates within the tetrahedron which can be expressed in the (linear) form:

$$N_k = a_k + b_k x + c_k y + d_k z, \tag{3}$$

where a_k, b_k, c_k, d_k are functions of the source coordinates x_i, y_i and z_i — for details, see e.g., [39]. Substituting (3) into (2), and thence into (1), we can convert the strain field calculation for a tetrahedron into discrete form. Using the node coordinates for each source and target tetrahedron in (1) and (2), the strain field can then be calculated tetrahedron by tetrahedron.

The simplest way to choose a strain field for intermediate shapes is to *linearly interpolate the strain* (which is not the same as linearly interpolating vertex positions!):

$$\boldsymbol{\varepsilon}^t = (1-t)\boldsymbol{\varepsilon}^0 + t\boldsymbol{\varepsilon}^1; \tag{4}$$

note that $\boldsymbol{\varepsilon}^0$ is zero.

Linear strain interpolation is usually an adequate method, producing good morphing results for most examples and avoiding oscillations in local size over time. However, in some extreme cases, linear strain interpolation gives unequal rates of change of shape over time, even though at each intermediate time the shape itself is good. In Subsection 5.5, we analyze this problem further, and provide an alternative method of interpolation for such extreme cases.

5.3 From Strain Field to Position

The above approach gives the required strain field at any intermediate time. Note that the strain field has six components, while the position field has three components. Clearly, as a result, the six components of the strain field cannot be independent, and in fact they are connected by a set of compatibility conditions[40]. In general, arbitrarily interpolated strain fields will not satisfy these compatibility conditions and hence do not correspond to physically realisable position fields. To resolve this issue, we attempt to find that position field for each intermediate shape whose actual strain field is as close as possible to the interpolated strain field. We define an energy function in (5) which computes the norm of the difference between the interpolated strain field and a physically correct strain field generated by some position field:

$$W = \frac{1}{2}\int (\boldsymbol{\varepsilon}^* - \boldsymbol{\varepsilon})^{\mathrm{T}} \cdot (\boldsymbol{\varepsilon}^* - \boldsymbol{\varepsilon}) \mathrm{d}\Omega. \tag{5}$$

Ω is taken over the whole source shape, ε is the interpolated strain field, and ε^* is calculated from the position field using (1). We find the position field which minimizes this energy function to give the intermediate shape. This problem can be solved by any multivariate optimization method; note that the variables are the coordinates of all nodes in the intermediate shape.

In practice, for efficiency, we convert this optimisation problem into a set of non-linear equations by setting the derivative of (5) to zero. This non-linear equation system can be written as:

$$\phi(\boldsymbol{V}) = 0 \tag{6}$$

where $\boldsymbol{V}$ is the coordinate vector of the intermediate shape. $\phi(\boldsymbol{V})$ has the form below, where the overall function is evaluated by summing a function over each tetrahedron in turn:

$$\phi(\boldsymbol{V}) = \sum_{i=1}^{m} \int (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{\varepsilon}^* - \boldsymbol{B}^{\mathrm{T}}\boldsymbol{\varepsilon}) \mathrm{d}\Omega_i. \tag{7}$$

Here m is the number of tetrahedra, and $\boldsymbol{B}$ and $\boldsymbol{\epsilon}$ are the strain matrix and strain for each separate tetrahedron as appropriate. $\boldsymbol{B}$ can be written as

$$\boldsymbol{B} = [\, \boldsymbol{B}_1 \quad \boldsymbol{B}_2 \quad \boldsymbol{B}_3 \quad \boldsymbol{B}_4 \,] \tag{8}$$

where $\boldsymbol{B}_j = [\,\boldsymbol{B}_{jx} \quad \boldsymbol{B}_{jy} \quad \boldsymbol{B}_{jz}\,]$, and $\boldsymbol{B}_{jX}$ ($X = x, y$, or z) is defined by:

$$\boldsymbol{B}_{jX} = \begin{bmatrix} \dfrac{\partial X^t}{\partial x}\dfrac{\partial N_i}{\partial x} \\ \dfrac{\partial X^t}{\partial y}\dfrac{\partial N_i}{\partial y} \\ \dfrac{\partial X^t}{\partial z}\dfrac{\partial N_i}{\partial z} \\ \dfrac{\partial X^t}{\partial y}\dfrac{\partial N_i}{\partial z} + \dfrac{\partial X^t}{\partial z}\dfrac{\partial N_i}{\partial y} \\ \dfrac{\partial X^t}{\partial z}\dfrac{\partial N_i}{\partial x} + \dfrac{\partial X^t}{\partial x}\dfrac{\partial N_i}{\partial z} \\ \dfrac{\partial X^t}{\partial x}\dfrac{\partial N_i}{\partial y} + \dfrac{\partial X^t}{\partial y}\dfrac{\partial N_i}{\partial x} \end{bmatrix}.$$

This non-linear equation can be solved using the Newton-Raphson method. We generally wish to calculate the position field for a series of intermediate shapes. The coordinate vector for the source shape is used as the initial value for iterative calculation of the coordinate vector for the first intermediate shape, and the calculation for each subsequent intermediate shape is initialized using the coordinate vector of the previous intermediate shape. We use the conjugate gradient method to solve the sparse linear equations in each iterative process. As each step has good initial values, rapid convergence is obtained. If the number of intermediate shapes required is very small, (6) could also be solved by using a continuation method, but in practice, simply adding more intermediate shapes is more efficient.

5.4 Incorporating Rigid Body Motion

Deformations for intermediate shapes were determined by the method in the previous section. Now, the appropriate rigid body motion must also be incorporated to give each final intermediate shape in its correct position and orientation. We simply reverse the process given in Subsection 5.1, adding back a linearly interpolated translation and rotation. First we rotate the intermediate shape at time t around the x axis by the angle:

$$\beta^t = -[(1-t)\beta^0 + t\beta^1]. \tag{9}$$

Then we find the axis of rotation

$$\boldsymbol{R}^t = (1-t)\boldsymbol{R}^0 + t\boldsymbol{R}^1 \tag{10}$$

and rotate around this axis by the angle

$$\alpha^t = -[(1-t)\alpha^0 + t\alpha^1]. \tag{11}$$

Finally, the intermediate shape is translated by

$$\boldsymbol{T}^t = -[(1-t)\boldsymbol{T}^0 + t\boldsymbol{T}^1], \tag{12}$$

giving the desired result.

5.5 Modified Interpolation Method

Using linear interpolation of strain as described in Subsection 5.2 is often adequate to produce good results. However, in a few extreme cases where the deformation is very large, although each intermediate shape has a good shape, and shape change is monotonic, shape change may occur at an uneven rate over time. The reason for this phenomenon is that (1) includes quadratic terms.

In fact, many morphing methods suffer from non-uniform (and often oscillatory) changes over time, but unlike other methods, in the case of strain field morphing we can find a theoretically-based way of compensating for this. The basis of this modification is to force the lengths of edges parallel to the coordinate axes to change linearly with time, which also constrains the length change ratio in other directions. Appropriate background ideas from mechanics can be found in [39, 40].

In essence, the strain field arises due to length changes of line segments. Consider an infinitesimal line segment PQ in the source shape. Let the coordinates of P be (a_x, a_y, a_z), or a for short. Let the coordinates of Q be $(a_x+da_x, a_y+da_y, a_z+\mathrm{d}a_z)$. Let the length of PQ be dS^0. The corresponding line segment in the target shape is P^1Q^1, of length dS^1. The relation between dS^0 and dS^1 is:

$$(dS^1)^2 - (dS^0)^2 = 2\boldsymbol{da}\cdot\boldsymbol{E}^1\cdot\boldsymbol{da}, \tag{13}$$

$$(dS^0)^2 = \boldsymbol{da}\cdot\boldsymbol{da}, \tag{14}$$

where $\boldsymbol{da}$ is the vector (da_x, da_y, da_z), and $\boldsymbol{E}^1$ is the matrix form of the strain field tensor for the target mesh. The general strain field tensor $\boldsymbol{E}$ is of the form:

$$\boldsymbol{E} = \begin{bmatrix} \epsilon_x & \gamma_{xy} & \gamma_{xz} \\ \gamma_{xy} & \epsilon_y & \gamma_{yz} \\ \gamma_{xz} & \gamma_{yz} & \epsilon_z \end{bmatrix}. \tag{15}$$

The length of segment PQ at time t obeys a similar relation:

$$(dS^t)^2 - (dS^0)^2 = 2\boldsymbol{da}\cdot\boldsymbol{E}^t\cdot\boldsymbol{da}. \tag{16}$$

If we want the length of PQ to change linearly with time, we should let:

$$\frac{dS^t - dS^0}{dS^1 - dS^0} = t. \tag{17}$$

Substituting (13) and (16) into (17), we obtain

$$(dS^0)^2 + 2\boldsymbol{da}\boldsymbol{E}^t\boldsymbol{da} = [t\sqrt{(dS^0)^2 + 2\boldsymbol{da}\boldsymbol{E}^1\boldsymbol{da}} + (1-t)dS^0]^2. \tag{18}$$

For a line segment parallel to the x axis, $\boldsymbol{da}$ is:

$$\boldsymbol{da} = [da_x, 0, 0]. \tag{19}$$

Substituting (19) into (18), and using (14), we find that:

$$\epsilon_x^t = \frac{1}{2}[[t\sqrt{1+2\epsilon_x^1} + (1-t)]^2 - 1]. \tag{20}$$

Similarly we may show that

$$\epsilon_y^t = \frac{1}{2}[[t\sqrt{1+2\epsilon_y^1} + (1-t)]^2 - 1] \tag{21}$$

and

$$\epsilon_z^t = \frac{1}{2}[[t\sqrt{1+2\epsilon_z^1} + (1-t)]^2 - 1]. \tag{22}$$

We now use (20) – (22) to interpolate the tension strains, while using still linear interpolation for shear strains γ_{xy}, γ_{yz} and γ_{zx}. Our experiments show that in all large deformation cases, this technique avoids the problem of uneven deformation over time, even for very large deformations, and provides good visual results.

6 Results

We have applied the technique of strain field interpolation to various kinds of models, both morphing between two different objects, and morphing processes corresponding to several kinds of fundamental deformation of a single object. Typical experimental results are shown in Figs. 4 – 6. All experiments were performed on a 3.2GHz Pentium 4 computer. Calculation times to produce 30 frames for the simple models in Figs .4 and 5 were 3.4 and 14.7 seconds; they have 80 and 164 vertices respectively. The more complex model in Fig.6 has about 17,000 vertices; it took about 55 minutes to produce 80 frames.

Our experiments show that the 3D morphing results generated by our method are smooth and natural. No shape *jittering* or *wobbling* occurs. The key is the use of *strain* in our method, which is a powerful tool to describe shape deformation in an infinitesimal way.

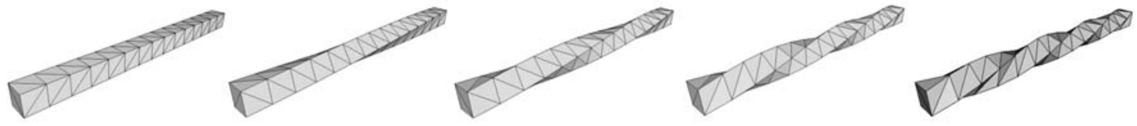


Fig.4. Twisting a rod.

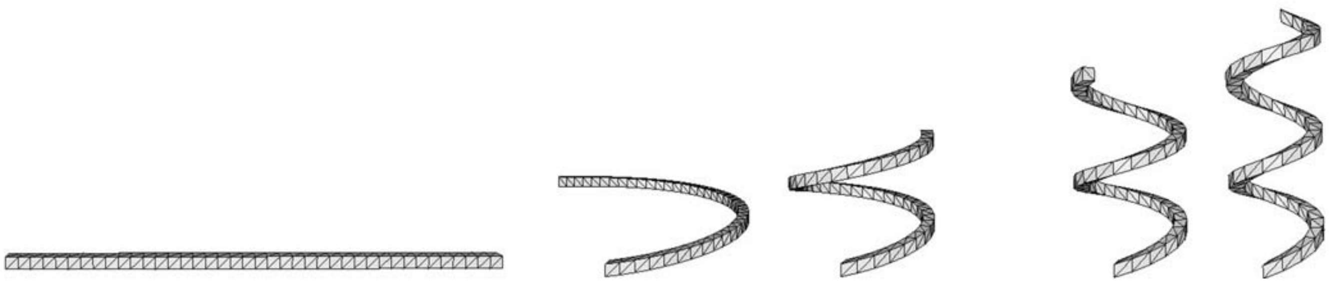


Fig.5. Helix.

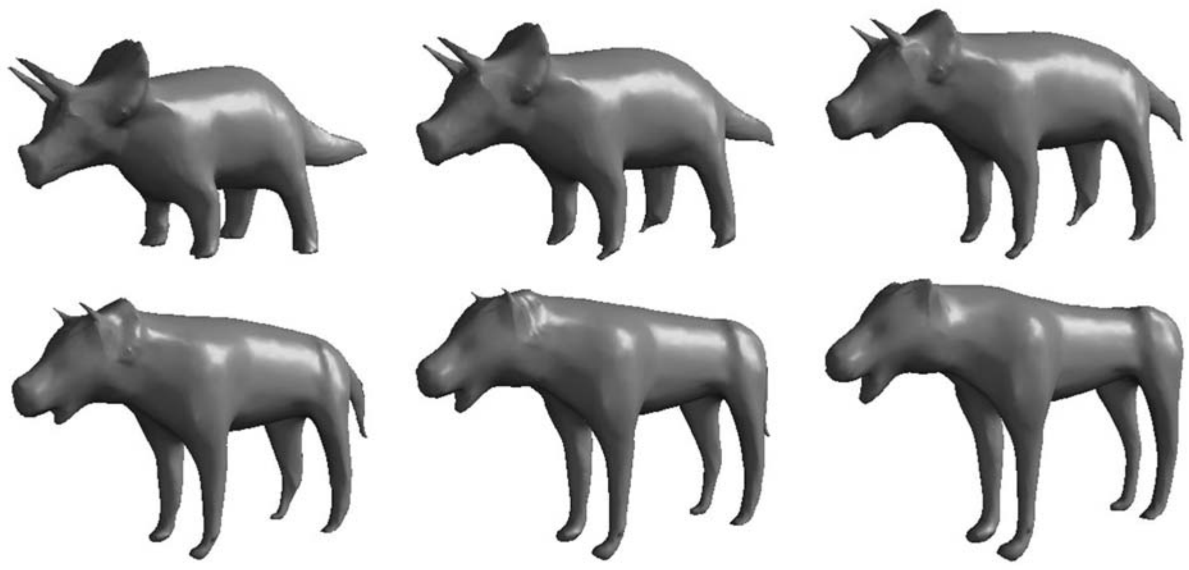


Fig.6. Animal morphing.

7 Discussion and Future Work

In this paper, we have presented a new method for 3D morphing. By making use of the concept of strain from mechanics to control deformation in 3D, we have developed an algorithm for 3D morphing based on strain field interpolation. While linear interpolation of strain is often adequate, in cases of very large deformations, this produces non-uniform rates of deformation over time, and we have further given a modified strain interpolation method which can avoid this problem. Experiments show that our method gives very good results which appear natural.

Although our strain field interpolation method gives good results, it has a high computational cost, but so do other physically based methods using solid meshes. Our future goal is to reduce the calculation cost of our method. We envision two approaches to do this. Firstly, the current method requires all elements of the solid mesh to be processed to compute the deformation. In practice it would be much more efficient if we could use a multiresolution method in which strain field interpolation is used to ensure the gross deformations of the shape are performed in a plausible manner, while using a simpler method such as direct linear interpolation of geometry to control the fine changes in shape. A second possibility is to extend our strain field interpolation method to work directly with 3D surface meshes rather than solid volume meshes, which would also significantly lower the calculational expense.

Acknowledgements We acknowledge Qi-Xing Huang and Qian-Yi Zhou's help in our work. We would also like to thank Bruno Levy for providing the elephant model.

References

[1] George Wolberg. Image morphing: A survey. *The Visual Computer*, 1998, 14(8/9): 360–372.

[2] Daniel Cohen-Or, Amira Solomovici, David Levin. Three dimensional distance field metamorphosis. *ACM Trans. Graphics*, 1998, 17(2): 116–141.

[3] Apostolos Lerios, Chase D Garfinkle, Marc Levoy. Feature based volume metamorphosis. *Computer Graphics*, 1995, 29: 449–456.

[4] Xiang Fang, Hujun Bao, Pheng-Ann Heng, Tien-Tsin Wong, Qunsheng Peng. Continuous field based free-form surface modeling and morphing. *Computer and Graphics*, 2001, 25(2): 235–243.

[5] Greg Turk, James F O'Brien. Shape transformation using variational implicit functions. In *Proc. ACM SIGGRAPH 1999*, Los Angeles, USA, 1999, pp. 335–342.

[6] Takashi Kanai, Hiromasa Suzuki, Fumihiko Kimura. Three-dimensional geometric metamorphosis based on harmonic maps. *The Visual Computer*, 1998, 14(4): 166–176.

[7] Aaron W F Lee, David Dobkin, Wim Sweldens, Peter Schroder. Multiresolution mesh morphing. In *Proc. SIGGRAPH*, Los Angeles, USA, 1999, pp.343–350.

[8] Marc Alexa, Daniel Cohen-Or, David Levin. As-rigid as-possible shape interpolation. In *Proc. SIGGRAPH*, New Orleans, USA, 2000, pp.157–164,

[9] Han-Bing Yan, Shi-Min Hu, Ralph Martin. Morphing based on strain field interpolation. *Computer Animation and Virtual Worlds*, 2004, 15(3-4): 443–452

[10] Marc Alexa. Merging polyhedral shapes with scattered features. *The Visual Computer*, 2000, 16(1): 26–37.

[11] James R Kent, Wayne E Carlson, Richard E Parent. Shape transformation for polyhedral objects. *Computer Graphics*, 1992, 26(2): 47–54.

[12] Francis Lazarus, Anne Verroust. Metamorphosis of cylinder-like objects. *Journal of Visualization and Computer Animation*, 1997, 8(3): 131–146.

[13] Avner Shapiro, Ayellet Tal. Polyhedron realization for shape transformation. *The Visual Computer*, 1998, 14(8/9): 429–444.

[14] Hujun Bao, Qunsheng Peng. Interactive 3d morphing. *Computer Graphics Forum*, 1998, 17(3): 23–30.

[15] Arthur Gregory, Andrei State, Ming C Lin, Dinesh Manocha, Mark A Livingston. Feature-based surface decomposition for correspondence. In *Proc. Computer Animation*, Philadelphia, USA, 1998, pp.64–71.

[16] Tong-Yee Lee, Po-Hua Huang. Fast and intuitive metamorphosis of 3d polyhedral models using smcc mesh merging scheme. *IEEE Trans. Visualization Comput. Graphics*, 2003, 9(1): 85–98.

[17] Jin-Bey Yu, Jung-Hong Chuang. Consistent mesh parameterizations and its application in mesh morphing. In *Proc. Computer Graphics Workshop*, Hualian, 2003, http:/cggmwww.csie.nctu.edu.tw/~jbyu/thesis/CMP_CGW.pdf.

[18] Emil Praun, Wim Sweldens, Peter Schroder. Consistent mesh parameterizations. In *Proc. SIGGRAPH*, Los Angeles, California, USA, 2001, pp.179–184.

[19] Michael S Floater, Craig Gotsman. How to morph tilings injectively. *Journal of Computational and Applied Mathematics*, 1999, 101(1-2): 117–129.

[20] Vitaly Surazhsky, Craig Gotsman. Intrinsic morphing of compatible triangulations. In *Proc. 4th Bi-National Israel-Korea Conference on Geometric Modeling and Computer Graphics*, Tel Aviv, Israel, 2004, pp.45–50.

[21] Rob Blanding, George Turkiyyah, Duane Storti, Mark Ganter. Skeleton based three-dimensional geometric morphing. *Journal of Computational Geometry: Theory and Applications*, 2000, 15(1-3): 129–148.

[22] Marc Alexa. Local control for mesh morphing. In *Proceedings of the International Conference on Shape Modeling and Applications*, Genoa, Italy, 2001, pp.209–215.

[23] Marc Alexa. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 2003, 19(2): 105–114.

[24] Yizhou Yu, Kun Zhou, Dong Xu *et al.* Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graphics* (*Proc. SIGGRAPH'2004*), 2004, 23(3): 644–651.

[25] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman. Laplacian surface editing. In *Proc. the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Nice, France, 2004, pp.179–188.

[26] Alla Sheffer, Vladislav Kraevoy. Pyramid coordinates for morphing and deformation. In *Proc. The 2nd Int. Symp. 3D Data Processing, Visualization and Transmission*, Thessaloniki, Greece, 2004, pp.68–75.

[27] Yaron Lipman, Olga Sorkine, David Levin, Daniel Cohen-Or. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graphics*, 2005, 24(3): 479–487.

[28] Dong Xu, Hongxin Zhang, Qing Wang, Hujun Bao. Poisson shape interpolation. In *Proc. ACM Symposium on Solid and Physical Modeling*, Vienna, Austria, 2005, pp.267–274.

[29] Shi-Min Hu, Chen-Feng Li, Hui Zhang. Actual morphing: A physical-based approach for blending. In *Proc. ACM Symp. Solid Modeling and Application*, Nice, France, 2004, pp.309–314.

[30] Yunfan Bao, Xiaohu Guo, Hong Qin. Physically based morphing of point-sampled surfaces. *Computers Animation Virtual Worlds*, 2005, 16: 509–518.

[31] Sumner R-W, Popovic J. Deformation transfer for triangle meshes. *ACM Trans. Graphics* (*Proc. SIGGRAPH 2004*), 2004, 23(3): 399–405.

[32] Tong-Yee Lee, Chien-Chi Huang. Dynamic and adaptive morphing of three-dimensional mesh using control maps. *IEICE Trans. Information and Systems*, 2005, 88(3): 646–651.

[33] Chao-Hung Lin, Tong-Yee Lee. Metamorphosis of 3d polyhedral models using progressive connectivity transformations. *IEEE Trans. Visualization Computer Graphics*, 2005, 10(6): 2–12.

[34] Tong-Yee Lee, Chih-Yuan Yao, Hung-Kuo Chu *et al.* Generating genus-n-to-m mesh morphing using spherical parameterization. *Computer Animation and Virtual Worlds*, 2006, 17: 433–443.

[35] Li-Gang Liu, Guo-Jin Wang. Three-dimensional shape blending: Intrinsic solutions to spatial interpolation problems. *Computers & Graphics*, 1999, 23(4): 535–545.

[36] Wu-Jun Che, Shi-Min Hu, Ralph R Martin. Skeleton-driven 2d mesh morphing with controllable topological transition. In *Proc. 17th Int. Conf. Computer Animation and Social Agents*, Geneva, Switzerland, 2004, pp.147–154.

[37] Marc Alexa. Recent advances in mesh morphing. *Computer Graphics Forum*, 2002, 21(2): 173–197.

[38] Francis Lazarus, Anne Verroust. Three-dimensional metamorphosis: A survey. *The Visual Computer*, 1998, 14(8/9): 373–389.

[39] Zienkiewicz O C. The Finite Element Method. McGraw-Hill, 2000.

[40] Flugge Wilhelm. Tensor Analysis and Continuum Mechanics. Springer-Verlag, 1972.

Han-Bing Yan obtained his Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, China in 2006. He is now working in the National Computer Network Emergency Response Technical Team/Coordination Center of China. His research interests include computer graphics, computer animation, computer network security and information security.

Shi-Min Hu obtained his Ph.D. degree in 1996 from Zhejiang University. He is currently a professor of computer science at Tsinghua University. His research interests include digital geometry processing, video-based rendering, rendering, computer animation, and computer-aided geometric design. He is on the editorial board of Computer Aided Design.

Ralph R Martin obtained his Ph.D. degree in 1983 from Cambridge University and is now professor at Cardiff University. He has published over 140 papers and 9 books covering such topics as solid and surface modeling, intelligent sketch input, geometric reasoning, reverse engineering, and various aspects of computer graphics. He is on the editorial boards of Computer Aided Design and the International Journal of Shape Modelling.