# **Poisson Shape Interpolation**

Dong Xu Hongxin Zhang Qing Wang Hujun Bao \* State Key Lab of CAD&CG, Zhejiang University, P.R. China

## Abstract

In this paper, we propose a novel shape interpolation approach based on Poisson equation. We formulate the trajectory problem of shape interpolation as solving Poisson equations defined on a domain mesh. A non-linear gradient field interpolation method is proposed to take both vertex coordinates and surface orientation into account. With proper boundary conditions, the in-between shapes are reconstructed implicitly from the interpolated gradient fields, while traditional methods usually manipulate vertex coordinates directly. Besides of global shape interpolation, our method is also applicable to local shape interpolation. Our approach can generate visual pleasing and physical plausible morphing sequences with stable area and volume changes. Experimental results demonstrate that our technique can avoid the shrinkage problem appeared in linear shape interpolation.

**CR Categories:** I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—object representations

**Keywords:** Shape Interpolation, Trajectory Problem, Domain Mesh, Poisson Equation, Gradient Field Manipulation

## 1 Introduction

Shape interpolation, also known as shape blending or morphing, has been widely applied to various aspects of computer graphics industry. Given two input models, shape interpolation can generate a sequence of intermediate shapes which gradually change from the source shape to the target one. It is one of the basic tools to enhance visual effects in computer animation. Shape interpolation can also be applied in industrial design to forecast new product shapes from knowns [Chen and Parent 1989].

It is well-understood that there are two major issues in B-rep (boundary representation) based shape interpolation [Alexa 2002; Lazarus and Verroust 1998]. The first is how to find a feature preserving correspondence map between the given models, known as the *correspondence* problem. The second is how to interpolate the positions for each pair of corresponding vertices along predetermined paths, known as the *trajectory* problem. Most of previous B-rep based algorithms (e.g. [Kanai et al. 2000; Kraevoy and Sheffer 2004; Praun et al. 2001]) are mainly concerned with the first issue while simply adopting linear interpolation to generate vertex trajectories. However, as illustrated in Figure 1, the linear shape interpolation may suffer the *shrinkage* problem even though the correspondence map has been correctly established. The reason is that the large scale rotations, as non-linear factors, can not be correctly expressed by linear interpolation. Furthermore, undesired local wrinkles may appear due to the inconsistent displacement of neighboring vertices.

In this paper, we focus on the trajectory problem of 3D mesh morphing. We assume the correspondence map has been established using existing methods [Kraevoy and Sheffer 2004; Praun et al. 2001; Schreiner et al. 2004]. Our key observation is that the uniform change of surface orientation plays an important role in the quality control of morphing process. Therefore, local orientation should be explicitly considered when shape interpolation methods are designed. Although surface normal is a local differential property, which can be easily determined, it is nontrivial to generate optimal global shapes with surface normal constraints.

The Poisson equation possesses an elegant characteristic to bridge among local differential properties and global effects. Recently, it has been successfully applied in computer graphics applications [Peréz et al. 2003; Stam 1999]. Yu et al. [2004] propose a mesh editing technique with Poisson-based gradient field manipulation. Inspired by their work, input models in our shape interpolation approach are treated as different scalar fields defined on a common domain mesh. The gradient field manipulation is then applied to control desired vertex positions and surface orientations implicitly in the morphing process.

This paper makes two major contributions. First, we formulate the trajectory problem as gradient field interpolation, and propose a novel shape interpolation approach based on the Poisson equation. The in-between shapes are reconstructed from the interpolated gradient fields. As a global optimizer, the Poisson equation can effectively attenuate the gradient field inconsistency introduced by local interpolation, which avoids the appearance of local wrinkles. Second, we propose a non-linear gradient field interpolation algorithm based on polar decomposition, which explicitly considers local surface orientations and stretch.

We show the effectiveness of our method by several aesthetic and physical plausible morphing sequences, which outperform the results generated by linear shape interpolation. Moreover, we demonstrate the application of our method to local control of shape morphing. Incorporating with deformation, our shape interpolation method provides additional user controls, yielding versatile visual effects.

#### 1.1 Related work

The consideration of trajectory problem can date back to the work of Sederberg and Greenwood [1992], which provides a physical based approach of polyline morphing. In that work, the deformation of in-between shapes are decomposed into stretch and bend. Based on this observation, their method interpolates edge length and angles between adjacent edges rather than vertex location. Later, Sederberg et al. [1993] propose a geometric algorithm with global optimization to ensure these blended polylines are close without local self-intersection. A generalization for 3D meshes is given by [Liu and Wang 1999]. However, the final morphing results are

<sup>\*</sup>e-mail: { xudong,zhx,qwang,bao }@cad.zju.edu.cn, corresponding author: Hujun Bao.



Figure 1: Shape interpolation comparisons. The left most column shows the two input models. The top row lists the resultant images of linear interpolation, and the bottom row shows our results.

dependent of the computation order of dihedral angels and edge length.

As a well-known fact, it is uneasy to handle large scale rotations in B-rep based morphing. Many practical tools usually require heavy user interactions to achieve this goal. In literature, Lee et al. [1999] provide a multi-resolution based framework that decomposes the shape information into different level of details for interpolation. However, it needs additional cost to build hierarchical structures and complex correspondence maps. Alternatively, Hu et al. [2004] propose an automatic approach to compute the vertex pathes by minimizing the strain energy with a complex non-linear optimization process.

To address this problem, several researchers consider the interior information of given shapes and propose methods to control local volumes distortions. Shapira and Rappoport [1995] propose a 2D morphing method based on star-skeleton representation. By using the polar decomposition [Shoemake and Duff 1992], Alexa et al. [2000] introduce a generic shape interpolation approach between two compatible 2D triangulations or 3D tetrahedralizations. It requires additional computing cost for interior constraints and it is nontrivial to obtain satisfied compatible tetrahedralizations of given meshes. Recent work of Surazhsky and Gotsman [2003] interpolates the mean value barycentric coordinates for 2D morphing.

Alternatively, local shape representation is a good choice to control local orientation and stretch for shape interpolation. Sheffer and Kraevoy [2004] propose a local representation, called pyramid coordinates, which is invariant under rigid transformation. Based on it, they present a morphing method by linear interpolating the components of pyramid coordinates and achieve comparable results to ours. However, the reconstruction process from pyramid coordinates to vertex coordinates is time-consuming, since this method requies to solve a non-linear system iteratively.

Recently, differential mesh representations, such as Laplacian coordinates [Lipman et al. 2004; Sorkine et al. 2004] and the Poisson equation based method [Yu et al. 2004], are successfully applied to mesh editing. These PDE based methods share the common aspect that target edited models are reconstructed from least-squares minimization of modified differential properties. Since differential properties are sensitive to non-translational transformation, [Lipman et al. 2004; Sorkine et al. 2004; Yu et al. 2004] propose different approaches to translate user interaction into local transforms to obtain desired differential properties. In [Yu et al. 2004], local transforms is specified by user to control uniform scale and rotation. [Lipman et al. 2004] estimates rotations of the local frames from the underlying smooth surface, but does not consider scale. [Sorkine et al. 2004] implicitly fits an optimal local transform for each vertex from the local coordinates of control points. Based on properly-defined local frame, we uniquely determine a local transform for each triangle pair, including rotation and anisotropic scale. In [Alexa 2003], Laplacian coordinates are applied to local mesh morphing which indicates the possibility to use differential mesh representations to solve the trajectory problem. However, [Alexa 2003] proposes to interpolate Laplacian coordinates linearly while we interpolate gradient fields via non-linear interpolation algorithm (See section 4.2 for detailed comparison).

To deal with models of different topological structure, volumebased approaches are proposed by several authors. Typical work include distance field-based morphing [Cohen-Or et al. 1998], dimension lifting [Turk and O'Brien 1999] and the level-set based approach [Breen and Whitaker 2001]. These methods are stable and have strong theoretical background from image processing. The volume-based approaches work well for (nearly) closed shapes, and do not need to explicitly construct topological compatible structures. However, sharp features are not easily retained due to the sampling or quantization issues.

## 2 Framework of Poisson Shape Interpolation

Our proposed Poisson shape interpolation approach is a B-rep based one. It is outlined as follows:

- 1. Compute correspondence map and generate compatible meshes from two input 3D meshes.
- 2. For each corresponding triangle pair of compatible meshes, determine the local transform from source triangle to target one; and decompose the transform into rigid rotation and stretch part.
- 3. Given time *t*, compute interpolated gradient fields by local transforms interpolation; and reconstruct the intermediate shape by Poisson equation solver.

In the rest of this section, we present the framework in details.

#### 2.1 Preliminaries

The *Poisson equation* is a second-order partial differential equation. Recall that  $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})^{\top}$  is the gradient operator in  $\mathbb{R}^3$ . Then we denote  $\Delta = \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$  as the Laplacian operator, and  $\nabla \cdot \mathbf{w} = \frac{\partial w_x}{\partial x} + \frac{\partial w_y}{\partial y} + \frac{\partial w_z}{\partial z}$  is the divergence over a vector field  $\mathbf{w} = (w_x, w_y, w_z) \in \mathbb{R}^3$ . The Poisson equation with Direchlet boundary conditions is formulated as

$$\Delta f = \nabla \cdot \mathbf{w}, \ f \mid_{\partial \Omega} = f^* \mid_{\partial \Omega} , \tag{1}$$

where f is an unknown scalar function,  $f^*$  provides the desirable value on the boundary  $\partial \Omega$ . From the viewpoint of variational method, the Poisson equation is equivalent to the following least-squares minimization:

$$\underset{f}{\operatorname{argmin}} \int_{\Omega} (\nabla f - \mathbf{w})^2 d\Omega.$$
 (2)

It is not straightforward for theoretical analysis and practical computing to discretize Poisson equation defined on an irregular domain, such as on a 3D triangular mesh. Polthier and Preuss [2002], and Tong et al. [2003] proposed well-defined discrete differential operators of scalar and vector fields on irregular domains. Based on their results, the discrete Poisson equation on triangular meshes is formulated as follows.

**Domain mesh.** The input of our algorithm are two or more triangular mesh models sharing the same vertex connectivity. Each model can be interpreted as three scalar fields (vertex positions) defined on a common domain that is actually an abstract mesh structure. We call this structure *domain mesh*. A domain mesh maintains its own vertex positions to provide metric information for discrete differential operators in our method. Without loss of generality, we assume the input models and domain meshes are all single-connected and 2-manifold triangular meshes throughout this paper.

**Mesh scalar (or vector) field.** Let  $f_i$  be the scalar (or vector) value attached to vertex  $v_i$  of domain mesh  $\mathcal{M}$ . A *mesh scalar(or vector) field* f on  $\mathcal{M}$  is defined to be a piecewise linear combination  $f(v) = \sum_i f_i \phi_i(v)$  (v is a point on  $\mathcal{M}$ ), where  $\phi_i(\cdot)$  is piecewise linear basis function defined on domain mesh valued 1 at vertex  $v_i$  and 0 at all other vertices. It is obvious that  $\mathcal{M}$  itself is a special mesh vector field of  $\mathcal{M}$ .

**Discrete differential operators.** The *discrete gradient* of mesh scalar function f on the domain mesh  $\mathcal{M}$  is expressed as

$$\nabla f(v) = \sum_{i} f_i \nabla \phi_i(v). \tag{3}$$

Given a piecewise constant vector field  $\mathbf{w}$ , which has constant value in each triangle of  $\mathcal{M}$ , the *discrete divergence* of  $\mathbf{w}$  at vertex  $v_i$  is defined as

$$(\operatorname{div} \mathbf{w})(v_i) := \sum_{T \in N_T(v_i)} \mathbf{w}(T) \cdot \nabla \phi_i |_T A_T.$$
(4)

where  $A_T$  denotes the area of triangle *T*. Therefore, the *discrete Laplacian operator* on domain mesh  $\mathcal{M}$  is

$$\Delta f(v_i) = \sum_{v_j \in N_v(v_i)} \frac{1}{2} (\cot \alpha_j + \cot \beta_j) (f_i - f_j), \tag{5}$$

where  $\alpha_j$  and  $\beta_j$  are the two angles opposite to the edge  $(v_i, v_j)$  (Figure 2).

Finally, the discrete Poisson equation is expressed as

$$\Delta f \equiv \operatorname{div}(\nabla f) = \operatorname{div}\mathbf{w}.$$
(6)

With specified boundary conditions, the above equation can be reformulated as a sparse linear system

$$A \mathbf{x} = \mathbf{b}.\tag{7}$$

where the unknown vector  $\mathbf{x}$  represents coordinates to be reconstructed. The coefficient matrix A is determined by Eq. (5) and depends on  $\mathcal{M}$ . The vector  $\mathbf{b}$  corresponds to known vector field as well as the boundary conditions.

#### 2.2 Poisson Shape Interpolator

The fundamental of our algorithm is a Poisson shape interpolator. Two models  $M_0$  and  $M_1$ , given as input of our algorithm, are referred as the source and the target, respectively. Note that this is not a restriction since our method is also applicable to multiple input models. In the pre-processing step, the one-to-one correspondence map is established. After that, the domain mesh  $\mathcal{M}$  is determined. The re-sampled versions of  $M_0$  and  $M_1$ , treated as mesh scalar fields

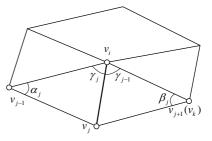


Figure 2: The 1-ring neighborhood setup.

defined on  $\mathcal{M}$ , are denoted as  $S_0^i$  and  $S_1^i$  (i = x, y, z), respectively. We denote the corresponding gradient fields as  $G_j^i = \nabla S_j^i$  (i = x, y, z; j = 0, 1).

Most of previous vertex interpolation methods are position-based. They interpolate vertex coordinates directly, i.e.,

$$S_t^i = Interpolate(S_0^i, S_1^i, h(v, t)).$$
(8)

On the contrary, our method is gradient field-based interpolation. It is formulated as

$$\begin{cases} G_t^i = Interpolate(G_0^i, G_1^i, h(v, t); \mathcal{M}); \\ S_t^i = Poisson(G_t^i). \end{cases}$$
(9)

where *v* denotes a point on the domain mesh  $\mathcal{M}$ ; *t* represents time whose value lies in [0, 1], *Interpolate*(·) is the interpolation operator, and *Poisson*(·) indicates the Poisson equation solver for reconstructing in-between shapes. The function h(v,t) is the so-called *transition state function* whose value also lies in [0, 1], and satisfies  $h(\cdot, 0) = 0$  and  $h(\cdot, 1) = 1$ . The simplest transition state function is h(v,t) = t. The transition state function is used to provide flexible non-uniform controls, which will be discussed in Section 4.3.

There are two motivations of applying gradient-based method, which are also emphasized in [Yu et al. 2004]. First, as a differential property, the gradient can be modified locally, which allows the local analysis and interpolation to be carried out in a more canonical way (Section 2.3). Second, the process of reconstructing shapes from the interpolated gradient fields  $G_t^i$  can be regarded as performing Helmholtz-Hodge decomposition [Tong et al. 2003] on  $G_t^i$ . The gradient of  $S_t^i$  is curl-free part of  $G_t^i$ . Note that the discrete form of Poisson equation is equivalent to the following least-squares minimization

$$\min_{S_f^i} \sum_{T \in \mathcal{M}} \left\| \nabla S_f^i - G_f^i \right\|^2 A_T \tag{10}$$

Since the least-squares minimization tends to distribute errors uniformly across the function, the reconstruction process can effectively attenuate the inconsistency evoked by local gradient interpolation. Therefore, the local winkles are smoothed.

One question aroused immediately is whether our Poisson interpolator can produce smooth-changing results along time *t*, since a nonlinear reconstruction is involved in our algorithm. According to Eq. (7), the vertex coordinates  $\mathbf{x}(t)$  are obtained by  $A^{-1}\mathbf{b}(t)$ . It is clear that the matrix *A* is constant because the domain mesh is fixed during morphing, and  $\mathbf{b}(t)$  comes from the smoothly changing gradients and boundary conditions. It follows that  $\mathbf{x}(t)$  is smoothly changed.

A well-defined shape interpolation method must hold the following two properties. One is *uniqueness property*, i.e., the reconstructed mesh model should be unique. Another is *end-point interpolation* 

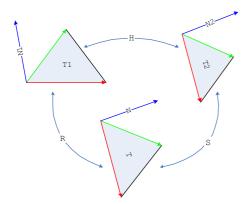


Figure 3: The matrix polar decomposition.

property, i.e., the reconstructed mesh model should be equivalent to the input models when transition state function h = 0, 1. Provided the domain mesh M is a non-degenerated, single-connected, 2manifold one and at least one vertex coordinates are fixed as Dirichlet boundary condition, we can prove a sufficient condition for the uniqueness property with similar deductions to [Floater 1997]. This sufficient condition is, for every edge  $(v_i, v_j)$  in M, the sum of two opposite angles  $\alpha_i$  and  $\beta_i$  is less than  $\pi$  (see also Figure 2). Under this condition, it is easy to show that the end-interpolation property also holds.

#### 2.3 Nonlinear Gradient Field Interpolation

In general, any gradient field interpolator satisfying end-point conditions  $G_0^i$  and  $G_1^i$  can be used in our framework. For example, direct vertex linear interpolator is equivalent to applying  $(1-h)G_0^i + hG_1^i$ , since both the gradient operator and the divergence operator are linear on a given domain mesh. It works well for well-aligned objects. However, according to the analysis above, the naive linear interpolation may cause shape shrinkage when objects with large orientation difference. Therefore, we design a nonlinear gradient field interpolation method.

One possible solution is to employ additional nonlinear warping along with linear interpolation, which is commonly used in image based morphing techniques [Cohen-Or et al. 1998; Lazarus and Verroust 1998]. In these techniques, warping is applied to globally align objects by combining rotation, stretching and translation factors. Different from this two-step solution, we directly interpolate these transformation factors for each triangle, and overall global effects are ensured by the Poisson solver. The detail of our solution is described as follows.

To determine transformation, local affine frames are defined for every triangle pair, which are served as the canonical basis for local movement analysis. Let  $T_0$  and  $T_1$  be a pair of corresponding triangles in  $S_0$  and  $S_1$ , respectively. We denote  $\mathbf{v}_0^i$  and  $\mathbf{v}_1^i$  (i = 1, 2, 3) as corresponding vertices,  $\mathbf{n}_0$  and  $\mathbf{n}_1$  as corresponding unit face normals, as shown in Figure 3. For gradients are translation invariant, we choose the first vertex  $\mathbf{v}_j^1$  (j = 0, 1) of each triangle as the origin, and the three axes of affine frames  $F_j$  are  $\mathbf{v}_i^2 - \mathbf{v}_j^1$ ,  $\mathbf{v}_j^3 - \mathbf{v}_j^1$  and  $\mathbf{n}_j$ .

We can determine a unique transform matrix H such that  $F_1 = H(F_0)$ . The matrix H can be regarded as the deformation gradient relating the reference configuration  $F_0$  and the present configuration  $F_1$ . Inspired by continuum mechanics [Gurtin 1981], we factorize the deformation gradient tensor into the rigid rotation part and the

pure stretch part with the polar decomposition. That is, H = RS, where *R* is the closest rotation matrix to *H* in Frobenius norm, *S* is a symmetric, positive definite matrix.

The rotation part and the stretch part are interpolated independently. Our experimental results indicate that this can avoid shape shrinkage (see Figure 1 and the attached video). Since there is no prior knowledge of movements in morphing process, it is reasonable to assume that the rotation angle and the scaling components change linearly. Similar to [Alexa et al. 2000; Shoemake and Duff 1992], we define the local continuous transform function  $H_t$  for a given transition state h(v,t) = h as

$$H_t = R_h((1-h)I + hS)$$
(11)

where  $R_h$  is the rotation matrix defined by linearly interpolating the rotation angle of R using quaternion, and I is the identity matrix.

Since the three scalar fields come from the three vertex coordinates of mesh model, manipulating them independently may cause undesired reconstruction results. Therefore, we compute the local continuous transform  $H_t$  for each triangle using the above algorithm. We then apply  $H_t$  to three source gradient vectors simultaneously for generating interpolated gradient vectors. That is,

$$G_t^i = H_t(G_1^i) \ (i = x, y, z).$$
 (12)

In some sense, our shape interpolation algorithm can be considered as a mixture of [Alexa et al. 2000] and [Sumner and Popović 2004]. All three methods adopt local affine transformation matrix to describe the relationship between two compatible shapes. However, there are several differences among them. In [Alexa et al. 2000], the local affine transformation matrices are determined between simplicial complexes. In a 3D space, a simplicial complex is a tetrahedral mesh, while we only deal with triangular meshes for better computational efficiency. Same as [Sumner and Popović 2004], we also consider the corresponding face normals when defining local transforms between triangle pairs. But [Sumner and Popović 2004] aims at how to transfer the transformation matrices from the source model to the target one, instead of interpolating between them. The other difference is the formulation of opimization problem. [Alexa et al. 2000; Sumner and Popović 2004] reconstruct meshes by minimizing the differences among transformation matrices using the Frobenius norm, while our method directly minimizes least-squares differences between vectors using the  $L^2$  norm which is an inherent property of Poisson equation itself.

After gradient fields interpolation, each triangle is locally transformed by the transformation  $H_t$ . The triangles of source mesh become disconnected, i.e., yielding a triangular soup. The Poisson equation stitches together the triangles in the final step.

## 3 Implementation Issues

In Section 2, we present our basic framework of Poisson-based shape interpolation. From the implementation aspect, there are several issues that need to be elaborated. Obtaining compatible meshes requires building correspondence map. During the interpolation procedure, boundary conditions are specified to determine continuous movements of in-between shapes. Our method is robust for the choice of domain meshes. With better selections of domain mesh, our method will provide more satisfied results. The details of these issues are discussed in the following paragraphs.

**Obtain compatible meshes.** Our method requires that the source model and the target one should be represented by compatible

meshes, i.e. meshes with the same connectivity. In general, the input models do not satisfy this requirement. Recently, Kraevoy and Sheffer [2004] have presented a novel feature-preserving remeshing method that can generate high quality output meshes with significantly fewer number of vertices compared to previous techniques. We adopt a variant of their method to generate compatible meshes from input models. Our method is independent of compatible meshes generating methods. Other methods, such as [Praun et al. 2001; Schreiner et al. 2004], are also suitable for our framework.

In our implementation, several pairs of corresponding feature vertices are manually selected. Then, base domain is constructed based on these feature vertices. Both models are parameterized onto the common base domain and relaxation is performed to reduce the parametrization distortion. Finally, the target model is remeshed using the connectivity of the source model. Iterative error-driven vertex relaxation and edge splitting are performed until approximation error is under user-specified threshold.

**Specify boundary conditions.** Since the gradient fields are translation independent, Poisson equations require specifying at least one vertex coordinates as the boundary conditions. Note that, in most cases, input models to be interpolated have more than one corresponding feature vertex pairs. We can select one of these vertex pairs and specify their coordinates by linear interpolation as the boundary conditions. Alternatively, we can set the coordinates of an arbitrary vertex as boundary conditions and solve the Poisson equation first, and then translate these intermediate models so that their barycenters linearly interpolate the barycenters of the source and the target models.

**Determine domain mesh.** In principle, any mesh sharing the same connectivity with  $\mathcal{M}$  can be selected as the domain mesh. Different domain meshes generally give rise to different interpolation results, since the domain mesh affects the coefficients of the linear system. Due to the nature of the least-squares minimization in Eq. (10), the interpolated gradient vectors associated with large triangles in the domain mesh are better approximated than those associated with small ones. The area of the triangles serves as the weight coefficients. In practice, either the source model or the target model can be selected as the domain mesh. In all our experiments, we do not observe significant difference between two morphing sequences generated with different choices of the domain mesh. In case that some vertices in the domain mesh violate the sufficient condition of the uniqueness property, local adjustment of vertex coordinates will be performed so that this condition can be satisfied.

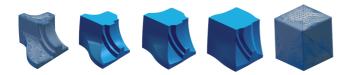


Figure 4: Shape interpolation between a fandisk and a cube.



Figure 5: Human pose morphing



Figure 6: Shape interpolation between a bunny and a rabbit.

### 4 Results and Discussions

Our framework can be applied in various scenarios, including global shape interpolation, local control of shape interpolation and morphing incorporating with deformation.

#### 4.1 Global Shape Interpolation

In order to present the advantages of our method, several examples are demonstrated in Figure 4-8 and attached video sequences. Figure 4 depicts the morphing process between a fandisk (CAD model) and a cube. As shown in this example, the sharp features of fandisk are preserved during the whole morphing process. The transitions from a bunny to a rabbit are illustrated in Figure 6. Note that the natural gluing of ears and the rotation of the head and tail. Figure 5 shows that our method can also work well on human pose interpolation. Human pose interpolation are commonly generated by skeleton-driven skinning in computer animation, which may require massive user interactions for tuning influence region parameters. In our example, no explicit bone information are specified. However, the arms and legs are smoothly transited, respectively. Meanwhile global rotations and local muscle deformations are all well generated. In Figure 8, morphing among multiple models is demonstrated. The three head models have different orientations. Our method can automatically produce the rotation effects during morphing.

Table 1 lists the computation time for the examples in this paper. The data are obtained on a standard PC with an Intel PIV 2.6GHz CPU and 512MB memory. In the third column, running time refers to the average executive time of computing one intermediate model, which is dominated by solving three Poisson equations for each coordinate components to reconstruct mesh models. In our implementation, we adopt conjugate gradient (CG) method to solve sparse linear systems derived from Poisson equations. In the fourth column of Table 1, the precision is refers to the convergence tolerance of our CG method, which is relative to the diagonal length of the model's bounding box. Our morphing technique can generate in-between models in interactive speed for moderate data sets using our non-optimized C/C++ code. As shown in [Yu et al. 2004], pre-computing can be applied to the Poisson matrix to accelerate generating speed when massive in-between models are required.

#### 4.2 Local Shape Interpolation

Local shape interpolation, also known as spatially non-uniform shape morphing, is usually carried out by specifying regiondependent transition states. This topic is also studied by Alexa [2003]. Unlike global morphing, linear interpolation of absolute coordinates will cause significant artifacts. To address this problem, Alexa proposes blending of two shapes by linearly interpolating the Laplacian coordinates instead. Laplacian coordinate is defined by the difference between given vertex position and the

Morphing models	#Unknowns	Per-frame Running	Precision
		Time (sec)	
Fandisk vs. Cube	6583	1.633	1.0e-6
Bunny vs. Rabbit	7862	3.626	1.0e-6
Igea, Man and Planck	10139	5.252	1.0e-6
Horse vs. Dinosaur	10993	7.440	1.0e-6
Woman vs. Man	25172	22.912	1.0e-5
Woman vs. Man (local)	2900	1.039	1.0e-6

Table 1: Computation time for the examples used in this paper.

average position of its 1-ring neighbors. Although Laplacian coordinate is translation invariant, it is sensitive to rotation and nonuniform scaling. As pointed out by Alexa, the result is equivalent to those of linear interpolation of absolute vertex coordinates, if global morphing using linear interpolation of Laplacian coordinates is performed.

Therefore, when local morphing is performed between two parts which are not well aligned, the method of Alexa [2003] has similar drawback as linearly interpolating absolute vertex coordinates, such as shape shrinkage. Figure 7 gives the comparison between the method proposed in [Alexa 2003] (in the bottom row) and ours (in the top row). We set region of interest (ROI) to be the left arm of the human model, then change the transition state of vertices in ROI gradually from 0 to 1, and keep the rest unchanged. Our results are visually pleasing and do not suffer shape shrinkage, which is supported by the volume-changing curve plot in the bottom of Figure 7. Our local interpolation algorithm achieves nearly constant volume changing rate. For similar articulated objects with different poses, our gradient field interpolator can preserve the shape locally in the means of least area distortion. Note that the Poisson solver is a global vector field optimizer to keep local geometric properties. Therefore, with additional surface orientation constraints, the superfluous volume variations can be attenuated.

#### 4.3 Incorporating Deformation into Shape Interpolation

The basic shape interpolation process focuses on how to change the shape gradually from a source model to a target one. Common approaches to enrich morphing effects are defining different vertex paths, or region-sensitive transition states. Intuitive ways are desired to provide user additional freedom on generating creative morphing sequences. In our point of view, deformation results can be cast as self-morphing sequences if the editing process does not change the original vertex connectivity. Based on this idea, our framework incorporates user-defined deformation into the morphing sequence. Hence, interesting visual effects can be obtained.

The users specify desired deformation by manually editing the source model with mesh editing tools. The deformation is encoded locally into one transformation matrix for each triangle pair, which is called *deformation matrix*. Similar to Section 2.3, we apply polar decomposition to this deformation matrix. Users specify how much deformation will be blended by setting one weight ranged in [0, 1]. Deformation matrix and morphing matrix are composed together to compute the intermediate gradient field. Finally, the intermediate meshes are reconstructed by the Poisson solver.

Figure 9 demonstrates an example of combining morphing with deformation. The top row shows the original morphing sequence from a horse to a dinosaur. We interactively edit the horse and gradually propagate such deformation into the whole morphing se-

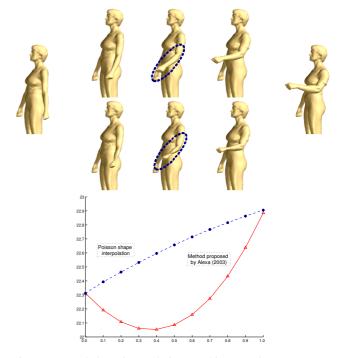


Figure 7: Local shape interpolation. In this example, users expect to gradually deform the arm part of the woman model to the corresponding part of the man model. The upper row is generated by our algorithm, while the lower row is generated by Alexa's algorithm. The comparison of corresponding volume changes is illustrated in the bottom plot.

quence. The middle and the bottom rows are results with 50% and 100% deformation incorporated, respectively.

## 5 Conclusions and Future Work

In this paper, we have proposed a novel shape interpolation approach, which is based on the discrete Poisson equation on triangular meshes. The main idea is to interpolate vector fields in gradient domain and to reconstruct in-between shapes from the interpolated gradient fields. Based on the polar decomposition of local transformations, a non-linear gradient field interpolation method is designed to gradually change both vertex coordinates and face normals. We demonstrate the superiority of our approach by versatile results, ranging from global morphing, local morphing to those combining with deformation.

The morphing results depend on the quality of compatible meshes in our current implementation. The computing speed is affected

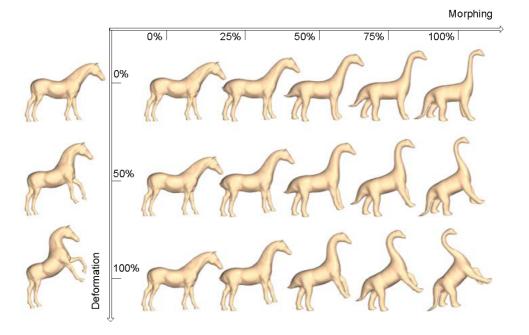


Figure 9: An example of incorporating deformation into morphing. The first row is original morphing sequence. The second and third row are results with 50% and 100% deformation incorporated, respectively.

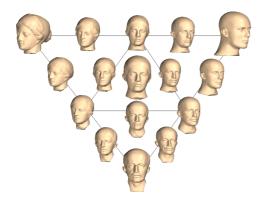


Figure 8: Shape interpolation among three head models.

by the condition number of Poisson matrix, which is related to the quality of the domain mesh. To improve the performance of our algorithms, multi-resolution techniques may be employed. Changing object topology during shape interpolation is a challenging problem. We will investigate it in the future. Three dimensional mesh morphing without self-intersection is a valuable research topic. Since our method provides a general framework to deal with shape interpolation, a similar 2D contour morphing can be derived by formulating discrete Poisson equation on a 2D simple polygon.

### Acknowledgements

We wish to thank the anonymous reviewers for their valuable comments, Dr. Kun Zhou for helpful discussions, Dr. Zhongding Jiang and Mr. Hongbo Fu for careful proof-reading. Thanks to Mr. Ran Zhou and Mr. Lu Chen for their helping in video production. Models are courtesy of Cyberware, Stanford University and Max-Planck-Institut für Informatik. This project is supported in partial by 973 Program of China (No.2002CB312102) and NSFC (No.60021201 and No.60033010).

### References

- ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigidas-possible shape interpolation. In *Proceedings of ACM SIG-GRAPH 2000*, ACM Press, 157–164.
- ALEXA, M. 2002. Recent advances in mesh morphing. *Comput. Graph. Forum 21*, 2, 173–196.
- ALEXA, M. 2003. Differential coordinates for local mesh morphing and deformation. *The Visual Computer 19*, 2 (May), 105– 114.
- BREEN, D., AND WHITAKER, R. 2001. A level-set approach for the metamorphosis of solid models. *IEEE Transactions on Visualization and Computer Graphics* 7, 2 (April-June), 173– 192.
- CHEN, E., AND PARENT, R. 1989. Shape averaging and it's applications to industrial design. *IEEE Computer Graphics and Applications* 9, 1 (January), 47–54.
- COHEN-OR, D., SOLOMOVIC, A., AND LEVIN, D. 1998. Threedimensional distance field metamorphosis. *ACM Trans. Graph. 17*, 2, 116–141.
- FLOATER, M. S. 1997. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design 14*, 3, 231–250.
- GURTIN, M. 1981. An Introduction to Continuum Mechanics. Academic Press, N.Y.

- HU, S.-M., LI, C.-F., AND ZHANG, H. 2004. Actual morphing: A phsical-based approach for blending two 2d/3d shapes. In ACM Symposium on Solid Modeling and Applications.
- KANAI, T., SUZUKI, K., AND KIMURA, F. 2000. Metamorphosis of arbitrary triangular meshes. *IEEE Computer Graphics and Applications* 20, 2, 62–75.
- KRAEVOY, V., AND SHEFFER, A. 2004. Cross-parameterization and compatible remeshing of 3d models. ACM Trans. Graph. 23, 3, 861–869.
- LAZARUS, F., AND VERROUST, A. 1998. Three-dimensional metamorphosis: a survey. *Visual Computer 14*, 373–389.
- LEE, A., DOBKIN, D., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution mesh morphing. *Computer Graphics Proceedings* (SIGGRAPH '99), 343–350.
- LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., RÖSSL, C., AND SEIDEL, H.-P. 2004. Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International*, IEEE Computer Society Press, 181–190.
- LIU, L.-G., AND WANG, G.-J. 1999. Three-dimensional shape blending: intrinsic solutions to spatial interpolation problems. *Computers & Graphics 23*, 4, 535–545.
- PERÉZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Trans. Graph.* 22, 3, 313–318.
- POLTHIER, K., AND PREUSS, E. 2002. *Visualization and Mathematics III*. Springer Verlag, ch. Identifying vector fields singularities using a discrete hodge decomposition.
- PRAUN, E., SWELDENS, W., AND SCHRÖDER, P. 2001. Consistent mesh parameterizations. In *Proceedings of ACM SIG-GRAPH 2001*, ACM Press, 179–184.
- SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. *ACM Trans. Graph.* 23, 3, 870–877.
- SEDERBERG, T. W., AND GREENWOOD, E. 1992. A physically based approach to 2-D shape blending. In *Computer Graphics* (*SIGGRAPH '92 Proceedings*), E. E. Catmull, Ed., vol. 26, 25–34.
- SEDERBERG, T. W., GAO, P., WANG, G., AND MU, H. 1993. 2-d shape blending: An intrinsic solution to the vertex path problem. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, vol. 27, 15–18.
- SHAPIRA, M., AND RAPPOPORT. 1995. A. shape blending using the star-skeleton representation. *IEEE Computer Graphics and Application 15*, 2, 44–51.
- SHEFFER, A., AND KRAEVOY, V. 2004. Pyramid coordinates for morphing and deformation. In *Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT04).*
- SHOEMAKE, K., AND DUFF, T. 1992. Matrix animation and polar decomposition. In *Proceedings of the conference on Graphics Interface* '92, 258–264.
- SORKINE, O., LIPMAN, Y., COHEN-OR, D., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Eurographics Association, 179–188.

- STAM, J. 1999. Stable fluids. In Proceedings of ACM SIGGRAPH 1999, 121–128.
- SUMNER, R. W., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph.* 23, 3, 399–405.
- SURAZHSKY, V., AND GOTSMAN, C. 2003. Intrinsic morphing of compatible triangulations. In 4th Bi-National Israel-Korea Conference on Geometric Modelling and Computer Graphics, 45–50.
- TONG, Y., LOMBEYDA, S., HIRANI, A. N., AND DESBRUN, M. 2003. Discrete multiscale vector field decomposition. ACM Trans. Graph. 22, 3, 445–452.
- TURK, G., AND O'BRIEN, J. F. 1999. Shape transformation using variational implicit functions. In *Proceedings of ACM SIG-GRAPH 1999*, 335–342.
- YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph.* 23, 3, 644–651.