Physically Based Morphing of Point-sampled Surfaces

Yunfan Bao Xiaohu Guo Hong Qin
Computer Science Department
State University of New York at Stony Brook
{ybao,xguo,qin}@cs.sunysb.edu
http://www.cs.sunysb.edu/{∼ybao,∼xguo,∼qin}

Abstract

This paper presents an innovative method for naturally and smoothly morphing point-sampled surfaces via dynamic meshless simulation on point-sampled surfaces. While most existing literature on shape morphing emphasizes the issue of finding a good correspondence map between two object representations, this research primarily investigates the challenging problem of how to find a smooth, physically-meaningful transition path between two homeomorphic point-set surfaces. We analyze the deformation of surface involved in the morphing process using concepts in differential geometry and continuum mechanics. The morphing paths can be determined by optimizing an energy functional which characterizes the intrinsic deformation of the surface away from its rest shape. As demonstrated in the examples, our method automatically produces a series of natural and physically-plausible in-between shapes, which greatly alleviates the shrinking, stretching, and self-intersection problems that often occur when linear interpolation is employed for the morphing of two objects. We envision that our new technique will continue to broaden the application scope of point-set surfaces and their dynamic animation.

Keywords: physically based animation, dynamic shape modeling, morphing, meshless method, point-based geometry

1 Introduction

Point-based representation for surfaces has gained strength and become an attractive, powerful modeling and rendering alternative in Computer Graphics in recent years. The rapid development of the 3D scanning devices has greatly facilitated the acquisition of the point samples from real-world physical objects. Since point-sampled geometry can be represented by a set of discrete points, no explicit connectivity information needs to be maintained. This leads to a more compact and flexible representation of geometry in terms of data storage and transfer. Many researchers have devoted considerable amount of effort to the accurate representation, efficient processing and rendering of point-sampled geometry. However, rapid and accurate animation/simulation of point-sampled geometry is still a challenging area that demands a great deal of research endeavors within point-based graphics.

As a popular animation technique in digital entertainment, shape morphing (or blending) has been a very active research topic in Computer Graphics. Given a source object and a target object, morphing techniques can be employed to create a series of in-between shapes that transform the source object into the target one.

In this paper, we systematically develop a physically based *meshless method* to morph two objects represented only by points. To our best

knowledge, this is a first attempt to conduct a dynamic, physically-meaningful shape morphing between two point-sampled surfaces. In contrast, most existing morphing techniques nowadays are based on the polygonal mesh representation, owing to the popularity and long history of mesh geometry (as the dominant shape representation) in Graphics. Despite the widespread use of mesh-based shape morphing techniques, certain drawbacks persist. For example, the map overlay algorithm needed to generate any intermediate mesh with consistent connectivity from source and target surface meshes are notoriously difficult to implement. This remeshing process can be circumvented by using point sampled surfaces since no connectivity information is necessary during shape morphing.

Given two shapes, there are infinite possible morphing paths. Our method focuses on finding an ideal transformation via physics-based energy optimization. We approach the morphing path problem from the viewpoint of both differential geometry and continuum mechanics. Our method is completely mesh free and only the vicinity information of the point set is utilized in the morphing procedure. In our framework, the point-sampled geometry is modeled as a meshless thin shell surface. The energy functional is defined using the intrinsic measurement of surface deformation derived from differential geometry. The morphing path is then automatically derived by minimizing the energy functional. One key benefit of our method is that it minimizes unnecessary distortion such as shrinking, stretching, and self-intersection of the surface as demonstrated in Figure 1, which often exists in interpolation-based methods. Furthermore, surface crack problem associated with point-based morphing [1] can be avoided in our physically based system. Therefore, the intermediate shapes generated by our method are both physically plausible and visually natural as shown in Figure 2.

2 Related Work

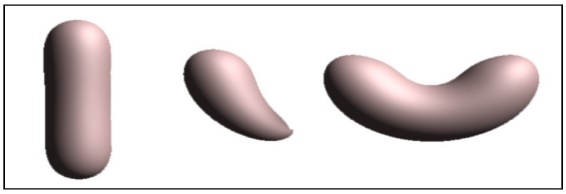
2.1 Shape Morphing

Shape morphing is an active and interesting research area in computer animation. A complete survey is beyond the scope of this paper and we shall only briefly review several literatures that are most relevant to our work here. The readers are referred to [2, 3] for more details. The key to mesh-based morphing is to establish a good mapping between the source object and the target object, which is known as the correspondence problem. Defining such a mapping is far from trivial, since it usually involves parameterization [4] of surfaces of arbitrary genus and the problem becomes much more challenging when the two surfaces have different topology. After the parameterization step, a map overlay algorithm is used to generate the common connectivity.

While the correspondence problem has been extensively studied by researchers, the path problem is relatively less-explored, hence demanding more research efforts towards further improvement. The goal of the path problem is to find an ideal transition path between the source and target shapes. However, the fundamental question of *"What Constitutes an Ideal Path?"* is rather subjective. A naive choice for most morphing applications is linear interpolation because of its simplicity. However, a straightforward application of linear interpolation can somehow lead to displeasing visual results as shown in Figure 1, especially in cases that involve dramatic, near-rigid-body transformation between the end shapes. To address this kind of problems, Sederberg et al. [5] introduced a technique that minimizes the deformation of the boundaries of 2D shapes. Alexa et al. [6] took the interior of shapes into account and decomposed an affine transformation to improve the morphing quality. One similar work to ours is that of Yan et al. [7], which is based on nonlinear strain field interpolation derived from physics. While their method can only morph two planar polygons, our novel method handles complicated 3D point-sampled surfaces. Furthermore, we derive the formulation for meshless, thin-shell dynamics as linear systems, which is then solved to reconstruct "in-between" shapes.

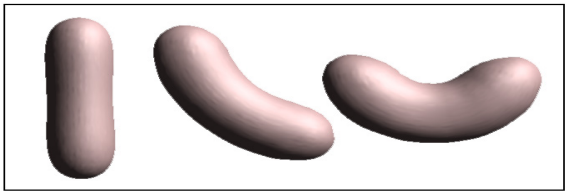
2.2 Physically Based Animation and Meshless Methods

Following the pioneering work of Terzopoulos and his co-workers [8], a large amount of re-



(a) Initial shape (b) 50 % morph (c) Final shape

Figure 1: The undesirable artifact caused by linear blending.



(a) Initial shape (b) 50 % morph (c) Final shape

Figure 2: Our method leads to a physically-plausible intermediate shape.

search has been carried out in the field of physically based animation. For example, many mesh-based methods for physical simulation of deformable objects have been proposed in computer graphics based on either the boundary element method [9] or the finite element method [10]. Cloth, which can be modeled as a thin-plate, was recently studied [11, 12]. Compared with cloth, thin-shell objects are naturally curved and can not be modeled using plate formulations [13]. Grinspun et al. [14] proposed a simple discrete shell model that can be derived geometrically for triangle meshes. Most of the existing physical simulation approaches are based on mesh structures. For point-sampled surfaces, however, it would be extremely challenging to simulate the continuum mechanics without the explicit information of mesh connectivity.

In recent years, considerable research has been devoted to the efficient modeling and animation of point-sampled geometry. In [15], the Pointshop 3D system was presented for interactive shape and appearance editing of 3D point sampled geometry. Later, Pauly et al. [16] presented a free-form shape modeling framework for point sampled geometry using the implicit surface definition of the Moving Least Squares (MLS) approximation. More recently, Guo et al. [17] developed a framework for local and global editing of point set surfaces based on level-set method. Mueller et al. [18] developed a method for animating elastic, plastic and melting point-based volumetric objects based on the MLS approximation of the gradient of the displacement field. Xiao et al. [1] proposed an approach for interactive morphing of point surfaces based on Floater's [19] meshless parameterization. Most recently, Pauly et al. [20] presented a new meshless animation framework for elastic and plastic materials that fracture. Guo and Qin [21] combined the meshless method with the modal warping technique to achieve real-time deformation. In this paper, we will demonstrate that the meshless method is a natural and intuitive solution to performing physical simulation directly on point-sampled surfaces, hence leading to the automatic, physically-plausible shape morphing of point-set geometry.

3 Deformation Analysis

3.1 Deformation of Surface

The deformation of surfaces and solids has been well studied in elasticity theory and continuum mechanics. In this paper, we define the deformation of a surface using concepts in differential geometry [22]. A surface in $\mathbb{R}^3$ can be defined parametrically as a vector function $\mathbf{X}(\theta^1, \theta^2)$, where θ^1 and θ^2 are the parametric coordinates of the surface. The quadratic form (in Einstein summation convention) $ds^2 = g_{ij} d\theta^i d\theta^j$ measures the length of an arc element on the surface, and is known as the first fundamental form. The coefficients g_{ij} are components of a covariant tensor which is called the metric tensor or fundamental tensor:

$$g_{ij}(\mathbf{X}) = \mathbf{X}_{,i} \cdot \mathbf{X}_{,j},$$

where $\mathbf{X}_{,i} = \frac{\partial \mathbf{X}}{\partial \theta_i}$. Intuitively, the first fundamental form measures the stretching and shearing of the underlying surface.

The first fundamental form alone does not completely determine the shape of a surface, because the curvature can be altered without affecting the metric. Hence the second fundamental form which takes the form $b_{ij} d\theta^i d\theta^j$ needs to be introduced. The second fundamental form measures the curvature of a surface and the coefficients b_{ij} are components of a tensor called

the curvature tensor:

$$b_{ij}(\mathbf{X}) = \mathbf{X}_{,ij} \cdot \mathbf{n} = -\mathbf{X}_{,i} \cdot \mathbf{n}_{,j},$$

where $\mathbf{X}_{,ij} = \frac{\partial^2 \mathbf{X}}{\partial \theta_i \partial \theta_j}$, $\mathbf{n}_{,i} = \frac{\partial \mathbf{n}}{\partial \theta_i}$ and $\mathbf{n}$ is the unit normal vector. The second fundamental form, together with the first fundamental form, can entirely determine the shape of a surface and therefore is an intrinsic measurement of the shape of the surface.

3.2 Deformation Energy

Having defined the metric tensor and the curvature tensor, the elastic strain energy for a deformable surface is given by [8]

$$\xi = \int_\Omega (\alpha ||g^n - g^0||^2 + \beta ||b^n - b^0||^2) d\Omega,$$

where g^0, b^0 are the metric tensor and curvature tensor associated with the rest shape of the surface; g^n, b^n are corresponding tensors associated with the deformed shape of the surface and $||\cdot||$ is the Frobenius norm. The first term in the strain energy formulation is the membrane energy which resists stretching and shearing and the second term is the bending energy which resists bending and twisting. The membrane and bending energy distribution of a deforming torus is shown in Figure 7.

The above formulation of the deformation energy is purely motivated by differential geometry and applies to any surface in $\mathbb{R}^3$. For any point-sampled surfaces, if we assume that one dimension, i.e., the thickness, of the surface body is significantly smaller than the other two dimensions, we can consider the point-sampled surface as a thin shell. The generic configuration of the shell can be described as

$$S = \{\mathbf{x} \in R^3 | \mathbf{x} = \mathbf{X}(\theta_1, \theta_2) + \theta_3 \mathbf{X}_{,3}(\theta_1, \theta_2), \\ \theta_1, \theta_2 \in \Omega \text{ and } -\frac{h}{2} \leq \theta_3 \leq \frac{h}{2}\},$$

where $\mathbf{X}_{,3}$ is a unit director vector normal to the middle surface of the shell both in the reference and deformed configuration under the Kirchhoff-Love hypothesis. In the Kirchhoff-Love thin shell framework, the deformation of the surface body is fully characterized by the deformation of the middle surface. The Green-Lagrange strain tensor can therefore be derived from the first and second fundamental forms of the middle surface of the shell. The membrane and bending strain tensors are related to the deformation of the shell surface as follows

$$\epsilon_{ij} = \frac{1}{2}(\mathbf{X}_{,i} \cdot \mathbf{X}_{,j} - \mathbf{X}^0_{,i} \cdot \mathbf{X}^0_{,j}),$$
$$\rho_{ij} = \frac{1}{2}(\mathbf{X}_{,i} \cdot \mathbf{X}_{,3j} - \mathbf{X}^0_{,i} \cdot \mathbf{X}^0_{,3j}).$$

Here, we use the superscript "0" to denote the measurement in the original (reference) configuration. Assuming linearized kinematics, the displacement field of the middle surface is introduced as $\mathbf{u}(\theta_1, \theta_2) = \mathbf{X}(\theta_1, \theta_2) - \mathbf{X}^0(\theta_1, \theta_2)$. Thus, the linearized membrane and bending strain can be written as

$$\epsilon_{ij} = \frac{1}{2}(\mathbf{X}^0_{,i} \cdot \mathbf{u}_{,j} + \mathbf{X}^0_{,j} \cdot \mathbf{u}_{,i}), \quad (1)$$
$$\rho_{ij} = \frac{1}{2}(\mathbf{X}^0_{,i} \cdot \Delta\mathbf{X}_{,3j} + \mathbf{X}^0_{,j} \cdot \Delta\mathbf{X}_{,3i} \\ + \mathbf{u}_{,i} \cdot \mathbf{X}^0_{,3j} + \mathbf{u}_{,j} \cdot \mathbf{X}^0_{,3i}). \quad (2)$$

Introducing the Kirchhoff-Love hypothesis explicitly here, the deformed shell director vector is constrained to coincide with the unit normal vector to the deformed middle surface of the shell, i.e.,

$$\mathbf{X}_{,3} = \mathbf{J}^{-1}(\mathbf{X}_{,1} \times \mathbf{X}_{,2}),$$

where $\mathbf{J} = |\mathbf{X}_{,1} \times \mathbf{X}_{,2}|$. Therefore, the derivatives of the director vector in the reference configuration are

$$\mathbf{X}^0_{,3i} = (\mathbf{J}^0)^{-1}(\mathbf{X}^0_{,1i} \times \mathbf{X}^0_{,2} + \mathbf{X}^0_{,1} \times \mathbf{X}^0_{,2i}) \quad (3)$$

The increment $\Delta\mathbf{X}_{,3} = \mathbf{X}_{,3} - \mathbf{X}^0_{,3}$ can also be derived straightforwardly by only keeping linear terms.

$$\Delta\mathbf{X}_{,3} \approx (\mathbf{J}^0)^{-1}(\mathbf{u}_{,1} \times \mathbf{X}^0_{,2} + \mathbf{X}^0_{,1} \times \mathbf{u}_{,2}).$$

Similar to the derivation of Equation (3), we can write the derivatives of $\Delta\mathbf{X}_{,3}$ as follows

$$\Delta\mathbf{X}_{,3i} = (\mathbf{J}^0)^{-1}(\mathbf{u}_{,1i} \times \mathbf{X}^0_{,2} + \mathbf{X}_{,1} \times \mathbf{X}^0_{,2i} \\ + \mathbf{X}^0_{,1i} \times \mathbf{u}_{,2} + \mathbf{X}^0_{,1} \times \mathbf{u}_{,2i}).$$

Finally, the bending strain can be expressed in terms of the displacement field as

$$\rho_{ij} = -\mathbf{u}_{,ij} \cdot \mathbf{X}^0_{,3} + (\mathbf{J}^0)^{-1}[\mathbf{u}_{,1} \cdot (\mathbf{X}^0_{,ij} \times \mathbf{X}^0_{,2}) \\ + \mathbf{u}_{,2} \cdot (\mathbf{X}^0_{,1} \times \mathbf{X}^0_{,ij})]. \quad (4)$$

Note that the derivation of the membrane strain is independent of the introduction of the Kirchhoff-Love hypothesis.

Given two homeomorphic point-set surfaces, we can measure the metric tensor and curvature tensor pointwisely on the two surfaces. Through the use of interpolation, we can obtain the tensor field for any intermediate time step $t \in [0, 1]$. The difference between the interpolated tensor field and the tensor field of the rest configuration gives us the strain field that characterizes the intrinsic deformation of the surface away from its rest, initial shape. We can then apply the linear membrane (1) and bending strain (4) which are functions of the displacement to minimize the following energy

$$\mathbf{W} = \frac{1}{2}\int_{\Omega}(\alpha||\epsilon(u)-\epsilon^t||^2+\beta||\rho(u)-\rho^t||^2)d\Omega, \tag{5}$$

where ϵ^t and ρ^t are the membrane and bending strain obtained by interpolation at time step t. In our examples, we use simple linear interpolation for the membrane and bending strain: $\epsilon^t = t \cdot (g^n - g^0)$ and $\rho^t = t \cdot (b^n - b^0)$. More sophisticated interpolation scheme can be used for smoother visual quality.

4 Meshless Techniques

Since the invention of the finite element method (FEM) in the 1950s, FEM has become the most popular and widely used method in engineering, scientific computing, and computer animation. In FEM, the individual elements are connected together by a topological map which is usually called a mesh. The finite element interpolation functions are then built upon the mesh, which ensures the compatibility of the interpolation. However, this procedure is not always advantageous, because the numerical compatibility condition is not the same as the physical compatibility condition of a continuum. For instance, in a Lagrangian type of computations, one may experience mesh distortion, which can either end the computation altogether or result in drastic deterioration of accuracy. Therefore, it would be computationally efficacious to discretize a continuum by only a set of nodal points without mesh constraints. In this paper, we utilize the Moving Least Squares (MLS) approximation method which is used in the Element Free Galerkin (EFG) method [23] to arrive at the numerical discretization.

4.1 MLS Approximation

The local approximation u^h of a field function $u(\mathbf{x})$ defined in a solution domain of arbitrary dimension, Ω, can be expressed as the inner product of a vector of the polynomial basis, $\mathbf{p}(\mathbf{x})$, and a vector of the coefficients, $\mathbf{a}(\mathbf{x})$

$$u^h(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{a}(\mathbf{x}) = \sum_{j=1}^{m} p_j(\mathbf{x})a_j(\mathbf{x}), \tag{6}$$

where m is the number of monomials in the polynomial basis. In our current work on 2-manifold surfaces, a linear basis, $\mathbf{p}^T = (1, x, y)$ corresponding to $m = 3$, is used. If the field values at a set of nodes, $\mathbf{x}_i$, $i = 1, ..., n$, are known a priori, the coefficient vector $\mathbf{a}(\mathbf{x})$ can be determined by minimizing a weighted, discrete L_2 error norm defined as:

$$L = \sum_{i=1}^{n} w(\mathbf{x}, \mathbf{x}_i)[u^h(\mathbf{x}_i) - u_i]^2, \tag{7}$$

where $w(\mathbf{x}, \mathbf{x}_i)$ is a weighting function defined over a compact support (also called the domain of influence of node i), u_i is the nodal value at $\mathbf{x}_i$, and n the number of nodes whose domain of influence contains the evaluation point $\mathbf{x}$. The stationary of L with respect to $\mathbf{a}(\mathbf{x})$ leads to the solution of $\mathbf{a}(\mathbf{x})$. Substitution of $\mathbf{a}(\mathbf{x})$ into Equation (6) gives

$$u^h(\mathbf{x}) = \sum_{i=1}^{n} \phi_i(\mathbf{x})u_i, \tag{8}$$

with $\phi_i(\mathbf{x})$ being the MLS shape function. More details on the derivation of the shape functions can be found in [24]. Figure 3 shows the simulation nodes scattered over a thin plane and their corresponding support regions.

4.2 Meshless Discretization

It now sets the stage for us to address the discretization issue of the energy defined in Equation (5) that we attempt to minimize. To make

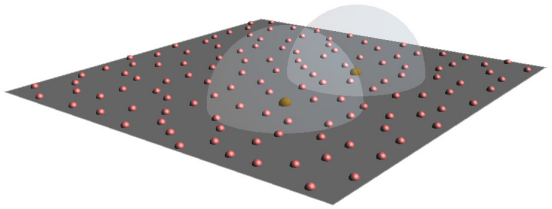


Figure 3: The smaller red balls represent the simulation nodes and the larger translucent white hemispheres represent the support radii of two of the simulation nodes

the mathematical formulation concise and consistent, we arrange the elements in 2×2 symmetric tensors into 3×1 vectors ϵ and ρ, for membrane and bending strain, respectively.

$$\epsilon = \begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ 2\epsilon_{12} \end{bmatrix}, \qquad \rho = \begin{bmatrix} \rho_{11} \\ \rho_{22} \\ 2\rho_{12} \end{bmatrix}.$$

Using this notation, the energy functional to be minimized can be reformulated as

$$\mathbf{W} = \int_\Omega \{\alpha[\epsilon(\mathbf{u}) - \epsilon^t]^T[\epsilon(\mathbf{u}) - \epsilon^t] + \beta[\rho(\mathbf{u}) - \rho^t]^T[\rho(\mathbf{u}) - \rho^t]\} d\Omega.$$

Setting the derivative with respect to $\mathbf{u}$ to be 0, we obtain

$$\frac{\partial \mathbf{W}}{\partial \mathbf{u}} = \int_\Omega \{\alpha \frac{\partial \epsilon(\mathbf{u})}{\partial \mathbf{u}}[\epsilon(\mathbf{u}) - \epsilon^t] + \beta \frac{\partial \rho(\mathbf{u})}{\partial \mathbf{u}}[\rho(\mathbf{u}) - \rho^t]\} d\Omega = 0.$$

Applying Equation (8), the displacement field can be approximated by the MLS shape functions $\phi_I(\theta_1, \theta_2)$ as

$$\mathbf{u}^h(\theta^1, \theta^2) = \sum_{I=1}^{N} \phi_I(\theta_1, \theta_2)\mathbf{u}_I, \qquad (9)$$

where $\mathbf{u}_I$ are the nodal displacement vectors and N is the number of simulation nodes.

Substituting Equation (9) into Equation (1) and Equation (4) gives the approximated membrane and bending strain in matrix form

$$\epsilon(\theta^1, \theta^2) = \sum_{I=1}^{N} \mathbf{M}^I(\theta^1, \theta^2)\mathbf{u}_I, \qquad (10)$$

$$\rho(\theta^1, \theta^2) = \sum_{I=1}^{N} \mathbf{B}^I(\theta^1, \theta^2)\mathbf{u}_I, \qquad (11)$$

where $\mathbf{M}^I$ and $\mathbf{B}^I$ are the membrane and bending strain matrices associated with simulation node I. Finally, introducing Equation (10) into Equation (11) yields the linear equation:

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \qquad (12)$$

where $\mathbf{K}$ is the stiffness matrix, and $\mathbf{u}$ is the nodal displacement vector and $\mathbf{f}$ is the virtual nodal force vector. The global stiffness matrix $\mathbf{K}$ is a block matrix which can be conveniently assembled by filling in low-level 3×3 stiffness matrices defined as

$$\mathbf{K}^{IJ} = \int_\Omega [\alpha(\mathbf{M}^I)^T\mathbf{M}^J + \beta(\mathbf{B}^I)^T\mathbf{B}^J] d\Omega.$$

And the virtual nodal force vector can be assembled similarly

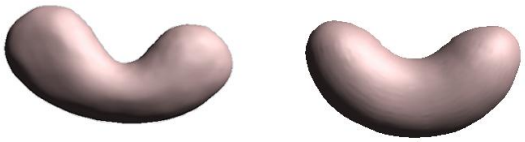
$$\mathbf{f}^I = \int_\Omega [\alpha(\mathbf{M}^I)^T\epsilon^I + \beta(\mathbf{B}^I)^T\rho^I] d\Omega.$$

The linear system in (12) can be solved using the bi-conjugate gradient method.

4.3 Local Parameterization

In previous discussion, we have assumed that a common analysis domain exists for both the source and target point-set surfaces, however, this may not be true in general. In fact, in order to obtain such a global analysis domain, global parameterization of the point-set surfaces is needed. In this paper, we adopt local parameterization to bypass the expensive global parameterization step, and the physical simulation can be directly performed on the point-based surfaces using the vicinity information only.

We proceed by defining, for every surface point $\mathbf{p}_i$, a neighborhood of surface points N_i. The neighborhood can be obtained by simply taking the K nearest neighboring points. In our implementation, we set K to be between 20 and 30. The local parameterization is defined on the local tangent space, which can



(a) (b)

Figure 4: Physically based morphing of a bending bar: (a) without and (b) with incrementally updating stiffness matrix.

be computed using principle component analysis (PCA). The two eigenvectors corresponding to the two largest eigenvalues span the tangent space at point $\mathbf{p}_i$. The coordinate of any point in the vicinity of $\mathbf{p}_i$ is then obtained by projecting the point onto its local tangent space.

In order to compute the strain matrices, partial derivatives of $\mathbf{X}$ up to the second order are needed. We compute the derivatives by minimizing an MLS error function. As an example, we consider the x-component of the position in the neighborhood of a point $\mathbf{p}_i$. By Taylor expansion, we can approximate the x-component of the surface position on the local parameter domain to first order as

$$\widetilde{x}(\mathbf{\Omega}(\mathbf{p}_i) + \Delta\mathbf{\Omega}) \approx x(\mathbf{\Omega}(\mathbf{p}_i)) + \nabla x|_{\mathbf{p}_i} \cdot \Delta\mathbf{\Omega},$$

where $\mathbf{\Omega}(\mathbf{p}_i)$ is the local parameter value of point $\mathbf{p}_i$. Therefore, we can estimate the derivative $\nabla x|_{\mathbf{p}_i}$, by minimizing the following weighted least square error:

$$\tilde{L} = \sum_{j \in N_i} \mathbf{w}_{ij} [\widetilde{x}(\mathbf{\Omega}(\mathbf{p}_j)) - x(\mathbf{\Omega}(\mathbf{p}_j))]^2.$$

Note that, these partial derivatives are only defined on the local parameter domain of $\mathbf{p}_i$ and therefore all coordinate values must be computed with respect to the local tangent space of $\mathbf{p}_i$. The second order partial derivatives of $\mathbf{p}_i$ can be obtained similarly after the first order partial derivatives for all neighboring points of $\mathbf{p}_i$ have been computed.

4.4 Incremental Update

Due to the fact that linear membrane and bending strains are used to minimize the energy functional (5), large deformation from the rest configuration can lead to distortion artifacts as shown in Figure 4(a). We correct this problem by incrementally updating the global stiffness matrix $\mathbf{K}$. Each updating step only recomputes the partial derivatives and tensors of the surface with respect to the fixed local parameter domain. This process of incrementally updating the stiffness matrix can be considered as using small linear steps to approximate the nonlinear deformation in its limit. The corrected shape is shown in Figure 4(b). The updating step size can be adaptively controlled by setting threshold of the residual energy after minimizing the energy functional $\mathbf{W}$. Once the residual energy exceeds the threshold, the stiffness matrix is dynamically updated. Our experiment shows that recomputing the stiffness matrix 10 times with uniform step length yields satisfactory results for most of our experiments.

4.5 Numerical Integration

Assembling the stiffness matrix $\mathbf{K}$ and the virtual nodal force $\mathbf{f}$ involves numerical integration over a global analysis domain. This integration is typically computed using Gaussian quadrature. Since our computation is performed on local parameter domain, such integration is infeasible. Alternatively, we approximate the integration as

$$\mathbf{K}^{IJ} = \sum_{i}^{NP} \mathbf{w}_i [\alpha (\mathbf{M}^I)^T \mathbf{M}^J + \beta (\mathbf{B}^I)^T \mathbf{B}^J],$$

$$\mathbf{f}^I = \sum_{i}^{NP} \mathbf{w}_i [\alpha (\mathbf{M}^I)^T \epsilon^I + \beta (\mathbf{B}^I)^T \rho^I],$$

where NP is the number of surface points and $\mathbf{w}_i$ is the surface area approximation associated with the surface point $\mathbf{p}_i$.

5 Implementation and Results

We have implemented our prototype point-based morphing system on a Microsoft Windows XP PC with Xeon 2.2 GHz CPU, 1.0GB RAM and an nVidia Quadro4 700 XGL GPU. The system is written in VC++ 6.0 and the renderer is built upon OpenGL. The algorithmic flow of our system is described in Figure 5. The computational

time for generating the intermediate morphing frames is documented in Table 1.

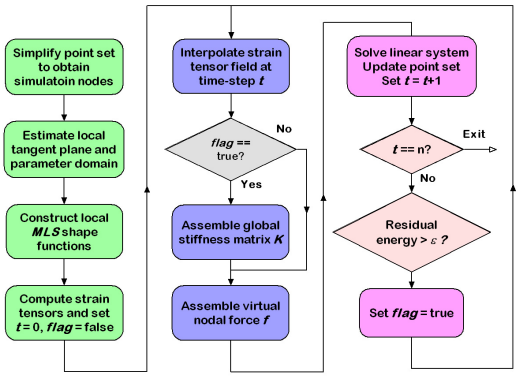


Figure 5: Algorithmic overview and flowchart.

Model	Surfels	Nodes	Frames	Time
bar	4000	512	100	3
torus	12000	1024	100	18
isis	10000	1024	100	11
rabbit	16760	1024	85	20
moai	10000	2048	100	65
sphere	10000	2048	100	70

Table 1: Model statistics and computational cost (seconds/frame)

Ideally, we would like to incorporate all the point samples on the surface as simulation nodes. However, this becomes numerically prohibitive for densely sampled point set in terms of both memory storage and computation time. For extremely dense point sets, we acquire the simulation nodes by simplifying the source surface. In our implementation, we use the adaptive hierarchical clustering method [25] to select a subset of the original surface sample points as the simulation nodes. The number of simulation nodes ranges from 500 to 2000 for the examples that we have demonstrated. Figure 6 shows different number of simulation nodes for the same point surface. Using the simplified point set as simulation nodes will undoubtedly lose many deformation details since the generated displacement field is relatively smooth based on MLS approximation. This problem can be tackled by using the multiresolution modeling framework of [26] to capture the deformation details. We can compute the offset distance δ between the

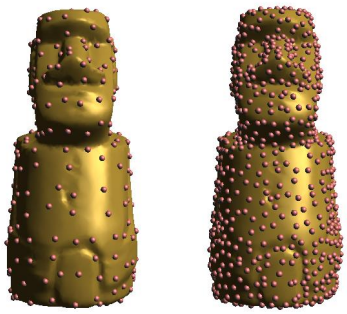


Figure 6: Different number of simulation nodes for the same point-set surface.

original point surface and the low-pass filtered surface based on simplified simulation nodes. For the source and target shape, we can compute their offset distance δ^0 and δ^n, respectively. The offset distance associated with each frame can be simply calculated via linear interpolation between δ^0 and δ^n. See Figures 7-11 for some morphing experiments that we have conducted.

One important characteristic of our system is that material properties can be associated with the surface by manipulating the weights associated with the two energy terms in Equation (5). In our implementation, we adopt the weights associated with isotropic materials:

$$\alpha = \frac{Eh}{1-v^2}, \qquad \beta = \frac{Eh^3}{12(1-v^2)},$$

where E is Young's Modulus and v is Poisson's Ratio. In fact, the user can either fine-tune the material property by setting appropriate values for the Young's Modulus and the Poisson's Ratio or manipulate the weights directly. Further extending our system to accommodate anisotropic properties can be easily achieved by making both α and β spatially-varying functions across the entire point-set surface during the morphing process.

6 Conclusion

In this paper, we have systematically developed an automatic and novel method to apply the dynamic, meshless thin-shell simulation principle to the physically-plausible morphing of point-based models. Our innovative framework is inspired and derived from differential geometry and continuum mechanics. The physically-

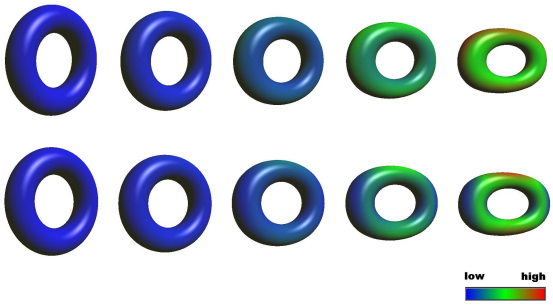


Figure 7: Temporal change of deformation energy when morphing a standard torus to a squeezed one. Upper row: stretching energy. Lower row: bending energy

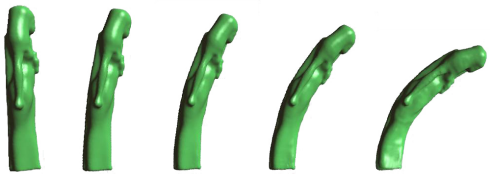


Figure 8: Morphing frames of a bowing isis statue.

plausible, natural morphing is the direct result of physics-based energy optimization, thus minimizing user intervention. Moreover, our method is based on local parameterization of the underlying point set surface, and therefore, is computationally efficient. In essence, our method is equivalent to solving a boundary-value PDE problem, where all the virtual forces are derived from the change of the intrinsic geometric measurement of the surface. Our experiments show that the proposed point surface morphing technique can generate physically-meaningful, convincing morphing sequences while avoiding most of the undesirable, mesh-based artifacts (such as triangle flip-over or point surface cracks).

Some immediate future work to further extend our framework's functionalities are as follows. First, we shall consider using the mature technique for multiresolution analysis and hierarchical decomposition to handle extremely complicated, large-scale, point-based objects. Third, we shall seek a practical solution on the problem of best possible correspondence between two point-set surfaces.

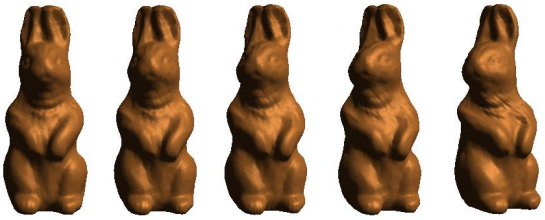


Figure 9: Morphing frames of a rabbit model twisting its head.

Figure 10: Morphing frames of an inflating moai model.

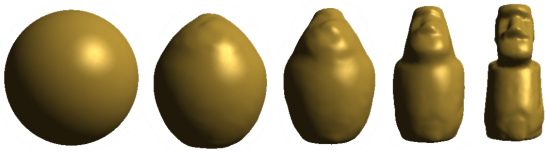


Figure 11: Morphing from sphere to moai.

Acknowledgements

This research work was partially supported by the NSF grant ACI-0328930, the ITR grant IIS-0326388, and Alfred P. Sloan Fellowship to Hong Qin. The Isis model and the rabbit model are courtesy of Cyberware Inc.

References

[1] C. Xiao, W. Zheng, Q. Peng, and A.R. Forrest. Robust morphing of point-sampled geometry. *Comput. Animat. and Virtual Worlds*, 15:201–210, 2004.

[2] F. Lazarus and A. Verroust. Three-dimensional metamorphosis: a survey. *The Visual Computer*, 14:373–389, 1998.

[3] M. Alexa. Recent advances in mesh morphing. *Computer Graphic Forum*, 21(2):173–196, 2002.

[4] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. *Ad-*

vances in Multiresolution for Geometric Modelling, pages 157–186, 2004.

[5] T. W. Sederberg, P. Gao, G. Wang, and H. Mu. 2-d shape blending: an intrinsic solution to the vertex path problem. In *SIGGRAPH 1993*, pages 15–18, 1993.

[6] M. Alexa, D. Cohen-Or, and D. Levin. As-rigid-as-possible shape interpolation. In *SIGGRAPH 2000*, pages 157–164, 2000.

[7] H.-B. Yan, S.-M. Hu, and R. Martin. Morphing based on strain field interpolation. *Comput. Animat. and Virtual Worlds*, 15:443–452, 2004.

[8] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *SIGGRAPH 1987*, pages 205–214, 1987.

[9] D. L. James and D. K. Pai. Artdefo: accurate real time deformable objects. In *SIGGRAPH 1999*, pages 65–72, 1999.

[10] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *SIGGRAPH 2001*, pages 31–36, 2001.

[11] D. Baraff and A. Witkin. Large steps in cloth simulation. In *SIGGRAPH 1998*, pages 43–54, 1998.

[12] Robert Bridson, Ronald Fedkiw, and John Anderson. Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH 2002*, pages 594–603, 2002.

[13] F. Cirak, M. Ortiz, and P. Schroder. Subdivision surfaces: a new paradigm for thin-shell finite element analysis. *Inter. J. Numer. Methods Eng.*, 47:2039–2072.

[14] E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schroder. Discrete shells. In *ACM SIGGRAPH/Eurographics Symp. Comput. animation*, pages 62–67, 2003.

[15] M. Zwicker, M. Pauly, O. Knoll, and M. Gross. Pointshop 3d: an interactive system for point-based surface editing. In *SIGGRAPH 2002*, pages 322–329, 2002.

[16] M. Pauly, R. Keiser, L. P. Kobbelt, and M. Gross. Shape modeling with point-sampled geometry. *ACM Trans. Graph.*, 22(3):641–650, 2003.

[17] X. Guo, J. Hua, and H. Qin. Scalar-function-driven editing on point set surfaces. *IEEE Comput. Graph. Appl.*, 24(4):43–52, 2004.

[18] M. Muller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. In *ACM SIGGRAPH/Eurographics Symp. Comput. animation*, pages 141–151, 2004.

[19] M. S. Floater and M. Reimers. Meshless parameterization and surface reconstruction. *Comput. Aided Geom. Design*, 18(2):77–92, 2001.

[20] M. Pauly, R. Keiser, B. Adams, P. Dutre, M. Gross, and L. J. Guibas. Meshless animation of fracturing solids. In *SIGGRAPH 2005 to appear*, 2005.

[21] X. Guo and H. Qin. Real-time meshless deformation. *to appear in Comput. Animat. and Virtual Worlds*, 2005.

[22] E. Kreyszig. *Differential Geometry*. Dover Publications, Incorporated, 1991.

[23] T. Belytschko, Y. Y. Lu, and L. Gu. Element free galerkin methods. *Inter. J. Numer. Methods Eng.*, 37:229–256, 1994.

[24] X. Guo and H. Qin. Point-based dynamic deformation and crack propagation. Technical report, Stony Brook University, October 2004.

[25] M. Pauly, M. Gross, and L. P. Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization '02*, pages 163–170, 2002.

[26] M. Pauly, M. Gross, and L. P. Kobbelt. Multiresolution modeling of point-sampled geometry. Technical report, ETH Zurich, 2002.