To appear in the ACM SIGGRAPH conference proceedings

Handle-Aware Isolines for Scalable Shape Editing

Oscar Kin-Chung Au Hongbo Fu Chiew-Lan Tai Daniel Cohen-Or

Hong Kong University of Science and Technology Tel Aviv University

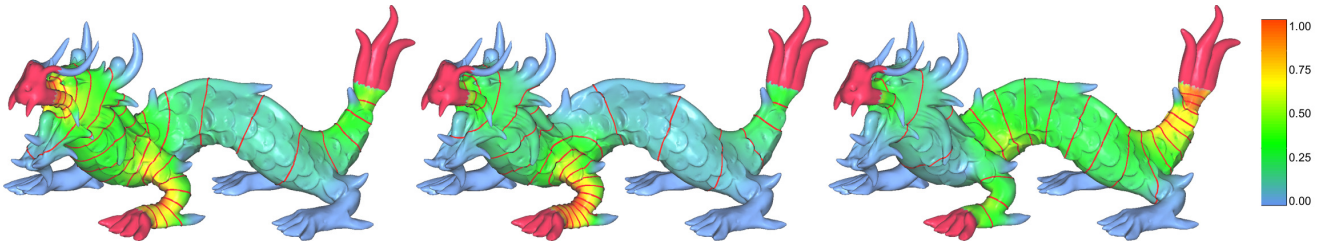

Figure 1: Our method uses handle-aware isolines to build a reduced model. The three images visualize the isolines and rigidity information both associated with the handles (in crimson) at the upper jaw (left), the left fore foot (middle), and the tail (right), respectively. Note that the isolines respect the handles and the shape geometry. We associate a transformation to each isoline and identify the transformations of isolines associated with all the handles as the reduced domain.

Abstract

Handle-based mesh deformation is essentially a nonlinear problem. To allow scalability, the original deformation problem can be approximately represented by a compact set of control variables. We show the direct relation between the locations of handles on the mesh and the local rigidity under deformation, and introduce the notion of *handle-aware rigidity*. Then, we present a reduced model whose control variables are intelligently distributed across the surface, respecting the rigidity information and the geometry. Specifically, for each handle, the control variables are the transformations of the isolines of a harmonic scalar field representing the deformation propagation from that handle. The isolines constitute a virtual skeletal structure similar to the bones in skinning deformation, thus correctly capturing the low-frequency shape deformation. To interpolate the transformations from the isolines to the original mesh, we design a method which is local, linear and geometry-dependent. This novel interpolation scheme and the transformation-based reduced domain allow each iteration of the nonlinear solver to be fully computed over the reduced domain. This makes the per-iteration cost dependent on only the number of isolines and enables compelling deformation of highly detailed shapes at interactive rates. In addition, we show how the handle-driven isolines provide an efficient means for deformation transfer without full shape correspondence.

Keywords: Scalable Shape Editing, Handle-Aware, Rigidity-Aware, Harmonic Fields, Isolines

1 Introduction

Handles have become a popular intuitive metaphor for differential surface editing [Sorkine et al. 2004; Yu et al. 2004]. Users manipulate the handles, and the transformations at the handles are propagated to the rest of the regions [Botsch and Kobbelt 2004]. 3D rotation transformations involved in the propagation are nonlinearly dependent on (unknown) vertex positions. Therefore handle-based deformation is essentially a nonlinear problem, generally requiring to be solved iteratively. Solving such a problem fast is challenging, especially for highly detailed models.

To allow scalability, reduced models approximately formulate the original deformation problem. Let $\mathbf{V} \in \Re^n$ denote the domain of the original problem, where n is the number of vertices. The aim is to find a reduced domain consisting of a compact set of control variables, $\mathbf{U} \in \Re^m$ ($m \ll n$), and an appropriate interpolation function $\mathbf{P} : \mathbf{U} \rightarrow \mathbf{V}$. The original problem is then projected to the reduced domain and represented in terms of $\mathbf{U}$. The reduction of the problem size leads to better performance in terms of both time complexity and memory cost [Huang et al. 2006]. Since the reduced size of the model necessarily incurs some degradation in the deformation quality, the choices of $\mathbf{U}$ and $\mathbf{P}$ are crucial to the design of an effective reduced model.

To achieve desirable deformation quality with as few control variables as possible, an efficient reduced model should distribute the control variables intelligently, respecting potential deformations and the given geometry. In example-based deformation, the potential deformations are learned from example shapes, and thus the control variables can be specifically chosen to attain the desirable deformation quality, alleviating redundancy of control variables [James and Twigg 2005; Der et al. 2006]. In contrast, in handle-based interactive editing, since the user is allowed to move the handles freely, it is challenging to determine the *a priori* degree of local deformation of the mesh to aid the construction of reduced models. We observe that the deformation propagation is along paths connecting the handles, resulting in features along the paths being deformed more significantly than those further away. Moreover, these underlying propagation paths are fixed once the handles are specified by the user, making the *relative rigidity* of features *deformation invariant*. For example, for the dinosaur model with handles placed at its head and feet (Figure 3), the arms and tail always behave very rigidly during editing. This fact motivates us to find a representation of the relative rigidity information and use it to guide the distribution of the control variables.

We choose to use transformations rather than vertex positions as control variables, since the former is a more natural and simpler way to represent deformations which consist of local transformations. Noticing that the combined deformation propagated from all the handles is complex (e.g., at the middle branching part of a 'Y'-shape mesh with handles at all its three ends), while the individual propagation field for each handle is simple and regular, we introduce a separate set of transformations to capture the deformations propagated from each handle, and blend them to represent the combined deformation from all the handles. We model the deformation influence caused by a handle's manipulation using a harmonic scalar field valued 1 at that handle, and 0 at all other handles. The vertices along an isoline of a harmonic field receive the same deformation influence from the associated handle. Therefore, we associate one transformation to each isoline and identify the transformations of the isolines of *all* the harmonic fields as the control variables. The handle-driven characteristic of each harmonic field allows us to sample it at equal parametric intervals to obtain an isoline set that respects the relative rigidity field (Figures 1 and 3).

In a sense, all the sampled isolines form generalized bones, like those employed in skinning deformation for interactive posing [Lewis et al. 2000]. Designing a function interpolating the isoline transformations to the deformed vertex positions is similar to the skinning process. Skinning deformations, however, require sufficient deformation examples or tedious manual painting of appropriate interpolation weights to achieve satisfactory results [Mohr and Gleicher 2003]. In contrast, since the isolines in our method provide a natural discrete parametric domain for the propagation field of each handle, they enable the design of a local, linear interpolation scheme that effectively relates the reduced domain and the original domain. Here the interpolation of the reduced model is surface-based rather than space-based [Huang et al. 2006], thus alleviating deformation artifacts.

Our transformation-based reduced domain coupled with the linear interpolation scheme allows effective encoding of local surface features, e.g., the differential coordinates in differential mesh editing. With such an encoding, our reduced model reduces each iteration of solving the original nonlinear problem to a transformation updating step whose time complexity is dependent on only the number of isolines (which depends on the number of handles), instead of the mesh resolution. After obtaining the converged transformations, we interpolate the transformations to compute the deformed positions of all vertices.

We apply our reduced model to perform differential mesh deformation. Experiments demonstrate that our method is scalable to deformations of very large models, with both faster per-iteration time and a smaller number of iterations for convergence. In addition, the handle-driven isolines provide a natural correspondence between two models, enabling the design of a simple and effective deformation transfer method without full surface correspondence.

2 Related Work

Differential Mesh Editing. Differential mesh editing is essentially a problem of surface reconstruction from differential coordinates representing local surface features [Sorkine et al. 2004]. This problem is nonlinear since the differential coordinates are nonlinearly dependent on the deformed vertex positions [Au et al. 2005; Au et al. 2006; Huang et al. 2006; Botsch et al. 2006].

For fast computation, most of the earlier work linearizes the problem by replacing the *implicit* nonlinear dependence with an *explicit* process of transformation propagation from handles [Yu et al. 2004; Lipman et al. 2004; Zayer et al. 2005; Zhou et al. 2005; Lipman et al. 2005; Shi et al. 2006; Sorkine 2006]. For each vertex, the relative degree of propagation from a handle is measured by either geodesic distances [Yu et al. 2004] or handle-driven harmonic fields [Zayer et al. 2005]. The latter characterizes deformation propagation *between handles* more naturally than the former [Lipman et al. 2005]. Our method employs the harmonic fields to *implicitly* define the transformations with respect to the deformed surface.

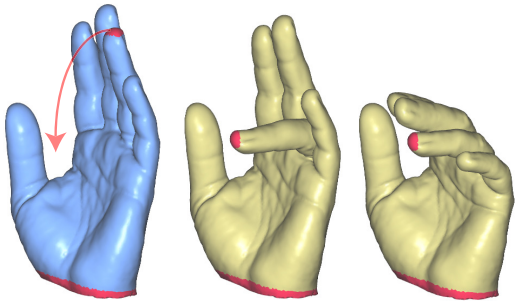

Figure 2: Manipulating the ring finger (left). Our surface-based method (middle) produces more intuitive deformation results than the space-based method proposed in [Huang et al. 2006] (right).

Linear approximation methods in general suffer from serious artifacts under large-scale deformation. More recently, several iterative frameworks to solve the original nonlinear deformation problem have been proposed, which are carefully designed to achieve interactive editing. Au et al. [2005] pre-compute the slow factorization of the system matrix and iteratively perform efficient updating of the differential coordinates during editing. Huang et al. [2006] present a general framework to incorporate the computationally efficient quasi-linear constraints as soft constraints, while allowing a small set of time-consuming non-quasi-linear constraints as hard constraints. Lipman et al. [2007] achieve computational efficiency by reducing the minimization of the changes of the second fundamental form to a Dirichlet-type functional optimization on a rotation field over the mesh.

Scalable Mesh Editing. Solving differential deformation in the original mesh domain is not scalable to large-scale models due to memory bottleneck, expensive per-iteration cost, and slow convergence. Huang et al. [2006] project the original nonlinear differential deformation problem into a subspace defined by a coarse base mesh surrounding the original mesh. The subspace projection greatly reduces the problem size, thus reducing per-iteration cost and making the factorization pre-computation memory affordable. They also demonstrate faster convergence for the projected problem. However, the per-iteration cost is still heavily dependent on mesh resolution.

Multiresolution mesh editing decomposes a mesh into a smooth base surface and a series of differences as local details (see [Kobbelt et al. 1998] and the references therein). Unlike reduced models, multiresolution methods do not involve any projections. Hence, they solve a new deformation problem over the base mesh, rather than a problem projected from the original unreduced domain. Instead of representing a mesh as a hierarchical level of details, Botsch et al. [2006] and Shi et al. [2006] employ a multigrid method to solve the nonlinear optimization problem directly defined over the original mesh.

Unlike our *handle-aware* reduced model, all the above methods which aim at achieving scalable shape editing are *handle-oblivious*. The method proposed by Huang et al. [2006] is most relevant to our

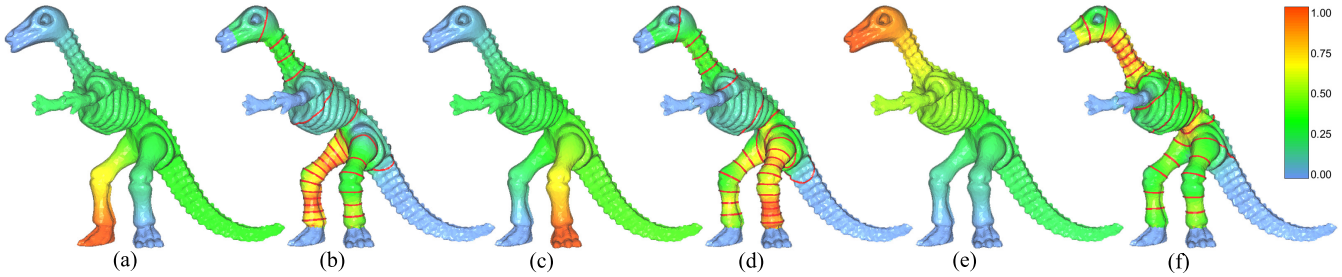

Figure 3: A dinosaur model with three handles. (a), (c), and (e) are the harmonic fields corresponding to the handles at the right foot, left foot and head, respectively. (b), (d), and (f) show the isolines and the relative rigidity information (gradient magnitude of harmonic field, with small values (in blue) meaning large rigidity) corresponding to the individual handles at the right foot, left foot and head, respectively. Note the sparse distribution of isolines at rigid regions (arms and tail).

reduced model. However, there are three main differences. First, due to the strong coherence within each isoline, our isoline-based representation needs fewer control variables than their vertex-based reduced model. Second, Huang et al. use the vertex positions of the base mesh as the control variables, making the projection of the differential coordinates to the reduced domain difficult. Third, unlike our geometry-dependent interpolation method, their interpolation method relating the subspace and the original full space is based on mean value interpolation [Ju et al. 2005], which is space-based. Therefore, like other space-based deformation methods [Sederberg and Parry 1986; Botsch and Kobbelt 2005], their method incorrectly assigns heavy influence to regions that are spatially close to, but geodesically far from, a manipulated handle (Figure 2).

3 Iterative Laplacian Mesh Editing

Our reduced model is general and can be applied to all the existing differential deformation frameworks (reviewed in Section 2) for performance improvement. However, for clear presentation, in this paper we introduce it within an iterative Laplacian editing framework [Au et al. 2005; Au et al. 2006; Huang et al. 2006], which we briefly review here.

The rationale of Laplacian mesh editing is to represent local features of a surface by the Laplacian coordinates [Lipman et al. 2004] and to reconstruct the deformed surface by minimizing the differences between the Laplacian coordinates before and after editing in a least-squares sense. Mathematically, Laplacian reconstruction is formulated as the following energy minimization

$$\arg\min_{\mathbf{X}} \|\mathbf{L}\mathbf{X} - \delta(\mathbf{X})\|^2, \tag{1}$$

where $\mathbf{L}$ is the Laplace operator matrix constructed from the original mesh before editing, and $\delta(\mathbf{X})$ is the Laplacian coordinates which are nonlinearly dependent on the deformed vertex positions $\mathbf{X} = (\mathbf{x}_1^\top, \mathbf{x}_2^\top, \ldots, \mathbf{x}_{n_{ver}}^\top)^\top$, $\mathbf{x}_i \in \Re^3$, with n_{ver} denoting the number of vertices in the edited mesh. During editing, the minimization problem is subject to the handle position constraint, usually enforced as a soft constraint [Sorkine et al. 2004; Huang et al. 2006]. The resulting Laplacian deformation is thus equivalent to solving the system $\mathbf{A}\mathbf{X} = \mathbf{b}(\mathbf{X})$ in a least-squares sense with

$$\mathbf{A} = \begin{bmatrix} \mathbf{L} \\ \omega\Phi \end{bmatrix} \text{ and } \mathbf{b}(\mathbf{X}) = \begin{bmatrix} \delta(\mathbf{X}) \\ \omega\mathbf{H} \end{bmatrix}, \tag{2}$$

where $\Phi\mathbf{X} = \mathbf{H}$ indicates the handle position constraint and ω is a large constant enforcing the soft constraint ($\omega = 1000$ in our experiments).

The nonlinear system in (2) is solved using an inexact Gauss-Newton method by iteratively updating the vertex positions

$$\mathbf{X}^{t+1} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}(\mathbf{X}^t), \tag{3}$$

where t is the time step. $\mathbf{X}^0$ is set as the current edited vertex positions to ensure that the vertex positions are updated smoothly. Each iteration basically involves two steps, the computation of $\mathbf{b}^{t+1} = \mathbf{b}(\mathbf{X}^t)$ and the solving of the linear system $\mathbf{A}^\top\mathbf{A}\mathbf{X}^{t+1} = \mathbf{A}^\top\mathbf{b}^{t+1}$. During deformation, $\mathbf{A}$ remains unchanged. Therefore, the time complexity of each iteration can be greatly reduced by precomputing the Cholesky factorization of $\mathbf{A}^\top\mathbf{A}$.

4 Handle-Aware Reduced Model

In this section, we introduce our handle-aware reduced model. The control variables are the transformations of isolines of harmonic fields, respecting the handle-aware rigidity information. We represent both the vertex positions and the Laplacian coordinates in terms of the isoline transformations to allow solving of the deformation propagation problem completely in the reduced domain.

4.1 Handle-Aware Rigidity and Isoline Construction

We use a set of harmonic fields to identify the handle-aware rigidity information over the mesh. For each handle i, we compute its corresponding harmonic field φ_i by solving the Laplace equation $\mathbf{L}\varphi_i = 0$, with the boundary values of its vertices set at one, and the values of the vertices of all other handles set at zero. Each resulting harmonic field reflects how the movement of the associated handle influences the deformation at each vertex of the mesh; large harmonic values imply significant deformation. From the harmonic field φ_i of each handle i, we identify the gradient magnitude of φ_i at each vertex as a rigidity field, i.e., $\psi_i = \|\nabla\varphi_i\|$. Small gradient magnitude implies large rigidity. Figure 3 shows three harmonic fields, each associated with a handle, located at the head and the two feet of the dinosaur model. The free protruding regions (i.e., tail and arms) have small gradient magnitudes in all the three harmonic fields, reflecting that these regions behave very rigidly during deformation.

For visualization and comparison (see Section 5), we define the combined rigidity field over the mesh as $\psi = \sqrt{\sum_i \psi_i^2}$ (see left image of Figure 4). We compare this combined rigidity field with the actual rigidity during deformation (produced by the iterative unreduced method). The actual rigidity field is defined as the errors deviating from local rigidity motion, that is, the residual $\mathbf{E} = \mathbf{A}\mathbf{X} - \mathbf{b}(\mathbf{X})$ for each vertex. The compatibility of these two

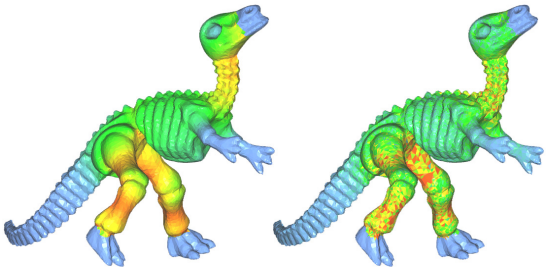

Figure 4: Our defined rigidity field (left) is compatible with the actual rigidity field (right).

fields, as shown in Figure 4, serves as evidence supporting our rigidity field definition. Note that, in all the images, ψ_i, ψ and $\mathbf{E}$ are normalized for visualization purposes.

All the vertices on an isoline receive the same degree of transformation propagation from the associated handle [Zayer et al. 2005]. Therefore, we sample the harmonic fields to get a set of isolines (see Figure 3) and use the transformations associated with the isolines to approximate the deformation field induced by the movement of that handle. Employing harmonic values to interpolate transformations at handles is not a new idea. Zayer et al. [2005] have used them to perform explicit interpolation without respecting the deformed surface, thus suffering from the translation-insensitivity problem [Au et al. 2005; Botsch et al. 2006; Fu et al. 2007]. Our method implicitly solves for the transformations dependent on the deformed surface through the optimization. Since more rigid regions correspond to smaller gradient magnitudes (i.e., smaller variations of harmonic values), uniformly distributing the isovalues automatically leads to sparser distribution of isolines in more rigid regions. This effect is illustrated around the tail and arms of the dinosaur in Figure 3. Note that the sampled isolines generally do not contain vertices or edges of the original mesh, and thus are independent of the mesh sampling.

We identify the following transformation set

$$\{\mathbf{r}_{i,j} \,|1 \le i \le n_{han}, 1 \le j \le n_{iso}\} \tag{4}$$

as the control variables of the reduced deformable model, where $\mathbf{r}_{i,j}$ is a 3×4 matrix associated with the j-th isoline of handle i, with isovalue $s_{i,j} = (j-1)/(n_{iso}-1)$; n_{han} is the number of user-specified handles, and n_{iso} is the number of isolines for each handle. Note that each isoline may consist of a single loop and multiple loops, each of which may have an arbitrary shape. As the isolines provide a natural parameterization of the propagation fields, it allows treating each isoline, possibly consisting of multiple loops, as one rigid component. For each handle, we include two special isolines: one isoline with the isovalue equal to 1 at that handle and one isoline with the isovalue equal to 0 at all the other handles. When only two handles are selected, since both isoline sets are exactly the same, we use only one set of isolines. The approximation precision of the reduced model is controlled by the number of isolines n_{iso}. Clearly, more isolines lead to more precise approximation.

The set of all isolines essentially serve as generalized bones, like those used in skinning deformation. Hence, the isoline transformations should reflect smooth low-frequency shape deformation. To enforce the smoothness of deformation, we require the transformations associated with neighboring isolines to be as similar to each other as possible:

$$\mathbf{r}_{i,j} - \mathbf{r}_{i,j+1} = \mathbf{0}, \quad 1 \le i \le n_{han}, 1 \le j \le n_{iso}-1. \tag{5}$$

To incorporate this smoothness term into the Laplacian reconstruction optimization in (1), we need to represent the vertex positions in terms of the transformations of isolines (see the details in the next subsection).

4.2 Vertex Position Interpolation

In this subsection, we describe how to interpolate the transformations of the isolines to compute the vertex positions of the original mesh. Similar to skinning deformation, we aim to compute a transformation for each vertex by interpolating the transformation set defined in (4), to transform the unedited position of that vertex to the edited position. We achieve this by first interpolating the transformations of isolines associated with each handle and then averaging the interpolated transformations among all the handles.

We compute the (deformation) transformation $\mathbf{r}_i^{(k)}$ of vertex k caused by handle i by linearly interpolating the transformations of the neighboring isolines:

$$\mathbf{r}_i^{(k)} = (1-\alpha)\mathbf{r}_{i,j} + \alpha\mathbf{r}_{i,j+1}, \text{ with } \alpha = \frac{\varphi_{i,k} - s_{i,j}}{s_{i,j+1} - s_{i,j}},$$

where $\varphi_{i,k}$ is the harmonic value associated with handle i at vertex k, and j is the index of the isoline with isovalue $s_{i,j}$ such that $s_{i,j} \le \varphi_{i,k} \le s_{i,j+1}$.

As larger (smaller) $\varphi_{i,k}$ means that the vertex k is parametrically closer to (farther away from) the corresponding handle i, it is a natural choice to use $\varphi_{i,k}$ as the weights to interpolate the transformations $\mathbf{r}_i^{(k)}$ of all the handles: $\mathbf{r}^{(k)} = \sum_i \varphi_{i,k}\mathbf{r}_i^{(k)}$ [1]. The final deformed vertex position is computed as $\mathbf{x}_k = \mathbf{r}^{(k)}\overline{\mathbf{x}}_k$, where $\overline{\mathbf{x}}_k$ is the original position of vertex k represented as homogeneous coordinates.

Note that the above transformation interpolations for individual handles (to compute $\mathbf{r}_i^{(k)}$) and for all the handles (to compute $\mathbf{r}^{(k)}$) are both linear, therefore we can represent the interpolation from the control variables to the vertex positions as $\mathbf{X} = \mathbf{W}_x\mathbf{R}_x$, where $\mathbf{R}_x$ is a column vector constructed from the transformation set $\{\mathbf{r}_{i,j}\}$ and $\mathbf{W}_x$ is a matrix constructed from $\{\overline{\mathbf{x}}_i\}$ and $\{\varphi_i\}$.

Handles that are specified on highly detailed models may contain a large number of vertices, making the enforcement of the position constraint computationally expensive. As the movement of a handle has a very low degree of freedom (usually involving only translation, rotation and uniform scaling), we adopt the method proposed in [Botsch and Kobbelt 2004] to represent a handle containing many vertices by four affine-independent vertices. The handle position constraint in Equation 2 can then be reformulated as $\Phi\mathbf{X} = \mathbf{H} = \mathbf{QC}$, where $\mathbf{Q}$ is a matrix relating the representative handle vertex positions $\mathbf{C}$ with all the handle vertex positions $\mathbf{H}$.

By replacing $\mathbf{X}$ with $\mathbf{W}_x\mathbf{R}_x$, the minimization problem in (1) enforced with the smoothness term in (5) is projected to the domain of the reduced model, rewritten as

$$\arg\min_{\mathbf{R}_x} \|\mathbf{L}\mathbf{W}_x\mathbf{R}_x - \delta(\mathbf{W}_x\mathbf{R}_x)\|^2 + \beta^2\|\mathbf{M}\mathbf{R}_x\|^2,$$

where $\mathbf{M}$ is a coefficient matrix constructed from (5) and β is a weight guiding the optimization ($\beta = 0.01$ in our experiments). The iterative updating in (3) becomes

$$\mathbf{R}_x^{t+1} = (\mathbf{U}^\top\mathbf{U})^{-1}[\mathbf{W}_x^\top\mathbf{L}^\top\delta(\mathbf{W}_x\mathbf{R}_x^t) + \omega^2\mathbf{W}_x^\top\Phi^\top\mathbf{QC}], \tag{6}$$

$$\mathbf{X}^{t+1} = \mathbf{W}_x\mathbf{R}_x^{t+1}, \tag{7}$$

[1] Note that, since the boundary conditions for all the harmonic fields satisfy a partition of unity, the principle of superposition for linear partial differential equations guarantees $\sum_i \varphi_{i,k} \equiv 1$.

where $\mathbf{U} = [\mathbf{W}_x^\top \mathbf{L}^\top \quad \omega \mathbf{W}_x^\top \Phi^\top \quad \beta \mathbf{M}^\top]^\top$. Since the definition of $\delta(\mathbf{X})$ is separable in the dimension of the vertices [Huang et al. 2006; Au et al. 2006], the above updating is performed separately for x, y, z coordinates of the deformed vertices. Subsequent discussion refers to the separate systems; however, for convenience, we continue to use the original symbols. Note that the size of matrix $\mathbf{W}_x$ is $n_{ver} \times 4n_{han}n_{iso}$. Hence, the original system matrix $\mathbf{A}^\top\mathbf{A}$ of size $n_{ver} \times n_{ver}$ is reduced to $\mathbf{U}^\top\mathbf{U}$ of size $4n_{han}n_{iso} \times 4n_{han}n_{iso}$ ($4n_{han}n_{iso} \ll n_{ver}$ for most cases) after model projection, speeding up the updating step significantly.

4.3 Laplacian Coordinate Interpolation

In (6), the evaluation of $\mathbf{R}_x^{t+1}$ is dependent on all the vertex positions $\mathbf{X}^t$ at time t (to compute $\delta(\mathbf{W}_x\mathbf{R}_x^t)$). This means that the updating of $\mathbf{X}^{t+1}$ and the updating of $\mathbf{R}_x^{t+1}$ are interdependent. Therefore, each iteration involves both updating steps, and thus is still computationally very expensive for large models (see evaluation details in Section 5). The reduced model proposed in [Huang et al. 2006] suffers from a similar problem. In this subsection, we present an approximation method which makes the computation of $\mathbf{R}_x^{t+1}$ only requiring the updating of a subset of triangles crossed by the isolines, instead of the whole mesh.

Similar to the vertex position interpolation, we aim to represent the Laplacian coordinates $\delta(\mathbf{X})$ based on the transformations of the isolines. We found that updating only a subset of the triangles crossed by the isolines sufficiently leads to a stable system. Thus, we uniformly sample the triangles crossed by the isolines such that the number of the sampled triangles is below a prescribed upper bound.

Let $\Gamma_{i,j}$ denote the set of the sampled triangles for the j-th isoline of handle i and $n_{tri} = \sum_{i,j} |\Gamma_{i,j}|$. Since the Laplacian coordinates approximate the curvature normals, we compute the isoline transformations using the normal fields of $\Gamma_{i,j}$. Specifically, we compute the local transformation $\widehat{\mathbf{r}}_{i,j}$ of the j-th isoline (without the translational part, thus represented by a 3×3 matrix) by solving the following system

$$\widehat{\mathbf{r}}_{i,j}\overline{\mathbf{n}}_k = \mathbf{n}_k, \quad \text{for} \quad k \in \Gamma_{i,j} \tag{8}$$

in a least-squares sense, where $\overline{\mathbf{n}}_k$ and $\mathbf{n}_k$ are the normals of triangle k before and after deformation, respectively. To better measure the contribution of triangle k in computing $\widehat{\mathbf{r}}_{i,j}$, we weigh the equations in (8) with the segment length l_k of the isoline crossing triangle k. By converting the above equations into the normal equations, the solution of $\widehat{\mathbf{r}}_{i,j}$ can be computed as follows:

$$\widehat{\mathbf{r}}_{i,j} = \Big(\sum_{k\in\Gamma_{i,j}} l_k^2(\mathbf{n}_k \otimes \overline{\mathbf{n}}_k)\Big)\Big(\sum_{k\in\Gamma_{i,j}} l_k^2(\overline{\mathbf{n}}_k \otimes \overline{\mathbf{n}}_k)\Big)^{-1}, \tag{9}$$

where $\otimes$ denotes the outer product.

The Laplacian coordinates in terms of the transformations $\{\widehat{\mathbf{r}}_{i,j}\}$ can then be expressed in matrix form $\delta = \mathbf{W}_\delta\mathbf{R}_\delta$, where $\mathbf{R}_\delta$ is a column vector constructed from $\{\widehat{\mathbf{r}}_{i,j}\}$ and $\mathbf{W}_\delta$ is a $n_{ver} \times 3n_{han}n_{iso}$ matrix constructed from the Laplacian coordinates and the harmonic values $\{\varphi_i\}$ both defined over the original mesh.

By replacing $\delta(\mathbf{W}\mathbf{R}_x)$ with $\mathbf{W}_\delta\mathbf{R}_\delta$, (6) then becomes

$$\mathbf{R}_x^{t+1} = (\mathbf{U}^\top\mathbf{U})^{-1}[(\mathbf{W}_x^\top\mathbf{L}^\top\mathbf{W}_\delta)\mathbf{R}_\delta^t + (\omega^2\mathbf{W}_x^\top\Phi^\top\mathbf{Q})\mathbf{C}]. \tag{10}$$

Since both $\mathbf{W}_x^\top\mathbf{L}^\top$ and $\mathbf{W}_\delta$ are fixed during updating, and computing $\mathbf{R}_\delta^t$ only involves a constant number of triangles crossed by the isolines, pre-computing $\mathbf{W}_x^\top\mathbf{L}^\top\mathbf{W}_\delta$ makes the above updating of $\mathbf{R}_x^{t+1}$ dependent only on the number of the crossed triangles, n_{tri}, (i.e., independent of mesh size), thus leading to faster updating of each iteration. All the deformed vertex positions are computed after the updating of $\mathbf{R}_x^{t+1}$ in (10) converges.

Figure 5: Our reduced model (middle: with only vertex position interpolation, right: with interpolation of both vertex positions and Laplacian coordinates) and the original unreduced model (left) both produce visually natural deformation results, with only subtle differences in low-frequency geometry. The isolines in the middle and right images are associated with the handles at the right hand and the left foot, respectively.

In comparison, Huang et al. [2006] use the vertex positions of a simplified base mesh as the control variables. Since it is hard to represent the Laplacian coordinates in terms of position-based control variables, it is unclear whether their method can be extended to achieve resolution-independent deformation.

Besides making the updating of each iteration independent of mesh size, another advantage of projecting both the vertex positions and the Laplacian coordinates to the reduced domain is that the propagation of transformation (deformation) is performed completely in the reduced domain. This means that the low-frequency deformation propagation is applied over the reduced domain, leading to faster convergence. We discuss this further in the following section.

5 Interactive Mesh Deformation

In this section, we analyze the results of applying our reduced model to interactive mesh deformation. To edit a mesh, the user specifies and manipulates either point or region handles [Au et al. 2005; Botsch et al. 2006]. Region handles have six degrees of freedom (translations and rotations); point handles have three degrees of freedom (translations), and their orientations are computed by the optimization. The user prescribes n_{iso} and an upper bound of

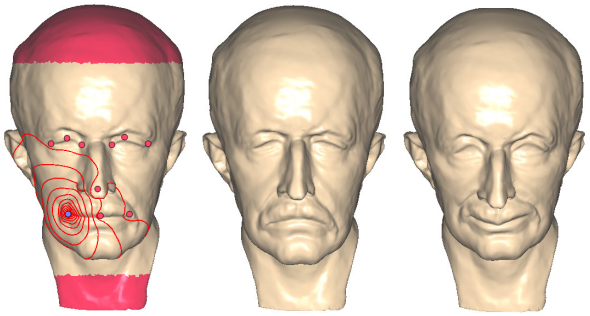

Figure 6: Changing facial expressions of the Max Planck model. Left: the original model with two region handles and ten point handles. The isolines displayed are associated with the blue point handle. Middle: sad mood. Right: happy mood.

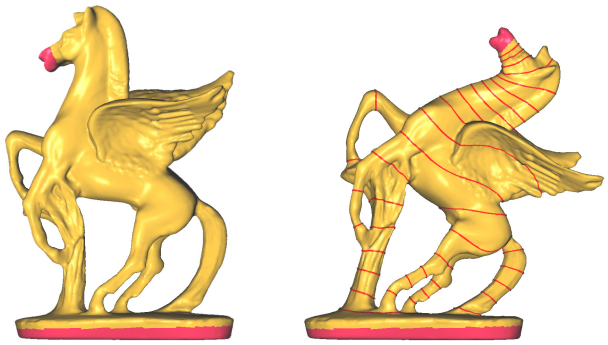

Figure 7: Our method works well for models with high genus and complex isoline components. Left: input model. Right: deformation result and isolines associated with the handle at mouth.

n_{tri}. The system pre-computes $\mathbf{W}_x^\top \mathbf{L}^\top \mathbf{W}_\delta$ and $\omega^2 \mathbf{W}_x^\top \Phi^\top \mathbf{Q}$ as well as the Cholesky factorization of $\mathbf{U}^\top \mathbf{U}$ in (10). Whenever the user specifies a new set of handles to continue editing, a new deformation system is set for the new handles based on the most recently edited result.

The isolines serving as a virtual skeletal structure also allow interactive posing. Figures 5 and 13 (top) show a variety of natural poses designed by the user using only a few handles. Sampling the deformation propagation fields as isolines according to the handle-aware rigidity information allows us to use only a small set of isolines ($n_{iso} = 20$) to approximate the low-frequency geometry correctly. Note that the high-frequency details are well preserved by the Laplacian coordinates.

Our method also supports local editing. In Figure 6, we locally deform the Max Planck model to change his facial expressions. This example also illustrates the ease of editing using point handles. Note the natural orientations automatically found through optimization at the corners of the mouth.

Figure 7 shows that our method works well for models with high genus. Such models generally have isolines of very complex components. This example also demonstrates that our choice of treating multiple loops of an isoline as one rigid component is effective.

Our reduced model is dependent on the handle locations and the mesh geometry, rather than mesh sampling. Given two meshes of different sampling densities of the same model, our deformation tool produces visually identical deformations of the model when manipulated with the same set of handles and the same number of isolines. The deformation of the Asian dragon shown in Figure 8 demonstrates this fact.

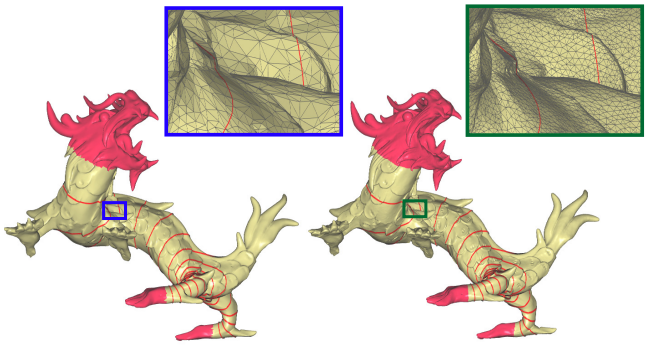

Figure 8: Visually identical deformation results of the Asian dragon model of different mesh sampling (left: $n_{ver} = 50K$, right: $n_{ver} = 250K$).

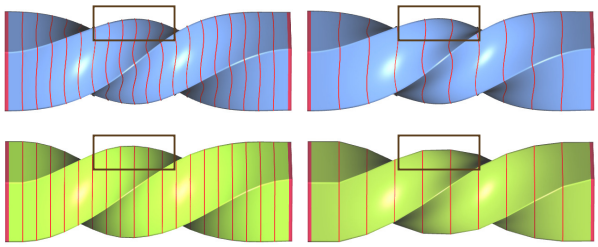

Figure 9: Twisting a horizontal bar. Top row: with original unreduced method (isolines drawn only for comparison purposes). Bottom row: with both vertex position and Laplacian coordinate interpolation. Note the linear effect of the reduced model, more obvious when fewer isolines are used (left: $n_{iso} = 20$, right: $n_{iso} = 10$).

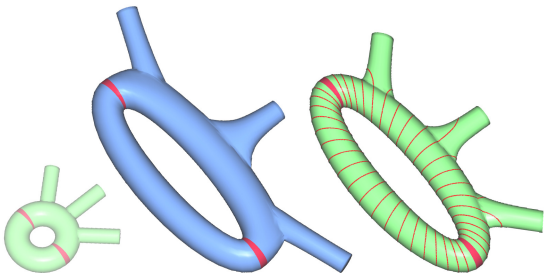

Figure 10: Our reduced model can preserve low-frequency information better than the unreduced model. Left: input model. Middle: result with the original unreduced model. Right: result with our reduced model.

Clearly, the reduced model suffers from approximation errors, due to both the vertex position interpolation and Laplacian coordinate interpolation. In Figure 9, we show that the resolution of the reduced model (i.e., the number of isolines) affects the deformation quality. The bar model is twisted by 175 degrees, and there is a larger linear effect when fewer isolines are used. From our experience, 20 isolines are usually sufficient to generate acceptable results for most models. Figure 5 compares the results of the reduced model and unreduced model. All the deformation results are visually natural, with only subtle differences in low-frequency geometry. In fact, in certain deformation scenarios, our reduced method yields better deformation quality than the original unreduced method, as demonstrated in Figure 10. Since the differential coordinates only record the (local) high-frequency details, differential-based deformation cannot preserve well the low-frequency geometry (global shape of the torus which determines the orientations of the bars). In contrast, our reduced model can better maintain low-frequency information, due to the strong coherence within each isoline.

Performance Analysis. The overall timing performance of iterative Laplacian deformation depends on two factors: the computational cost of each iteration, t_{iter}, and the number of iterations required to achieve convergence, n_{iter}. We demonstrate that our reduced model greatly reduces both t_{iter} and n_{iter}.

Table 1 gives the mesh statistics and detailed timing of our experiments, measured on a 3.2 GHz Pentium 4 PC with 3GB of RAM. Note that with only the vertex position interpolation, our reduced method already greatly improves the performance. However, since

Model	n_{ver}	n_{han}	n_{iso}	n_{tri}	t_0	t_1	t_2
Armadillo	172974	4	20	2486	2675.9	987.4	**10.9**
Bar	28802	2	20	570	335.1	34.8	**6.7**
Bar	28802	2	10	570	339.2	31.7	**6.4**
Buddha	959420	2	20	624	n/a	1455.7	**16.2**
Buddha	1400051	2	20	624	n/a	2220.4	**20.0**
Dinosaur	14050	3	20	1655	118.6	26.3	**5.8**
Dragon	50002	3	20	1758	472.0	141.9	**6.2**
Dragon	249677	3	20	1830	3518.5	667.8	**7.3**
Dragon	249677	4	20	2461	3515.2	1088.4	**10.2**
Dragon	984346	3	20	1837	n/a	1952.3	**19.4**
Hand	24795	2	20	615	292.4	31.2	**6.5**
Max Planck	49889	12	20	4210	625.1	306.6	**102.3**
Pegasus	63518	2	20	631	857.9	129.6	**10.8**
Torus	2747	2	20	483	27.3	3.7	**1.2**

Table 1: Columns t_0, t_1 and t_2 show the performance comparison between the unreduced method, and the reduced model with only vertex position interpolation, and with interpolation of both the vertex positions and the Laplacian coordinates, respectively. All timings are per-iteration updating costs (in milliseconds), with the display time excluded for fair comparison.

the algorithm updates the transformations $\mathbf{R}_x$ and the vertex positions $\mathbf{X}$ of the whole mesh alternately, per-iteration cost t_{iter} is still heavily dependent on the mesh resolution, making interactive editing of extremely large models prohibitive. By representing both the vertex positions and the Laplacian coordinates in terms of the control transformations, we make t_{iter} largely dependent on only the number of sampled triangles defining the transformations of the isolines, n_{tri}, independent of mesh resolution. Due to the limited CPU cache, t_{iter} may fluctuate slightly among models of similar n_{han}, n_{iso}, and n_{tri} but of different sizes.

The graph in Figure 11 compares the convergence rate of the original unreduced method and our reduced model for two specific examples, the Armadillo in Figure 5 and the dinosaur in Figure 12. As the two models have free protruding regions (the head and tail in the Armadillo, and the arms and tail in the dinosaur), causing higher nonlinearity of $\mathbf{b}(\mathbf{X})$, the unreduced method needs a large n_{iter} to converge. It is observed that projecting only the vertex positions to the reduced domain already makes the convergence rate much faster. With the interpolation of both the vertex positions and the Laplacian coordinates, the propagation of deformation is done completely in the reduced domain, thus further reducing n_{iter} significantly. For all the examples shown in this paper, the average

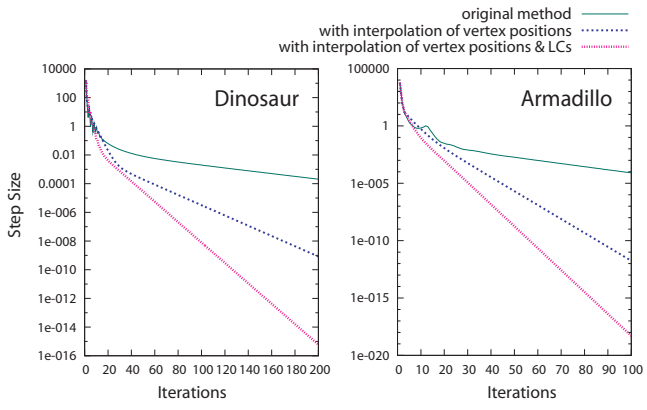

Figure 11: Convergence comparisons for the Armadillo and dinosaur deformation examples in Figures 5 and 12, respectively. The y-axis represents the step size in the logarithmic scale.

n_{iter} is less than 20.

Implementation and Discussions. The computation of the handle-driven harmonic fields is the bottleneck of our system, in terms of both time complexity and memory cost. Fortunately, this costly computation is done only at the initialization stage whenever the handles are set, rather than during interactive editing. Our initial implementation is based on a direct solver [Toledo 2003]. However, the memory cost of factorizing the system matrices constructed over the original mesh domain makes our unoptimized implementation prohibitive for models more than 500K vertices with 4 handles. To alleviate the problem, we resort to a multigrid method [Aksoylu et al. 2005]. Since the multigrid method has linear memory cost, theoretically the memory cost of our reduced model is only $O(n_{han}n_{ver})$. With the multigrid implementation, our system can handle larger models (see Table 1). Like most differential mesh editing frameworks, our system suffers from the problem of slow initialization of a large-scale system. The multigrid implementation also alleviates this problem. For example, it takes about 11 seconds to initialize the system for the dragon model with 250K vertices and three handles, while a direct solver requires 90 seconds.

As discussed above, we compute only a small subset of the vertex positions to update the transformations of the isolines. However, to visualize the deformation of the entire model during convergence we display a coarse version of the mesh (see the accompanying video). The coarse mesh is obtained by QEM simplification prior to interactive editing, and the positions of the vertices are also computed through Equation 7 as they are a subset of the original set of vertices. An alternative solution is to update the display of the whole mesh, say, every 30 transformation updating iterations. This functionality can be greatly accelerated using a GPU-based technique similar to matrix palette skinning [Lindholm et al. 2001].

A straightforward handle-driven reduced model would be one that uses vertices to summarize both mesh geometry and the derived rigidity information. We compare our isoline-based reduced model with such a vertex-based reduced model. We obtain the support domain of a rigidity-aware vertex-based reduced model by employing rigidity-aware mesh simplification, which is a variation of

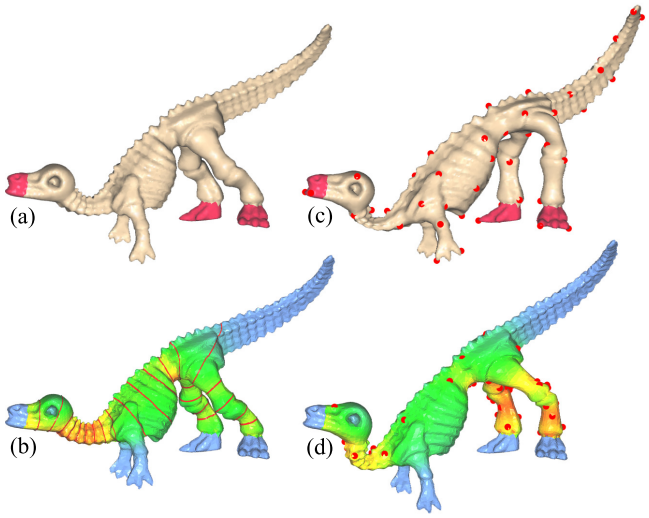

Figure 12: (a) original unreduced model. (b) isoline-based reduced model (the isolines and rigidity field shown are associated with the handle at the head). (c) rigidity-oblivious vertex-based reduced model. (d) rigidity-aware vertex-based reduced model (with the combined rigidity field visualized).

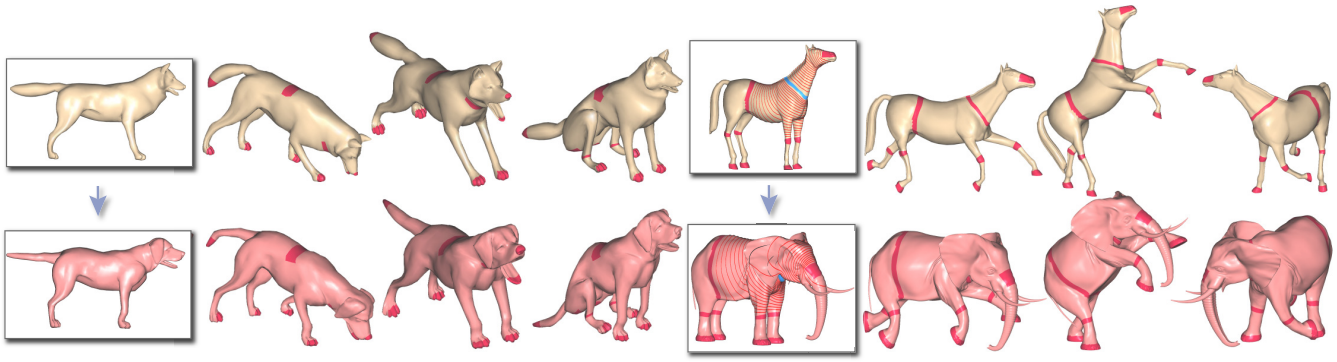

Figure 13: Examples of deformation transfer based on handle correspondence, instead of full triangle correspondence (from wolf to dog, from horse to elephant).

QEM [Garland and Heckbert 1997] with errors guided by the rigidity field ψ. We then associate a transformation with each vertex in the simplified mesh. Like the harmonic field for each handle, we define a harmonic field for each vertex to relate the reduced domain and the original domain. For comparison, we have also implemented a rigidity-oblivious vertex-based reduced model, with the vertices obtained through standard QEM.

Figure 12 compares isoline-based and vertex-based reduced models under the same number ($= 60 \times 12$) of control variables. As expected, the rigidity-oblivious vertex-based reduced model produces the worst result due to the waste of precious control variables at rigid regions. Our isoline-based reduced model generates better deformation quality than the rigidity-aware vertex-based reduced model, because each isoline influences a larger region, thus better preserving the shapes. The strong coherence within each isoline also enables our reduced model to use much fewer control variables. In addition, it is noteworthy that vertex-based methods are much slower (in terms of per-iteration cost) than our isoline method because the interpolation of the former is global, leading to much denser system matrices. We note that our choice of using one set of isolines for every handle might not be optimal. However, we found it hard to design alternative representations with fewer control variables to capture the complex combined transformation influence from all the handles.

We adopted the cotangent weighting scheme to define the discrete Laplace operator [Au et al. 2005]. Since the domain of the discrete Laplace operator is the 1-ring neighborhood of a vertex, harmonic fields may contain values outside the range $[0,1]$ near the boundary condition, especially when point handles or curve handles (rather than region handles) are used. However, the overflow of these out-of-range values is relatively very small (e.g., $-5e^{-17}$), and its influence on the isoline sampling and the final deformation is visually insignificant. Therefore, we only sample the harmonic field within the range $[0,1]$ to capture the influence of a handle.

Limitations. As both the time complexity and the memory cost of our reduced model is linear in n_{han}, our method achieves less gain for editing scenarios with a large number of handles. Furthermore, in some special editing scenarios, the isoline subspace does not capture well the essence of the true transformations. For example, assume a planar region constrained by a handle around it. If the user drags a handle in the middle of the planar region vertically, the isolines would merely be translated without the expected rotations, and the details over the planar region will not be reoriented as expected. A possible solution is to employ multiple transformations rather than only one for each of such isolines to better capture the local rotations. Lastly, like other iterative Laplacian editing frameworks [Huang et al. 2006; Au et al. 2006], our system requires specifying extra in-between handles to achieve deformations with rotation angles larger than π.

6 Deformation Transfer

In this section, we apply our reduced model to perform deformation transfer for triangular meshes. Existing related techniques require building per-triangle or per-vertex correspondence between the source and the target models [Sumner and Popović 2004; Zayer et al. 2005]. Although the mapping between models is not required to be one-to-one, to obtain a meaningful mapping, these methods need dozens of correspondence pairs, especially for models with large shape differences.

As the harmonic isolines provide a natural parameterization of low-frequency geometry of surface, we build correspondence between the harmonic isolines for deformation transfer. The user only has to specify corresponding pairs of handles on the source and the target. Choosing the same number of isolines per handle then gives isoline correspondence. The deformations are then transferred from the source to the target through isolines. While the number of correspondence pairs needed in previous methods mainly depends on the shape differences between the source and the target models, the number of handle correspondences needed in our method largely depends on the deformation complexity to be transferred, with rigid regions not requiring correspondence. For example, in Figure 13 (right), as the user expects the elephant trunk and tusks to be rigid, no handles are placed at their ends.

Note that, like [Sumner and Popović 2004], our method requires the source and the target to have similar semantic correspondence and poses in order to produce visually pleasing deformation transfer results. In addition, due to the possible shape difference between the source and the target (e.g., the horse/elephant example in Figure 13), the transformation for each isoline should be modeled without the translational component. This motivates us to model the per-isoline transformation from every mesh edge (thus eliminating the local translations) by solving the following linear system:

$$\mathbf{W}_e^{src}\mathbf{R}_e = \mathbf{E}^{src}\mathbf{X}^{src}, \tag{11}$$

where $\mathbf{X}^{src}$ is a column vector consisting of the deformed source vertex positions and $\mathbf{E}^{src}$ is the vertex-edge transform matrix of the source model such that $\mathbf{E}^{src}\mathbf{X}^{src}$ are the directed edge vectors. $\mathbf{W}_e^{src}$ is constructed in a similar way as $\mathbf{W}_x$ and $\mathbf{W}_\delta$, but interpolating all the edges. Solving (11) in a least-squares sense results in the approximated translation-free transformations $\mathbf{R}_e$ at all isolines.

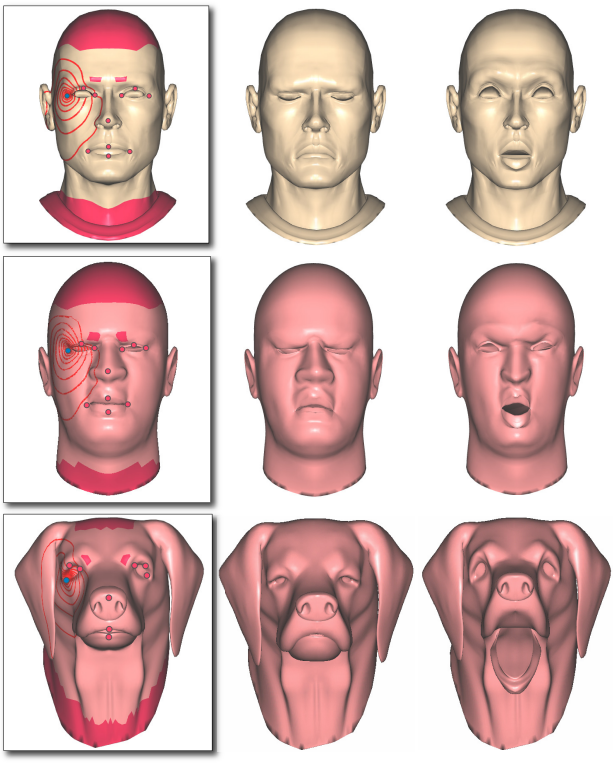

Figure 14: Expression transfer using handle correspondence.

Transferring $\mathbf{R}_e$ to the target model is done with the system below

$$\mathbf{E}^{tgt}\mathbf{X}^{tgt} = \mathbf{W}_e^{tgt}\mathbf{R}_e, \tag{12}$$

where $\mathbf{W}_e^{tgt}$ is the interpolation matrix of the edges for the target mesh and its given handles, and $\mathbf{X}^{tgt}$ and $\mathbf{E}^{tgt}$ are the deformed vertex positions and the vertex-edge transform matrix of the target mesh, respectively. Solving (12) in a least-squares sense produces a deformed target mesh with deformation similar to the source model. Since $\mathbf{E}^{tgt}$ is translation independent, we need to fix one target vertex to make (12) solvable.

Results. We apply our deformation transfer method to transfer poses and expressions. The deformations of the source models are all obtained using our interactive mesh deformation tool. In Figure 13, poses of articulated animals are transferred. By placing the handles at the skeletal joints of the animals, we can use a small set of handles to produce a variety of transferrable poses. For example, we use only 9 handle correspondences to transfer the sniffing and running poses of the wolf model to the dog model. With 13 handle correspondences, more complex poses, for example, the sitting pose, are transferrable. Building a full mesh correspondence between models with large shape differences, e.g., the horse and the elephant models, requires a large set of correspondence pairs, typically more than 60 pairs in [Sumner and Popović 2004; Zayer et al. 2005]. In contrast, without bothering with full mesh correspondence, our method focuses on the deformations to be transferred and is able to produce natural results using only a small set of handle correspondences, at most 15 pairs in our examples. Figure 14 shows an example of facial expression transfer. Subtle expressions in the source model are successfully captured by isoline-based transformations (driven by ten point handles and four region handles) and adapted to the target faces.

Limitations. When the source models are highly deformed, e.g., as in cloth deformation, more handles need to be placed to faithfully capture the source deformations, which increases the time complexity and the memory cost. Additionally, there is undesirable rubber-like deformation effect at the elephant legs in Figure 13. This is a common limitation of all differential mesh editing frameworks. To alleviate this effect, it is possible to either add more control handles or incorporate skeleton constraints [Huang et al. 2006] into our system.

7 Conclusion

We present the notion of handle-aware rigidity and develop an isoline-based reduced model that respects the handle-aware rigidity information and the geometry. The isolines serve as a generalized skeletal structure, which is independent of mesh sampling and transparent to the user who does not need to manipulate or control the isolines. We apply our reduced model to the applications of deformation transfer and interactive deformation, achieving detail-preserving deformation with resolution-independent per-iteration cost, fast convergence rate and linear memory cost. Our method retains the ease of implementation of the original differential mesh editing frameworks.

We believe that the handle-aware rigidity has large potential for other applications, such as 2D contour deformation and as-rigid-as-possible image manipulation [Igarashi et al. 2005; Weng et al. 2006].

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments, Nelson Chu for his great help during video production, and Tao Ju for early illuminating discussion on the topic. We are also grateful to Kun Zhou and Weiwei Xu for making the comparison example in Figure 2. This work was supported in part by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China, the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities), and the Israeli Ministry of Science.

References

AKSOYLU, B., KHODAKOVSKY, A., AND SCHRÖDER, P. 2005. Multilevel solvers for unstructured surface meshes. *SIAM Journal on Scientific Computing 26*, 4, 1146–1165. 7

AU, O. K.-C., TAI, C.-L., FU, H., AND LIU, L. 2005. Mesh editing with curvature flow laplacian operator. Tech. rep., Hong Kong University of Science Technology, Computer Science Technical Report, HKUST-CS05-10. 2, 3, 4, 5, 8

AU, O. K.-C., TAI, C.-L., LIU, L., AND FU, H. 2006. Dual Laplacian editing for meshes. *IEEE Transaction on Visualization and Computer Graphics 12*, 3, 386–395. 2, 3, 5, 8

BOTSCH, M., AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. *ACM Trans. Graph. 23*, 3, 630–634. 1, 4

BOTSCH, M., AND KOBBELT, L. 2005. Real-time shape editing using radial basis functions. In *Eurographics*, 611–621. 3

BOTSCH, M., PAULY, M., GROSS, M., AND KOBBELT, L. 2006. PriMo: coupled prisms for intuitive surface modeling. In *Symposium on Geometry Processing*, 11–20. 2, 4, 5

DER, K. G., SUMNER, R. W., AND POPOVIĆ, J. 2006. Inverse kinematics for reduced deformable models. *ACM Trans. Graph. 25*, 3, 1174–1179. 1

FU, H., AU, O. K.-C., AND TAI, C.-L. 2007. Effective derivation of similarity transformations for implicit Laplacian mesh editing. *Computer Graphics Forum 26*, 1, 34–45. 4

GARLAND, M., AND HECKBERT, P. S. 1997. Surface simplification using quadric error metrics. In *SIGGRAPH '97*, 209–216. 8

HUANG, J., SHI, X., LIU, X., ZHOU, K., WEI, L.-Y., TENG, S.-H., BAO, H., GUO, B., AND SHUM, H.-Y. 2006. Subspace gradient domain mesh deformation. *ACM Trans. Graph. 25*, 3, 1126–1134. 1, 2, 3, 5, 8, 9

IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. 2005. As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3, 1134–1141. 9

JAMES, D. L., AND TWIGG, C. D. 2005. Skinning mesh animations. *ACM Trans. Graph. 24*, 3, 399–407. 1

JU, T., SCHAEFER, S., AND WARREN, J. 2005. Mean value coordinates for closed triangular meshes. *ACM Trans. Graph. 24*, 3, 561–566. 3

KOBBELT, L., CAMPAGNA, S., VORSATZ, J., AND SEIDEL, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of ACM SIGGRAPH 98*, 105–114. 2

LEWIS, J. P., CORDNER, M., AND FONG, N. 2000. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of ACM SIGGRAPH 2000*, 165–172. 2

LINDHOLM, E., KLIGARD, M. J., AND MORETON, H. 2001. A user-programmable vertex engine. In *SIGGRAPH 2001*, 149–158. 7

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. 2, 3

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph. 24*, 479–487. 2

LIPMAN, Y., COHEN-OR, D., GAL, R., AND LEVIN, D. 2007. Volume and shape preservation via moving frame manipulation. *ACM Trans. Graph. 26*, 1, 5. 2

MOHR, A., AND GLEICHER, M. 2003. Building efficient, accurate character skins from examples. *ACM Trans. Graph. 22*, 3, 562–568. 2

SEDERBERG, T. W., AND PARRY, S. R. 1986. Free-form deformation of solid geometric models. In *Computer Graphics (Proceedings of ACM SIGGRAPH 86)*, 151–160. 3

SHI, L., YU, Y., BELL, N., AND FENG, W.-W. 2006. A fast multigrid algorithm for mesh deformation. *ACM Trans. Graph. 25*, 3, 1108–1117. 2

SORKINE, O., LIPMAN, Y., COHEN-OR, D., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Symposium on Geometry Processing*, 179–188. 1, 2, 3

SORKINE, O. 2006. Differential representations for mesh processing. *Computer Graphics Forum 25*, 4, 789–807. 2

SUMNER, R. W., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph. 23*, 3, 399–405. 8, 9

TOLEDO, S., 2003. TAUCS: a library of sparse linear solvers, version 2.2. Tel-Aviv University, Available online at http://www.tau.ac.il/ stoledo/taucs/. 7

WENG, Y., XU, W., WU, Y., ZHOU, K., AND GUO, B. 2006. 2D shape deformation using nonlinear least squares optimization. In *The Visual Computer (Pacific Graphics 2006)*, 653–660. 9

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. 1, 2

ZAYER, R., RÖSSL, C., KARNI, Z., AND SEIDEL, H.-P. 2005. Harmonic guidance for surface deformation. *Computer Graphics Forum 24*, 3, 601–609. 2, 4, 8, 9

ZHOU, K., HUANG, J., SNYDER, J., LIU, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2005. Large mesh deformation using the volumetric graph Laplacian. *ACM Trans. Graph. 24*, 3, 496–503. 2