As-Rigid-As-Possible Surface Modeling for Heterogeneous Deformable Surfaces

Koki Nagano Hongyi Xu
University of Southern California

Abstract

We demonstrate an as-rigid-as-possible surface modeling technique [Sorkine and Alexa 2007] and endeavour to adapt the scheme for modeling heterogeneous deformable surfaces. By requiring the rigidity of the local transformations, the as-rigid-as-possible surface modeling iteratively minimizes a non-linear energy under particular modeling constraints. The resulting deformation preserves the shape of the object locally. It assumes, however, a uniform rigidity across the surface. To accomodate heterogeneous rigidities distribution, we demonstrate an edge weight editing scheme using a texture as an input ridigity distribution. We show both of the original and adapted surface modeling methods on several examples, including primitive models, a human face model, and a heterogeneous creature model .

1 Introduction

As-rigid-as-possible surface modeling [Sorkine and Alexa 2007], is one of the most popular methods for surface deformation and editing. It follows the principle that surface details tend to be preserved when local surface deformations induced by user editing are close to rigid. It formulates a non-linear energy, which can be efficiently optimized with user-specified constraints by iteratively solving a linear system. This method creates detail-preserving and intuitive deformations. We implement this method based on VegaFEM [Barbič et al. 2012], which is a free physics library with half-edge data-structure for 3D deformable object simulation.

The existing as-rigid-as-possible surface editing scheme [Sorkine and Alexa 2007] assumes uniform rigidity across the surface. It could be an undesirable property for modeling real-life materials which exhibit heterogeneous rigidity distribution. Human skin under deformation, for instance, is more likely to form wrinkles at certain positions, implying weaker local rigidity around the corresponding parts. In this project, we adjust the as-rigid-as-possible editing scheme such that it can be applied on heterogeneous surfaces. For the original method, the local edge length preservation is of control with per-edge cotangent weights, to compensate for non-uniformly shaped cells and prevent discretization bias. In analogous to the heterogeneous mass spring system, where each edge can be assigned different stiffness, we modify the local weights to extend the local deformation controllability to conform to users favoured deformation. We demonstrate our adapted surface modeling results on a heterogeneous sheet mesh and a heterogeneous creature example.

2 Methodology

We denote a triangle mesh as $\mathcal{S}$, which consists of n vertices and m triangles. $\mathcal{N}(i)$ is the one-ring neighboring vertices connected to vertex i. The rest and deformed positions are denoted as $p \in \mathbb{R}^{3n}$ and $p' \in \mathbb{R}^{3n}$, respectively.

Given the cell $\mathcal{C}_i$ and its deformed version $\mathcal{C}'_i$, if the deformation is rigid, there exists a rotation matrix R_i such that

$$p'_i - p'_j = R_i(p_i - p_j), \forall j \in \mathcal{N}(i). \tag{1}$$

When the deformation is not rigid, we can still formulate a least-square energy for optimal rotation R as

$$E(\mathcal{C}_i, \mathcal{C}'_i) = \sum_{j\in\mathcal{N}(i)} w_{ij} \|(p'_i - p'_j) - R_i(p_i - p_j)\|^2, \tag{2}$$

where w_{ij} is the edge weights. The weights should compensate for non-uniformly shaped cells and prevent discretization bias. We therefore use the cotagent weight formula for w_{ij}

$$w_{ij} = \frac{1}{2}(cot\alpha_{ij} + cot\beta_{ij}), \tag{3}$$

where α_{ij} and β_{ij} are the opposite angles of the mesh edge (i, j). The rigidity of a deformation of the whole deformed mesh $\mathcal{S}'$ can be measured as the sum of the deviations from rigidity per cell, which can be formulated as

$$\begin{aligned} E(\mathcal{S}') &= \sum_{i=1}^{n} w_i E(\mathcal{C}_i, \mathcal{C}'_i) \\ &= \sum_{i=1}^{n} w_i \sum_{j\in\mathcal{N}(i)} w_{ij} \|(p'_i - p'_j) - R_i((p_i - p_j)\|^2, \end{aligned} \tag{4}$$

where w_i is the cell weights and we set to be $w_i = 1$, following the original surface deformation scheme. We need to solve for p' and R_i that minimizes $E(\mathcal{S}')$, under some user-defined modeling constraints. We use an alternating minimization strategy (it can also be refered as *block coordinate descent*). For the given set of positions p', we find the locally optimal rigid transformation R_i by

$$S_i = \sum_{j\in\mathcal{N}(i)} w_{ij}(p_i - p_j)(p'_i - p'_j)^T = U_i \Sigma_i V_i^T, \tag{5}$$

$$R_i = V_i U_i^T, \tag{6}$$

where $U_i \Sigma_i V_i^T$ is the singular value decomposition of S_i. With the given set of rigid transformations, we find positions p' that minimize $E(S')$ by formulating a Lagrange multiplier as

$$\begin{bmatrix} L & S^T \\ S & 0 \end{bmatrix} \begin{bmatrix} p' \\ \lambda \end{bmatrix} = \begin{bmatrix} b \\ \bar{p'} \end{bmatrix}, \tag{7}$$

where L is the Laplace-Beltrami operator with w_{ij} for entry (i, j) and $\bar{p'}$ is the user-displaced handle positions. b is a n-vector whose ith row is equal to

$$\sum_{j\in\mathcal{N}(i)} \frac{w_{ij}}{2}(R_i + R_j)(p_i - p_j). \tag{8}$$

Note that L only depends on the topology and therefore the left side of system matrix is fixed unless we add or remove handles. We only need to do a single sparse Cholesky decomposition when the constraint vertices are updated. We used the result of the previous frame as the initial guess, since the motion of the handle is expected to be continuous. For the detailed derivation, please refer to [Sorkine and Alexa 2007].

2.1 Heterogeneous Rigidity Editing

We endeavoured to adapt the original as-rigid-as-possible surface editing scheme to support the editing of heterogeneous deformable surfaces. Our surface editing scheme involves the Laplacian matrix, which is symmetric and structured as,

$$L_{ij} = \begin{cases} \sum\limits_{j \in \mathcal{N}(i)} w_{ij} & i = j \\ -w_{ij} & j \in \mathcal{N}(i) \\ 0 & otherwise, \end{cases}$$

where w_{ij} is an edge weight of choice, and it contributes to the local edge-length preservation during optimization. We edited the local edge weight w_{ij} in order to extend the flexibility of the surface modeling for heterogeneous rigidities. We use a texture as an input rigidity distribution field to the system, and map the texture values into rigidity multiplicators via $F : \Omega \rightarrow \Omega'$, where Ω is a field of $[0,1]$ and Ω' is a user's favoured distribution field with the range of $[a,b]$ $(a > 0)$. To assign an input weight value from a texture for each edge, we averaged the texture coordinates of the two end vertices of the edge for the look-up. Then we scale our original cotangent weight w_{ij} with the corresponding scalar value of Ω'. In the next section we show the heterogeneous surface editing results along with our choice of the mapping function F on different examples.

3 Results

We have implemented the as-rigid-as-possible deformaiton technique using C++ on Intel Xeon 2.9 GHz CPU (2x8 cores) machine with 32GB RAM, and a GeForce GTX 680 graphics card with 2GB RAM. We used the sparse linear solve, including the sparse Cholesky decomposition, and SVD implementation from VegaFEM [Barbič et al. 2012].

For uniform rigidities, we present some typical deformation results in Figures 1-4. The accompanying video shows several short editing sessions captured live. We always used 4 iterations per edit.

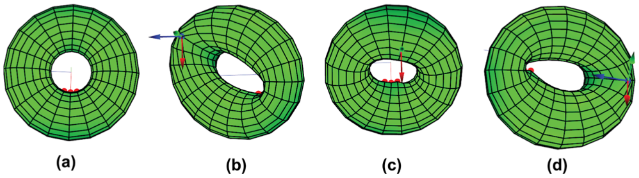

Figure 1: Editing the torus: *(a) is a torue model with* 400 *vertices, at rest configuration. (b), (c) and (d) display the editing results, with the static and handle achors denoted in red and blue, respectively. It runs interactively at* 45 *fps.*

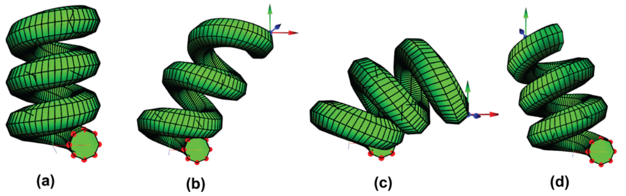

Figure 2: Deforming the helix: *(a) is a helix model with* 1,208 *vertices, at rest configuration. Position handles are translated to yield the results (b), (c) and (d). Notice that intesting rotation are generated. It runs interactively at* 21 *fps.*

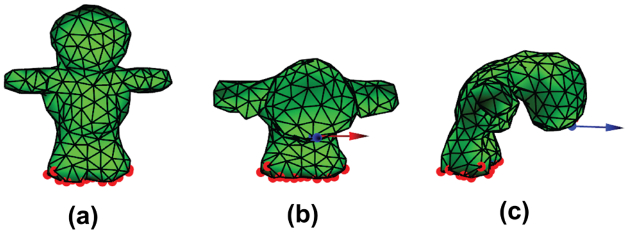

Figure 3: Deforming the turtle: *(a) is a turtle model with* 347 *vertices, at rest configuration. (b), (c) are the front view and side view for large deformation obtained by translating a single vertex constraint. It runs interactively at* 50 *fps.*

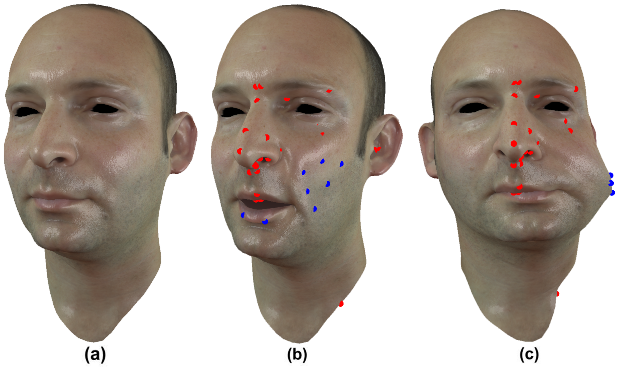

Figure 4: Editing the head: *(a) is a head model with* 14,097 *vertices, at rest configuration. (b), (c) are the deformed face under serveral positional constraints (denoted as blue dots). It runs interactively at* 1.5 *fps.*

3.1 Non-uniform Rigidities

We demonstrate results of our ridigity editing scheme on a simple square sheet mesh with 900 vertices (Fig.5 (a), (b)) and a heterogeneous armadillo model with 5000 vertices (c) with different rigidity distribution fields as input (first row). For all of the models, we found that $F = (10v+1)^3 w_{ij}$ achieved desireble results as shown in the figures, where $v \in [0,1]$ is a input weight value, and w_{ij} is an edge weight of choice. For the square sheet model, the smooth distribution weight (Fig.5 (a)) achieved heterogeneous edge-length preservation in a continuous manner. The effect is more obvious in the solid fringe pattern weight distribution (Fig.5 (b)), where the edges corresponding to the while stripes are firmly preserved while those corresponding the black stripes are substantially stretched. Finally we applied our technique to the heterogeneous creature model to achieve more flexible surface modeling. Notice that the local shape of its backshell, arms, and thighs are well preserved under the stretching and compression.

References

Barbič, J., Sin, F. S., and Schroeder, D., 2012. Vega FEM Library. http://www.jernejbarbic.com/vega.

Sorkine, O., and Alexa, M. 2007. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SGP '07, 109–116.

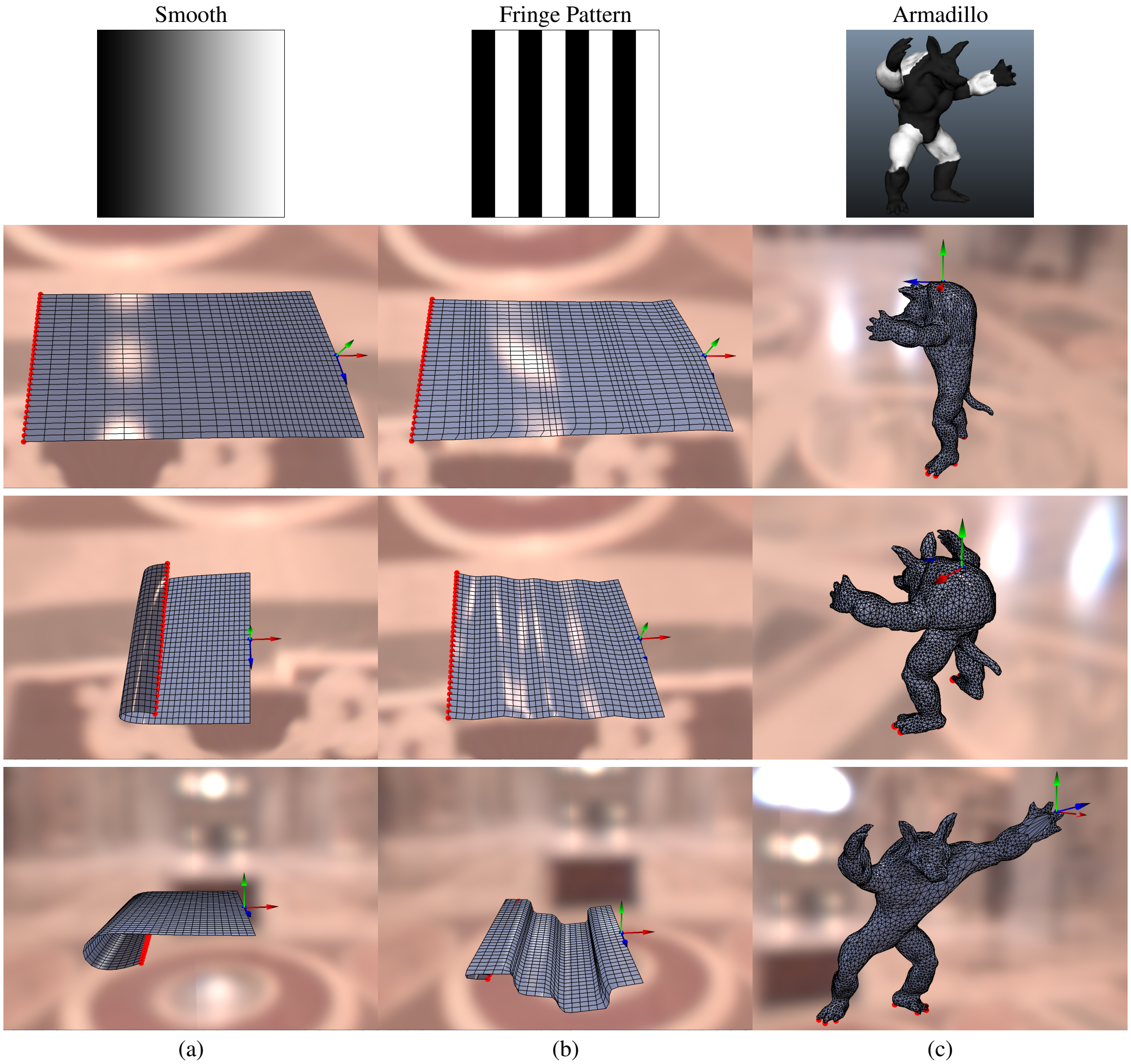

Figure 5: *Heterogeneous surface modeling results with smooth (a) and fringe pattern (b) distributions, and a heterogeneous creature model (c) under stretch (second row) and compression (third row) are shown in different views (forth row).*