To appear at the Eurographics Symposium on Geometry Processing (2007)
Alexander Belyaev, Michael Garland (Editors)

As-Rigid-As-Possible Surface Modeling

Olga Sorkine and Marc Alexa

TU Berlin, Germany

Abstract

Modeling tasks, such as surface deformation and editing, can be analyzed by observing the local behavior of the surface. We argue that defining a modeling operation by asking for rigidity of the local transformations is useful in various settings. Such formulation leads to a non-linear, yet conceptually simple energy formulation, which is to be minimized by the deformed surface under particular modeling constraints. We devise a simple iterative mesh editing scheme based on this principle, that leads to detail-preserving and intuitive deformations. Our algorithm is effective and notably easy to implement, making it attractive for practical modeling applications.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling – geometric algorithms, languages, and systems

1. Introduction

When we talk about *shape*, we usually refer to a property that does not change with the orientation or position of an object. In that sense, preserving shape means that an object is only rotated or translated, but not scaled or sheared. In the context of interactive shape modeling it is clear, however, that a shape has to to be stretched or sheared to satisfy the modeling constraints placed by the user. Users intuitively expect the deformation to preserve the shape of the object locally, as happens with physical objects when a smooth, large-scale deformation is applied to them. In other words, small parts of the shape should change as rigidly as possible.

Our goal is to create a shape deformation framework that is directly based upon the above principle. When local surface deformations induced by modeling operations are close to rigid, surface details tend to be preserved. This is a highly important property for surface editing schemes that are meant to be applied to complex, detailed surfaces, such as those coming from scanning real 3D objects or from sophisticated virtual sculpting tools. Recently, detail-preserving surface editing techniques have been receiving much attention in geometric modeling research [Sor06, BS07], thanks to the increasing proliferation of such detailed models, which usually come in the form of irregular polygonal meshes.

We propose the following conceptual model derived from the principle of local rigidity: The surface of the object is covered with small overlapping cells. An ideal deformation seeks to keep the transformation for the surface in each cell as rigid as possible. Overlap of the cells is necessary to avoid surface stretching or shearing at the boundary of the cells.

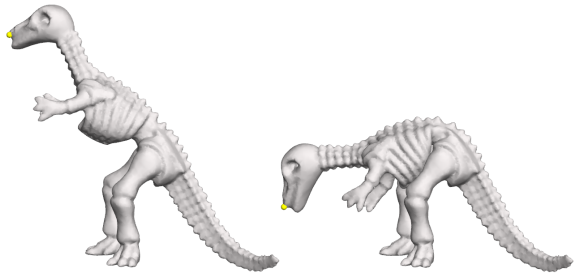

Figure 1: *Large deformation obtained by translating a single vertex constraint (in yellow) using our as-rigid-as-possible technique.*

For this modeling framework to become practical we shall define how rigidity is measured in each of the cells. A natural choice is to estimate the rigid transformation for each cell based on corresponding points on the initial and the deformed surfaces, then apply this rigid transformation to the original shape and measure the deviation to the deformed shape. Note that estimating a *linear* transformation from the corresponding surface points and then measuring its non-orthogonality is not a good measure of rigidity: an optimal approximate linear transformation of an arbitrary deformation could well be orthogonal. Consequently, in order to define locally shape-preserving deformation, a direct optimization of the rigid transformation should be performed instead.

Assuming we can measure deviation from rigidity in each cell, setting up the modeling framework requires deciding on the size and placement (or, equivalently, overlap) of

the cells. These parameters are interrelated: First, all cells should cover similar areas of the surface, or the area of the cells needs to be properly weighted in the objective funtion. Second, the overlap should be chosen in such a way that each part of the surface is covered by the same number of cells.

In the following, we will derive this modeling approach for the setting of discrete surfaces, namely triangular meshes. Surprisingly, a rather straightforward formulation of the main ideas outlined above leads to an intuitive iterative procedure with *guaranteed convergence* that is *simple to implement*. It also turns out that it can be interpreted as an iterative improvement of the now-common discrete Laplacian modeling frameworks [Sor06], which greatly facilitates its adoption into existing modeling frameworks. We will also relate the approach to other linear and non-linear modeling frameworks and discuss the tradeoffs.

1.1. Background

Local rigidity can be seen as the governing principle of various surface deformation models. If we look at the classical elastic energy that measures the difference between two shapes, or the so-called shell energy [TPBF87]:

$$E_s\left(\mathcal{S},\mathcal{S}'\right) = \int_\Omega k_s \left\|\mathbf{I}'-\mathbf{I}\right\|_F^2 + k_b \left\|\mathbf{II}'-\mathbf{II}\right\|_F^2 \mathrm{d}u\mathrm{d}v, \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius norm, $\mathbf{I}$, $\mathbf{II}$ are the fundamental forms of the surface $\mathcal{S}$ and $\mathbf{I}'$, $\mathbf{II}'$ are the fundamental forms of its deformed version $\mathcal{S}'$, then we can see that the energy is minimized when $\mathcal{S}'$ is a rigid transformation of $\mathcal{S}$. This is true globally as well as locally: the fundamental forms define the surface *locally* in a *unique* manner, up to a rigid transformation, such that local shape preservation occurs when $\mathcal{S}$ and $\mathcal{S}'$ are locally a rigid transformation of one into the other.

In any surface deformation scenario, apart from the trivial cases, complete local rigidity of the surface cannot hold, since it would follow that the surface is globally rigid. Thus the shape distance (1) does not reach zero, but rather has some minimum value. This minimum is attained when the local transformations that occur between $\mathcal{S}$ and $\mathcal{S}'$ are *as-rigid-as-possible* and *smooth*. Smooth means that the local deformations are locally similar; when they are in addition close to being rigid this means that the fundamental forms are almost preserved, i.e. the shape is locally preserved.

The shell energy (1) is a non-linear function of the surface positions, which makes it difficult to operate with in practice. Several works in geometric modeling [KCVS98, BK04] proposed to formulate a linearized discrete version of this energy (see [BS07] for detailed discussion). The linearization allows an efficient optimization, but it causes artifacts, such as local detail distortion and general shape distortion when large deformations (and in particular, rotations) are involved. As a partial compensation for local detail distortion, multi-resolution surface representations can be employed [ZSS97, GSS99], such that the smooth (low-frequency) component of the surface is deformed first, and then high-frequency details are added back by local displacement. Unfortunately, this solution quickly leads to local self-intersections when the deformation introduces bending [BSPG06].

As mentioned above, a different take on minimizing shape distortion is to try and devise deformations that are locally rigid. The principle of as-rigid-as-possible deformation was successfully applied to shape interpolation [ACOL00, XZWB05]. In this case, the source and target shapes are known, and the question is how to determine intermediate "shape path" such that the deformation from source to target appears as-rigid-as-possible; this can be done by performing polar decomposition of the transformation of each discrete surface element and interpolating the scaling and the rotation components separately. In shape editing the deformation task is harder because the target surface is not known, only several modeling constraints are given (typically some prescribed positions for a subset of surface elements).

One possible approach is to optimize for local rigid transformations and preservation of differential coordinates [Ale01] simultaneously; this was proposed by Sorkine et al. [SLCO*04] and Igarashi et al. [IMH05]. However, this optimization is non-linear, because rotations cannot be linearly parameterized in either 2D or 3D. In 2D, it is possible to linearly express *similarity* transformations; this can be exploited to design a two-step editing process for 2D shapes: in the first step, local similarity transformations are optimized, and in the second stage isotropic scaling is eliminated. The same idea was implemented by Schaefer et al. [SMW06] in the context of 2D free-form deformations (space-warps): they devised a moving-least-squares framework for 2D space warping, where each element of the space grid deforms as-rigidly-as-possible, and the warping is controlled by positional constraints on several grid points. In 3D, even similarity transformations cannot be linearly parameterized; to keep an efficient linear deformation framework Sorkine et al. [SLCO*04] use a first-order approximation of similarity transformations, which works well when only moderate rotations are involved in the deformation.

Note that the shape distance (1) has two parameters k_s and k_b that determine the relative weight of tangential distortion (i.e., the first fundamental form) against the normal direction (second fundamental form). Lipman et al. [LCOGL07] assumed isometric deformations, such that the first fundamental form is preserved, and devised an elegant method to minimize the deviation of the second fundamental form using Cartan's moving frames formulation for discrete surfaces. Wardetzky et al. [WBH*07] show that when the deformation is isometric, the bending energy term can be formulated as a quadratic energy of the surface positions. In different situations one can apply different "weighting" of the two terms; isometry is not always the most important part (when the desired surface deformation is stretching rather than bending,

for example). It is also worth noting that if more complex modeling constrains are admitted, namely, if the user explicitly specifies the desired rotations (or general affine transformations) to be applied to some control points on the surface, the editing can be relatively easily performed by propagating those transformations to the unconstrained parts of the surface [YZX*04, LSLCO05, ZRKS05, LCOGL07]. This leads to pleasing and intuitive results when the prescribed affine transformation and translation are *compatible* [LSLCO05]; in general, however, such editing methods are translation-insensitive and may lead to unintuitive shape distortions.

In this work, we directly formulate as-rigid-as-possible editing of discrete surfaces in 3D as a variational problem. The energy formulation is non-linear, and we show a simple iterative approach to minimize it. Several related non-linear variational deformation methods were proposed very recently: dual Laplacian editing [ATLF06] starts with naive Laplacian editing as an initial guess and iteratively adjusts the local Laplacian coordinates to coincide with the surface normals and refits the surface geometry to those Laplacians. This alternating iterative approach is similar to ours in spirit, but they do not formulate a specific energy that their iterations are meant to minimize, so it is not clear how to characterize the fixed points of the iterative process. Non-linear approaches are typically too expensive for interactive manipulation of high resolution models, therefore multiresolution is usually employed. This is the case, e.g., in the following methods. Pyramid coordinates [KS06] are formulated as non-linear displacements over locally fitted planes; Huang et al. [HSL*06] employ non-linear Laplacian constraints (coupled with other constraints, such as volume preservation) and minimize the resulting objective function on small subspace mesh. The deformation of the coarse subspace drives a space warp, implemented using 3D mean-value coordinates [JSW05], to deform the detailed surface. The PRIMO system [BPGK06] relates the surface to a collection of prisms that envelop it; the prisms are thought of as rigid objects, and the integrated distance between their adjacent faces is minimized using a modified Newton scheme, again combined with a clever multiresolution technique.

All the non-linear approaches above produce quite compelling results, as does the approach we present here; the advantage of our technique is its energy formulation that leads to an iterative minimization scheme that is very easy to implement while still guaranteeing convergence. Also, the steps in the minimization are related to Laplacian surface deformation techniques, which might be useful in modeling frameworks relying on this idea.

2. Discrete surface setup

In the following we denote by $\mathcal{S}$ a triangle mesh, whose topology is determined by n vertices and m triangles. We denote by $\mathcal{N}(i)$ the set of vertices connected to vertex i, also called the one-ring neighbors. The piecewise linear geometric embedding of $\mathcal{S}$ is defined by the vertex positions $\mathbf{p}_i \in \mathbb{R}^3$. Assume $\mathcal{S}$ is being deformed into $\mathcal{S}'$ that has the same connectivity and a different geometric embedding $\mathbf{p}'_i$.

It is natural to define the cells over the topological elements of the mesh. Because of the required overlap, each cell should consist of more than one triangle. We believe it is natural to choose a vertex-based definition, where each cell covers the triangles incident upon a vertex. We first analyze the approximate rigid transformation per cell and formalize an energy function that measures the deviation from this rigid transformation. Then we show how to minimize this energy and use it in the context of modeling operations.

2.1. Analyzing rigid transformations between two cells

Given the cell $\mathcal{C}_i$ corresponding to vertex i, and its deformed version $\mathcal{C}'_i$, we define the approximate rigid transformation between the two cells by observing the edges emanating from the vertex i in $\mathcal{S}$ and $\mathcal{S}'$. If the deformation $\mathcal{C} \rightarrow \mathcal{C}'$ is rigid, there exists a rotation matrix $\mathbf{R}_i$ such that

$$\mathbf{p}'_i - \mathbf{p}'_j = \mathbf{R}_i \left(\mathbf{p}_i - \mathbf{p}_j\right), \quad \forall\, j \in \mathcal{N}(i). \tag{2}$$

When the deformation is not rigid, we can still find the best approximating rotation $\mathbf{R}_i$ that fits the above equations in a weighted least squares sense, i.e., minimizes

$$E(\mathcal{C}_i, \mathcal{C}'_i) = \sum_{j \in \mathcal{N}(i)} w_{ij} \left\| \left(\mathbf{p}'_i - \mathbf{p}'_j\right) - \mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j) \right\|^2. \tag{3}$$

This is a weighted instance of the shape matching problem [Hor87]. We will discuss the proper choice of weights w_{ij} in the next section.

We briefly describe the derivation for the optimal rotation $\mathbf{R}_i$ for fixed $\mathcal{C}_i, \mathcal{C}'_i$. For convenience, let us denote the edge $\mathbf{e}_{ij} := \mathbf{p}_i - \mathbf{p}_j$, and similarly for $\mathbf{e}'_{ij}$ for the deformed cell $\mathcal{C}'_i$. We denote summation over j as a shorthand for $j \in \mathcal{N}(i)$. Then we can rewrite (3) as

$$\begin{aligned} &\sum_j w_{ij} \left(\mathbf{e}'_{ij} - \mathbf{R}_i \mathbf{e}_{ij}\right)^T \left(\mathbf{e}'_{ij} - \mathbf{R}_i \mathbf{e}_{ij}\right) = \\ &= \sum_j w_{ij} \left(\mathbf{e}'^T_{ij} \mathbf{e}'_{ij} - 2\mathbf{e}'^T_{ij} \mathbf{R}_i \mathbf{e}_{ij} + \mathbf{e}^T_{ij} \mathbf{R}^T_i \mathbf{R}_i \mathbf{e}_{ij}\right) = \\ &= \sum_j w_{ij} \left(\mathbf{e}'^T_{ij} \mathbf{e}'_{ij} - 2\mathbf{e}'^T_{ij} \mathbf{R}_i \mathbf{e}_{ij} + \mathbf{e}^T_{ij} \mathbf{e}_{ij}\right). \end{aligned} \tag{4}$$

The terms that do not contain $\mathbf{R}_i$ are constant in the minimization and therefore can be dropped. We are thus remained with

$$\begin{aligned} \operatorname*{argmin}_{\mathbf{R}_i} \sum_j -2w_{ij} \mathbf{e}'^T_{ij} \mathbf{R}_i \mathbf{e}_{ij} &= \operatorname*{argmax}_{\mathbf{R}_i} \sum_j w_{ij} \mathbf{e}'^T_{ij} \mathbf{R}_i \mathbf{e}_{ij} = \\ &= \operatorname*{argmax}_{\mathbf{R}_i} Tr\left(\sum_j w_{ij} \mathbf{R}_i \mathbf{e}_{ij} \mathbf{e}'^T_{ij}\right) = \\ &= \operatorname*{argmax}_{\mathbf{R}_i} Tr\left(\mathbf{R}_i \sum_j w_{ij} \mathbf{e}_{ij} \mathbf{e}'^T_{ij}\right). \end{aligned}$$

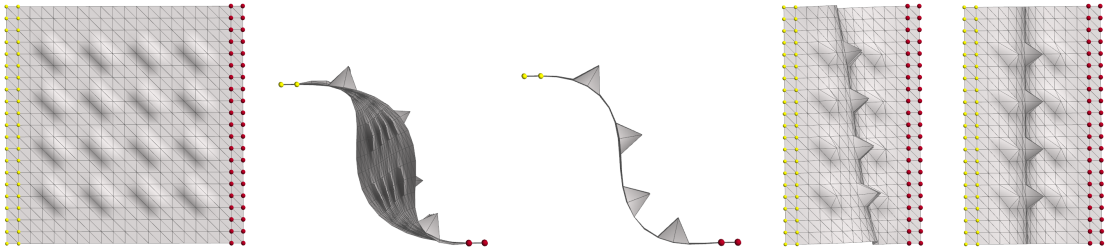

Figure 2: *Demonstration of the importance of proper edge weighting in the energy formulation (3). Deformation using uniform weighting ($w_{ij}=1$) leads to asymmetrical results, whereas cotangent weighting enables to eliminate the influence of the meshing bias.*

Let us denote by $\mathbf{S}_i$ the covariance matrix

$$\mathbf{S}_i = \sum_{j\in\mathcal{N}(i)} w_{ij}\,\mathbf{e}_{ij}\,\mathbf{e}_{ij}'^{T} = \mathbf{P}_i\,\mathbf{D}_i\,\mathbf{P}_i'^{T}\,, \tag{5}$$

where $\mathbf{D}_i$ is a diagonal matrix containing the weights w_{ij}, $\mathbf{P}_i$ is the $3\times|\mathcal{N}(v_i)|$ containing $\mathbf{e}_{ij}$'s as its columns, and similarly for $\mathbf{P}_i'$. It is well known that the rotation matrix $\mathbf{R}_i$ maximizing $Tr(\mathbf{R}_i\mathbf{S}_i)$ is obtained when $\mathbf{R}_i\mathbf{S}_i$ is symmetric positive semi-definite (if $\mathbf{M}$ is a psd matrix then for any orthogonal $\mathbf{R}$, $Tr(\mathbf{M})\geqslant Tr(\mathbf{RM})$). One can derive $\mathbf{R}_i$ from the singular value decomposition of $\mathbf{S}_i=\mathbf{U}_i\Sigma_i\mathbf{V}_i^T$:

$$\mathbf{R}_i = \mathbf{V}_i\mathbf{U}_i^T\,, \tag{6}$$

up to changing the sign of the column of $\mathbf{U}_i$ corresponding to the smallest singular value, such that $\det(\mathbf{R}_i)>0$.

2.2. The local rigidity energy

Our simple idea for measuring the rigidity of a deformation of the whole mesh is to sum up over the deviations from rigidity per cell, as expressed by (3). Thus, we obtain the following energy functional:

$$\begin{aligned} E(\mathcal{S}') &= \sum_{i=1}^{n} w_i\,E(\mathcal{C}_i,\mathcal{C}_i') = \\ &= \sum_{i=1}^{n} w_i \sum_{j\in\mathcal{N}(i)} w_{ij}\,\left\|(\mathbf{p}_i'-\mathbf{p}_j')-\mathbf{R}_i(\mathbf{p}_i-\mathbf{p}_j)\right\|^2\,, \end{aligned} \tag{7}$$

where w_i, w_{ij} are some fixed cell and edge weights. Note that $E(\mathcal{S}')$ depends solely on the geometries of $\mathcal{S},\mathcal{S}'$, i.e., on the vertex positions $\mathbf{p},\mathbf{p}'$. In particular, since the reference mesh (our input shape) is fixed, the only *variables* in $E(\mathcal{S}')$ are the deformed vertex positions $\mathbf{p}_i'$. This is because the optimal rotations $\mathbf{R}_i$ are well-defined functions of $\mathbf{p}'$, as was shown in the previous section.

The choice of per-edge weights w_{ij} and per-cell weights w_i is important for making our deformation energy as mesh-independent as possible, as demonstrated in Figure 2. The weights should compensate for non-uniformly shaped cells and prevent discretization bias. We therefore use the cotangent weight formula for w_{ij} [PP93, MDSB03]:

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

where α_{ij},β_{ij} are the angles opposite of the mesh edge (i,j) (for a boundary edge, only one such angle exists). We further note that the deviation from rigidity, as defined by (3), is an integrated quantity, so that the cell energy is proportional to the cell area, and we can set $w_i=1$. An alternative explanation for this would be using the area-corrected edge weights $w_{ij}'=(1/A_i)w_{ij}$, where A_i is the Voronoi area of cell $\mathcal{C}_i$ [MDSB03], and then also setting the cell weights to be the Voronoi area: $w_i'=A_i$. The area term simply cancels out, and we are left with the symmetric cotangent weights w_{ij}.

3. Modeling framework

In a modeling framework we need to solve for positions $\mathbf{p}'$ of $\mathcal{S}'$ that minimize $E(\mathcal{S}')$, under some user-defined modeling constraints. This means we do not know the rigid transformations $\{\mathbf{R}_i\}$ a priori and thus need to solve for them as well. Therefore, we first interpret $E(\mathcal{S}')$ as a function of $\mathbf{p}'$ and $\{\mathbf{R}_i\}$ and in any modeling situation we seek the minimum energy under the variation in both sets.

To solve for the next local minimum energy state (starting from a given initial vector of positions and rotations), we propose to use a simple alternating minimization strategy. This means, for a given fixed set of rigid transformations, we find positions $\mathbf{p}'$ that minimize $E(\mathcal{S}')$. Then, we find the rigid transformations $\{\mathbf{R}_i\}$ that minimize $E(\mathcal{S}')$ for the given set of positions $\mathbf{p}'$. We continue these interleaved iterations until the local energy minimum is reached.

Let us first look how to find optimal rigid transformations $\{\mathbf{R}_i\}$ for a given set of modified positions $\mathbf{p}'$. Each term in the sum (7) involves only the per-cell rigid transformation $\mathbf{R}_i$, i.e., we can compute an optimal transformation for each cell without regard for the other cells and their rigid transformations. Thus, we seek an $\mathbf{R}_i$ that minimizes the per cell energy in (3). The solution to this, however, is detailed in Section 2.1, namely, Equation (6).

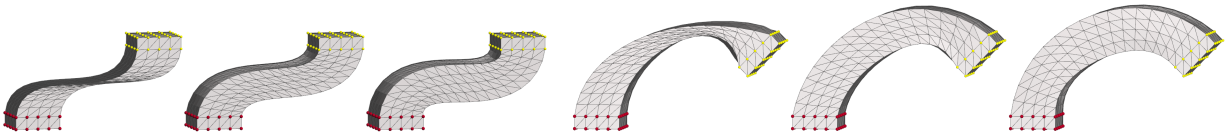

Figure 3: *Successive iterations of the as-rigid-as-possible editing method. The initial guess is the naive Laplacian editing result (as in [LSCO*04] but without any local rotation estimation). The original straight bar model is shown in Figure 6.*

In order to compute optimal vertex positions from given rotations, we compute the gradient of $E(\mathcal{S}')$ with respect to the positions $\mathbf{p}'$. Let us compute the partial derivatives w.r.t. $\mathbf{p}'_i$. Note that the only terms in $E(\mathcal{S}')$ whose derivative does not vanish are those involving vertices i and $j \in \mathcal{N}(i)$:

$$\begin{aligned}\frac{\partial E(\mathcal{S}')}{\partial \mathbf{p}'_i} &= \frac{\partial}{\partial \mathbf{p}'_i}\left(\sum_{j\in\mathcal{N}(i)} w_{ij}\left\|(\mathbf{p}'_i-\mathbf{p}'_j)-\mathbf{R}_i(\mathbf{p}_i-\mathbf{p}_j)\right\|^2 + \right.\\ &\left. + \sum_{j\in\mathcal{N}(i)} w_{ji}\left\|(\mathbf{p}'_j-\mathbf{p}'_i)-\mathbf{R}_j(\mathbf{p}_j-\mathbf{p}_i)\right\|^2\right) = \\ &= \sum_{j\in\mathcal{N}(i)} 2w_{ij}\left((\mathbf{p}'_i-\mathbf{p}'_j)-\mathbf{R}_i(\mathbf{p}_i-\mathbf{p}_j)\right) + \\ &+ \sum_{j\in\mathcal{N}(i)} -2w_{ji}\left((\mathbf{p}'_j-\mathbf{p}'_i)-\mathbf{R}_j(\mathbf{p}_j-\mathbf{p}_i)\right).\end{aligned}$$

Using the fact that $w_{ij} = w_{ji}$, we arrive at

$$\frac{\partial E(\mathcal{S}')}{\partial \mathbf{p}'_i} = \sum_{j\in\mathcal{N}(i)} 4w_{ij}\left((\mathbf{p}'_i-\mathbf{p}'_j)-\frac{1}{2}(\mathbf{R}_i+\mathbf{R}_j)(\mathbf{p}_i-\mathbf{p}_j)\right).$$

Setting the partial derivatives to zero w.r.t. each $\mathbf{p}'_i$ we arrive at the following sparse linear system of equations:

$$\sum_{j\in\mathcal{N}(i)} w_{ij}(\mathbf{p}'_i-\mathbf{p}'_j) = \sum_{j\in\mathcal{N}(i)} \frac{w_{ij}}{2}(\mathbf{R}_i+\mathbf{R}_j)(\mathbf{p}_i-\mathbf{p}_j). \tag{8}$$

The linear combination on the left-hand side is non-other than the discrete Laplace-Beltrami operator applied to $\mathbf{p}'$; the system of equations can be compactly written as

$$\mathbf{L}\mathbf{p}' = \mathbf{b}, \tag{9}$$

where $\mathbf{b}$ is an n-vector whose ith row contains the right-hand side expression from (8). We also need to incorporate the modeling constraints into this system. In the simplest form, those can be expressed by some fixed positions

$$\mathbf{p}'_j = \mathbf{c}_k, \quad k \in \mathcal{F}, \tag{10}$$

where $\mathcal{F}$ is the set of indices of the constrained vertices. These consist of static and handle vertices, interactively manipulated by the user. Incorporating such constraints into (9) simply means substituting the corresponding variables, effectively erasing respective rows and columns from $\mathbf{L}$ and updating the right-hand side with the values $\mathbf{c}_k$.

Note that the rigid transformations $\{\mathbf{R}_i\}$ only influence the right-hand side of the system, whereas the system matrix only depends on the initial mesh. Thus, we can employ a direct solver, and the system matrix has to be factored only once for minimizing $E(\mathcal{S}')$. Moreover, since $\mathbf{p}'$ consists of three columns (for the three coordinate functions), we only need to perform three times back-substitution to solve for each coordinate, using the same $n \times n$ factorization. Since $\mathbf{L}$ is symmetric positive definite, the sparse Cholesky factorization with fill-reducing reordering is an efficient solver choice [Tol03].

To summarize, the overall minimization of $E(\mathcal{S}')$ proceeds as follows. Firstly the coefficients w_{ij} are precomputed and the system matrix of (9) is pre-factored. Given an initial guess $\mathbf{p}'_0$, the local rotations $\mathbf{R}_i$ are estimated, as described in Section 2.1. New positions $\mathbf{p}'_1$ are obtained by solving (9), plugging $\mathbf{R}_i$ into the right-hand side. Then further minimization is performed by re-computing local rotations and using them to define a new right hand-side for the linear system, and so on. This leads to an efficient solution of the non-linear problem at hand, since only back-substitutions are necessary.

4. Results and discussion

We have implemented the as-rigid-as-possible deformation technique using C++ on a Pentium 4 2.16GHz laptop with 2GB RAM. We used the sparse Cholesky solver provided with the TAUCS library [Tol03] and standard SVD implementation (used for polar decomposition of 3×3 matrices) from [PTVF92].

We present some typical deformation results obtained with our technique in Figures 1, 4–8. Note that natural deformations are obtained, even when the manipulation handle is only being translated, because the optimization automatically produces the correct local rotations. The Cactus (Figure 7) is a particularly challenging example, especially for linear variational deformation methods, due to its long protruding features [BS07].

The results of our method can be compared with PRIMO [BPGK06], a state-of-the-art non-linear technique, as well as various linear variational techniques, by observing the canonical examples in Figures 5, 6, 7. Such deformations appear in the comparison table in [BS07]; it is evident that our method performs equally well to PRIMO and is generally superior to linear methods, especially when handle translation is involved. To emphasize this point, we compare the results of our method with Poisson mesh editing [YZX*04, ZRKS05] in Figure 5. Note that since the

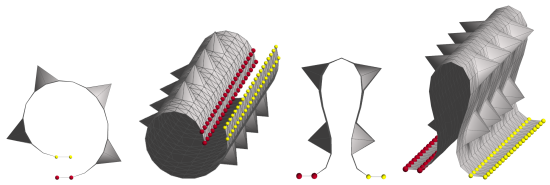

Figure 4: *Large deformation of the spiky plane (see Figure 2 for the original mesh). Note that the handle (in yellow) was only translated, without specifying any rotation.*

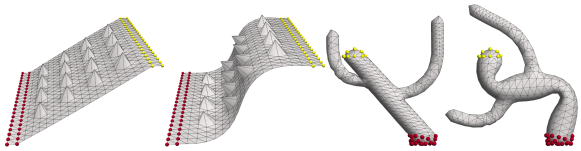

Figure 5: *Comparison with Poisson mesh editing. The original models appear in Figures 2 and 7. The yellow handle was only translated; this poses a problem for rotation-propagation methods such as [YZX*04, ZRKS05, LSLCO05].*

handle was only translated, Poisson editing cannot generate a proper rotation field (since there is no handle rotation to propagate), which results in detail distortion and lack of smoothness near the constraints. The same translation-insensitivity would be observed in the method of Lipman et al. [LCOGL07]; our technique handles translation well by optimizing for the local rotations, at the price of a global non-linear optimization. It is worth noting though that the required numerical machinery and the setup of the linear system is almost identical to the linear variational methods.

The accompanying video shows several short editing sessions captured live. Our unoptimized code runs interactively (at 10-30 fps) for regions of interest (ROI) of up to 10K vertices, using 2-3 iterations per edit. A number of improvements are possible to speed up convergence: a faster polar decomposition routine (i.e., one that reuses previous frame computations rather than starting from scratch each time) and a multiresolution technique, such as the one in [BPGK06] or [HSL*06], to allow the optimization to run on a coarse version of the mesh in order to quickly propagate the deformation across the ROI.

An important implementation issue is the initial guess which starts the optimization; since the energy we minimize is non-linear, multiple local minima may exist, and the solution depends on the initial guess in such case. It is important to use a reasonable-quality initial guess (i.e., not too far from the initial shape and the intuitively expected result) to allow quick convergence, yet it is desirable to compute it quickly. We experimented with several possibilities, which can be used in different scenarios:

Previous frame (for interactive manipulation): If the user interactively manipulates the control handle(s), it is reasonable to use the result of the previous frame as the initial guess, since the handle movement and/or deformation is expected to be continuous. Therefore, in this case we simply take the previous frame and assign the user-defined positions to the constrained vertices. This approach was used for all the figures in this paper, unless explicitly mentioned otherwise, and is also demonstrated in the accompanying video. The user experience reminds a lot of interacting with physical material.

Naive Laplacian editing: The starting guess is obtained by simple linear minimization of $\|\mathbf{L}\mathbf{p}' - \delta\|^2$ under the positional modeling constrains (10), where $\delta = \mathbf{L}\mathbf{p}$ are the differential coordinates of the input mesh. Although this guess produces distorted results for large deformations, the subsequent iterations manage to recover, as demonstrated in Figure 3. For significantly distorted initial guess the convergence may be slow, however.

Rotation-propagation: If the manipulation of the handle involves explicit rotation (along with translation), one can use any of the techniques that explicitly propagate the specified rotation to the unconstrained regions, such as [LSLCO05, ZRKS05, LCOGL07]. Subsequent optimization of our energy allows to consolidate the otherwise decoupled rotation and translation and improves the results; convergence is typically very fast since the starting rotational component of the deformation is already good.

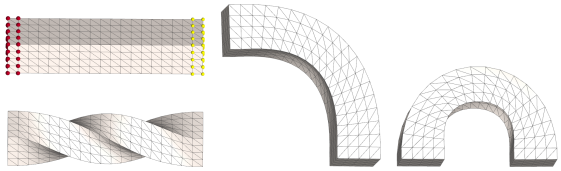

Figure 6: *Twist and rotation deformations.*

Model	Figure	Relative RMS error
Dino	Fig. 1	0.024
Spiky plane	Fig. 4 left	0.034
Spiky plane	Fig. 4 right	0.016
Twisted bar	Fig. 6 left	0.095
Armadillo	Fig. 8(b)	0.037
Armadillo	Fig. 8(c)	0.013
Armadillo	Fig. 8(e)	0.051

Table 1: *Relative RMS error of edge lengths for various deformations. When the modeling constraints do not necessitate stretching, the error is very low. The twist example does involve some slight stretching because the top of the bar is constrained to remain at the same height, hence the higher relative error in this case.*

An interesting property of our as-rigid-as-possible surface deformation is edge length preservation, to the extent allowed by the modeling constraints. If the modeling constraints do not impose stretching on the surface, the optimization always strives to converge to a state where the edge length error is small. This is clearly visible in the deforming

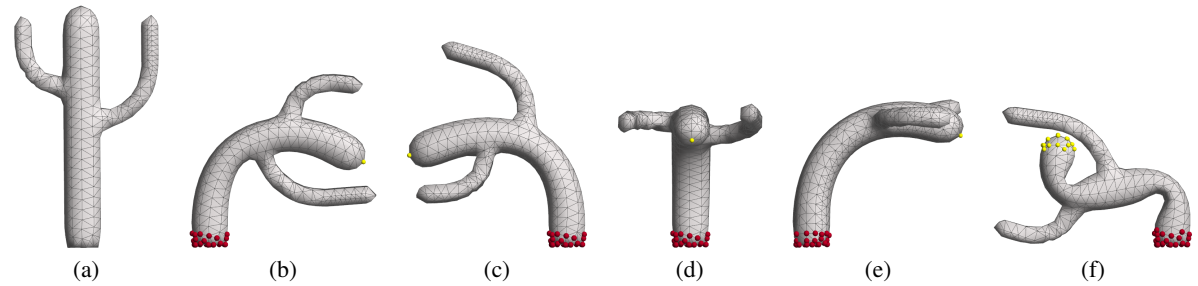

Figure 7: *Bending the Cactus. (a) is the original model; yellow handles are translated to yield the results (b-f). (d) and (e) show side and front views of forward bending, respectively. Note that in (b-e) a* single vertex *at the tip of the Cactus serves as the handle, and the bending is the result of translating that vertex, no rotation constraints are given.*

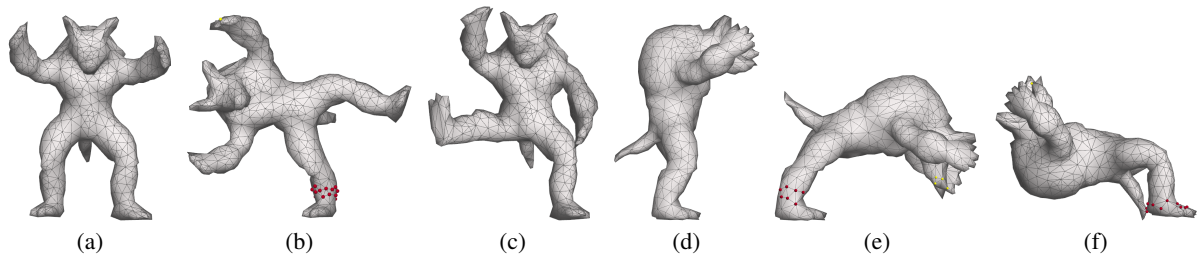

Figure 8: *Editing the Armadillo. (a) and (d) show views of the original model; the rest of the images display editing results, with the static and handle anchors denoted in red and yellow, respectively.*

plane example (Figure 4), for instance, which behaves similar to rubber-like material. Table 1 summarizes root-mean-square edge length error measurements for several deformations presented in this paper; it can be seen that the relative RMS error is very low.

5. Conclusions

The important features of our approach are (1) robustness, resulting from the minimization procedure that is guaranteed to not increase energy in each step; (2) simplicity, as each step of the minimization is conceptually similar to Laplacian modeling; and (3) efficiency, because the Laplace system matrix is constant throughout the iterations and has to be factored only once.

We have learned during our experiments that this combination is not evident, i.e., simply updating the right-hand side of a discrete Laplace system in a seemingly reasonable way would fail to converge in almost all cases. Convergence in our approach is the result of deriving an energy that cannot increase in each step of the iterations. Note that theoretically, the local minimum found by decreasing the energy might not be unique, i.e., there could be a connected set of minimum energy states. However, we have not experienced this problem and believe that if it exists at all then only for particularly derived examples.

The fact that each step in the iterations can be performed by solving a linear system with a constant matrix throughout the minimization procedure really is the result of a careful design of the energy functional. The number of iterations required to get reasonably close to a minimum depends on the condition number of the (anchored) Laplacian matrix, which is generally proportional to the mesh size. Specifically, if we keep the boundary conditions the same and refine the mesh, the condition number will grow proportionally, even if the shape of the mesh elements is perfect (for detailed analysis and bounds on the condition number of the uniform anchored Laplacian matrix, see [CCOST05]; the uniform Laplacian coincides with the cotangent Laplacian for tessellations with equilateral triangles, and in other cases the bounds for the cotangent Laplacian are probably more pessimistic). This means as the meshes are refined stability deteriorates, and typically more iterations are needed until convergence (in addition to the fact that each iteration becomes more costly). This practical efficiency problem could be easily alleviated with multi-resolution techniques.

Another interesting quality of our approach is that it trivially extends to volumetric cells, e.g., tetrahedra. As the rigidity is measured based on the edges in each cell, nothing would have to be changed in the setup of the energy – one would only have to plug-in the connectivity of a volumetric grid. So, if preservation of volume is of concern rather than preservation of surface, then this could easily be accomplished. Of course, as with other recent approaches, the optimization could be applied to a coarse volumetric grid

which controls the shape embedded in it, rather than directly to the discrete surface or volume.

In future work, we wish to experiment with several degrees of freedom that our modeling framework offers: changing the size and relative weights per cell, so as to control the overall and relative local rigidity of the surface.

Acknowledgement

We wish to thank Mario Botsch and Leif Kobbelt for insightful discussions and the anonymous reviewers for their valuable comments. This work was supported in part by the Alexander von Humboldt Foundation.

References

[ACOL00] ALEXA M., COHEN-OR D., LEVIN D.: As-rigid-as-possible shape interpolation. In *Proceedings of ACM SIGGRAPH* (2000), pp. 157–164.

[Ale01] ALEXA M.: Local control for mesh morphing. In *Proceedings of SMI* (2001), pp. 209–215.

[ATLF06] AU O. K.-C., TAI C.-L., LIU L., FU H.: Dual Laplacian editing for meshes. *IEEE TVCG 12*, 3 (2006), 386–395.

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. *ACM TOG 23*, 3 (2004), 630–634.

[BPGK06] BOTSCH M., PAULY M., GROSS M., KOBBELT L.: PriMo: Coupled prisms for intuitive surface modeling. In *Proceedings of SGP* (2006), pp. 11–20.

[BS07] BOTSCH M., SORKINE O.: On linear variational surface deformation methods. *IEEE TVCG* (2007). To appear.

[BSPG06] BOTSCH M., SUMNER R., PAULY M., GROSS M.: Deformation transfer for detail-preserving surface editing. In *Proceedings of VMV* (2006), pp. 357–364.

[CCOST05] CHEN D., COHEN-OR D., SORKINE O., TOLEDO S.: Algebraic analysis of high-pass quantization. *ACM TOG 24*, 4 (2005), 1259–1282.

[GSS99] GUSKOV I., SWELDENS W., SCHRÖDER P.: Multiresolution signal processing for meshes. In *Proceedings of ACM SIGGRAPH* (1999), pp. 325–334.

[Hor87] HORN B. K. P.: Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America 4*, 4 (1987).

[HSL*06] HUANG J., SHI X., LIU X., ZHOU K., WEI L.-Y., TENG S., BAO H., GUO B., SHUM H.-Y.: Subspace gradient domain mesh deformation. *ACM TOG 25*, 3 (2006), 1126–1134.

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J. F.: As-rigid-as-possible shape manipulation. *ACM TOG 24*, 3 (2005), 1134–1141.

[JSW05] JU T., SCHAEFER S., WARREN J.: Mean value coordinates for closed triangular meshes. *ACM TOG 24*, 3 (2005), 561–566.

[KCVS98] KOBBELT L., CAMPAGNA S., VORSATZ J., SEIDEL H.-P.: Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of ACM SIGGRAPH* (1998), ACM Press, pp. 105–114.

[KS06] KRAEVOY V., SHEFFER A.: Mean-value geometry encoding. *IJSM 12*, 1 (2006), 29–46.

[LCOGL07] LIPMAN Y., COHEN-OR D., GAL R., LEVIN D.: Volume and shape preservation via moving frame manipulation. *ACM TOG 26*, 1 (2007).

[LSCO*04] LIPMAN Y., SORKINE O., COHEN-OR D., LEVIN D., RÖSSL C., SEIDEL H.-P.: Differential coordinates for interactive mesh editing. In *Proceedings of SMI* (2004), pp. 181–190.

[LSLCO05] LIPMAN Y., SORKINE O., LEVIN D., COHEN-OR D.: Linear rotation-invariant coordinates for meshes. *ACM TOG 24*, 3 (2005), 479–487.

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Hege H.-C., Polthier K., (Eds.). Springer-Verlag, Heidelberg, 2003, pp. 35–57.

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experiment. Math. 2*, 1 (1993), 15–36.

[PTVF92] PRESS W. H., TEUKOLSKY S. A., VETTERLING W. T., FLANNERY B. P.: *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[SLCO*04] SORKINE O., LIPMAN Y., COHEN-OR D., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Proceedings of SGP* (2004), pp. 179–188.

[SMW06] SCHAEFER S., MCPHAIL T., WARREN J.: Image deformation using moving least squares. *ACM TOG 25*, 3 (2006), 533–540.

[Sor06] SORKINE O.: Differential representations for mesh processing. *Computer Graphics Forum 25*, 4 (2006), 789–807.

[Tol03] TOLEDO S.: TAUCS*: A Library of Sparse Linear Solvers, version 2.2*. Tel-Aviv University, Available online at http://www.tau.ac.il/~stoledo/taucs/, Sept. 2003.

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *Proceedings of ACM SIGGRAPH* (1987), pp. 205–214.

[WBH*07] WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete quadratic curvature energies. *CAGD* (2007). To appear.

[XZWB05] XU D., ZHANG H., WANG Q., BAO H.: Poisson shape interpolation. In *Proceedings of SPM* (2005), pp. 267–274.

[YZX*04] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.-Y.: Mesh editing with Poisson-based gradient field manipulation. *ACM TOG 23*, 3 (2004), 644–651.

[ZRKS05] ZAYER R., RÖSSL C., KARNI Z., SEIDEL H.-P.: Harmonic guidance for surface deformation. In *Computer Graphics Forum (Proceedings of Eurographics)* (2005), pp. 601–609.

[ZSS97] ZORIN D., SCHRÖDER P., SWELDENS W.: Interactive multiresolution mesh editing. In *Proceedings of ACM SIGGRAPH* (1997), pp. 259–268.