Linear Rotation-invariant Coordinates for Meshes

Yaron Lipman Olga Sorkine David Levin Daniel Cohen-Or
Tel Aviv University *

Abstract

We introduce a rigid motion invariant mesh representation based on discrete forms defined on the mesh. The reconstruction of mesh geometry from this representation requires solving two sparse linear systems that arise from the discrete forms: the first system defines the relationship between local frames on the mesh, and the second encodes the position of the vertices via the local frames. The reconstructed geometry is unique up to a rigid transformation of the mesh. We define surface editing operations by placing user-defined constraints on the local frames and the vertex positions. These constraints are incorporated in the two linear reconstruction systems, and their solution produces a deformed surface geometry that preserves the local differential properties in the least-squares sense. Linear combination of shapes expressed with our representation enables linear shape interpolation that correctly handles rotations. We demonstrate the effectiveness of the new representation with various detail-preserving editing operators and shape morphing.

Keywords: rigid-motion invariant shape representation, local frames, mesh editing, shape blending

1 Introduction

In this paper we introduce a rigid motion invariant mesh representation. The new representation describes the surface by its local properties, while filtering out the global spatial location and orientation. The representation consists of two discrete forms defined directly on the mesh. Reconstructing mesh geometry from locally defined quantities is a fundamental mechanism which allows editing a mesh while preserving its local appearance under some global constraints or boundary conditions. The focus here is on the local surface details rather than the spatial embedding.

Our mesh representation implicitly defines a local frame at each vertex, where the discrete forms encode the changes between adjacent frames. The key point is that the transitions between adjacent frames are expressed in relative coordinates. This relative encoding does not contain any global information that depends on the position and orientation of the mesh. The choice of local frames can be arbitrary. However, we define them analogously to the adapted frames [O'Neill 1969] or Cartan's moving frames [Guggenheimer 1963; Stoker 1989], that is, such that the third vector in the frame triplet is the normal to the surface. Such a definition enables intuitive decomposition of the representation into normal and tangential components.

The reconstruction of local frames from the discrete forms is expressed as a sparse linear system of equations. The global surface coordinates are obtained from the local frames by integration, also expressed as a solution of a linear system. We demonstrate that posing additional constraints, guided by interactive manipulation of the mesh, defines linear least-squares systems for surface editing. Using advanced numerical solvers allows interactive reconstruction.

The main contributions of this paper are:

- A rigid-motion invariant mesh representation based on discrete forms defined at each vertex.
- A linear surface reconstruction scheme that restores the geometry from the discrete forms.
- An interactive editing mechanism that strives to preserve local differential properties based on the surface reconstruction method.
- A linear shape interpolation technique which minimizes elastic distortion.

1.1 Related work

Interactive mesh editing is becoming a prominent field in geometric modeling due to the abundance of surface data in the form of irregular triangular meshes, originating mainly from 3D scanning devices. In the past years several mesh editing techniques were introduced [Zorin et al. 1997; Kobbelt et al. 1998; Guskov et al. 1999; Lee 1999; Bendels and Klein 2003; Botsch and Kobbelt 2004; Lipman et al. 2004; Sheffer and Kraevoy 2004; Sorkine et al. 2004; Yu et al. 2004; Zayer et al. 2005]. The main goals of interactive editing tools are an intuitive interface and preservation of surface details. Multiresolution approaches [Zorin et al. 1997; Kobbelt et al. 1998; Guskov et al. 1999; Botsch and Kobbelt 2004] enable detail-preserving deformations by decomposing the surface into several frequency bands. Roughly speaking, details are defined as the differences between successive levels in the multiresolution hierarchy, and are encoded with respect to the local frames of the lower level, in a rotation-invariant manner.

Some multiresolution techniques [Kobbelt et al. 1998; Botsch and Kobbelt 2004] work on a two-band decomposition: the smooth base mesh and the details, encoded in the local frames of the base mesh. This approach can be viewed as equivalent to the recent methods that work directly on the original mesh [Lipman et al. 2004; Sorkine et al. 2004; Yu et al. 2004]. The latter methods define details in a more implicit way, and do not require explicitly setting the smooth base level; on the other hand, the local frame orientation then needs to be handled explicitly. These methods strive to preserve certain differential properties, such as the discrete Laplacians [Lipman et al. 2004; Sorkine et al. 2004] or the gradients of the mesh coordinate functions [Yu et al. 2004]. These differential entities are vectors encoded in the *global* coordinate system this time, and therefore the main challenge of these techniques is to correctly modify the *local* frames to accommodate user-defined constraints and deformations. This is done by implicitly including a linearized version of the local frame transforms in the Laplacian fitting formulation [Sorkine et al. 2004], by explicitly assigning the local frames by propagating the user-defined transformation of the handle [Yu et al. 2004] or heuristically approximating the local rotations [Lipman et al. 2004]. In [Zayer et al. 2005] the transformation of the handle is interpolated over the mesh by using harmonic fields,

*e-mail:{lipmanya|sorkine|levin|dcor}@tau.ac.il

Copyright © 2005 by the Association for Computing Machinery, Inc.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions Dept, ACM Inc., fax +1 (212) 869-0481 or e-mail permissions@acm.org.
© 2005 ACM 0730-0301/05/0700-0479 $5.00

defined via the same Laplacian operator that is used in the editing operation itself.

The inherent problem of the above methods is that the local quantities, which constitute the surface representation, are not rotation invariant, and thus local rotations must be explicitly handled to achieve geometric shape preservation. Sheffer and Kraevoy [2004] represent the mesh by rotation-invariant pyramid coordinates. They achieve intuitive editing results for large deformation constraints, but the reconstruction process is not linear. The surface representation we present is truely rotation-invariant, which avoids the explicit handling of local frame transforms altogether, and the reconstruction is formulated as a sparse linear problem.

1.2 Overview

We describe our surface representation in the following section. We then proceed to formulate the surface equations that enable to reconstruct the surface geometry from our representation (Section 3). Mesh manipulation applications (detail-preserving editing and shape blending) are presented in Section 4. Finally, we discuss our method in Section 5 and conclude in Section 6.

2 Discrete Forms

In this section we introduce the *first* and *second discrete forms* $\widetilde{I}$ and $\widetilde{II}$ for meshes. These discrete quantities are invariant to rotation and translation of the mesh and contain enough information to reconstruct the mesh uniquely (up to rotation and translation). Let $M=(G,P)$ be a 2-manifold triangular mesh; $G=(V,E,F)$ is a graph where V, E, and F are the vertices, edges, and faces, respectively, and P is the geometry associated with each vertex in V. Vertex i is placed at $\hat{\mathbf{x}}^i$. Note that each vertex and its 1-ring can be locally parameterized. The *valence* of a vertex, denoted by d_i, is the number of edges which emanate from this vertex. The 1-ring neighborhood of each vertex is decomposed into the tangential part, represented by the first discrete form, and the normal part, which is represented by the second discrete form. The method for computing the tangent plane at each vertex should be invariant under rigid transformations of the mesh. In other words, if the mesh is transformed by some rigid transformation, the normal should be transformed by the same transformation. There are several advanced methods for computation of normals, such as [Meek and Walton 2000; Meyer et al. 2002; Cazals and Pouget 2003; Cohen-Steiner and Morvan 2003]. For our application, it is sufficient to define a vertex normal as the weighted average of the normals of the adjacent triangles, where the weights are proportional to the triangles' areas. We denote the tangent plane at vertex i by T_iM and its normal by $\mathbf{N}^i$.

Given a vertex i positioned at $\hat{\mathbf{x}}^i$, we denote by $\hat{\mathbf{x}}^i_1,\hat{\mathbf{x}}^i_2,\ldots,\hat{\mathbf{x}}^i_{d_i}$ the *vectors* emanating from the vertex $\hat{\mathbf{x}}^i$ to its neighbors. We denote by $\widetilde{\mathbf{x}}^i_k$ the projection of $\hat{\mathbf{x}}^i_k$ onto the tangent plane T_iM, and by $\mathbf{x}^i$ and $\mathbf{x}^i_k$ the representation of $\hat{\mathbf{x}}^i$ and $\hat{\mathbf{x}}^i_k$ in a *parameterization* U_i of the 1-ring in the plane, respectively [1] (illustrated in Figure 1). Note that all the following definitions and constructions are valid even if the projection of the 1-ring onto T_iM is not injective. If some edge is parallel to the normal at the vertex, one can perturb the normal to avoid this singularity. In practice we have never encountered such a situation.

[1] We use a hat ($\wedge$) for mesh vertices and edges embedded in $\mathbb{R}^3$ and tilde ($\sim$) for vertices and edges projected onto the tangent plane.

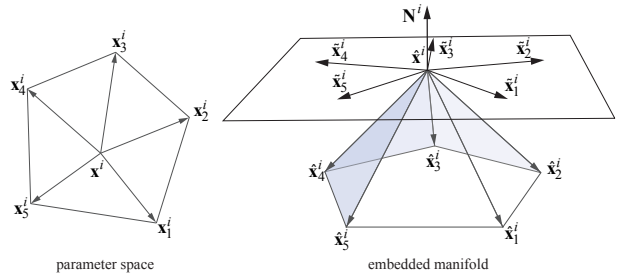


Figure 1: The 1-ring setting. Vertices in the parameter domain are denoted by $\mathbf{x}^i$ and the corresponding positions in $\mathbb{R}^3$ by $\hat{\mathbf{x}}^i$. The edge towards the kth neighbor of i is $\mathbf{x}^i_k$. Mesh edges in $\mathbb{R}^3$ are denoted by $\hat{\mathbf{x}}^i_k$, and their projection onto the tangent plane by $\widetilde{\mathbf{x}}^i_k$.

We define a piecewise inner product in U_i by assigning each parametric triangle $\triangle^i_k$ (formed by vertex $\mathbf{x}^i$ and its kth and $(k+1)$th neighbors) the standard inner product of its corresponding triangle in the tangent plane T_iM. Let $\mu=\mu_1\mathbf{x}^i_k+\mu_2\mathbf{x}^i_{k+1}$ be a vector in $\triangle^i_k$. Then we define the first discrete form $\widetilde{I}$ as:

$$\widetilde{I}^i(\cdot):\bigcup_{k=1}^{d_i-1}\triangle^i_k\longrightarrow\mathbb{R}.$$

$$\begin{aligned}\widetilde{I}^i(\mu) &= \langle\mu,\mu\rangle_{\mathbb{R}^3}=\langle\mu_1\widetilde{\mathbf{x}}^i_k+\mu_2\widetilde{\mathbf{x}}^i_{k+1},\ \mu_1\widetilde{\mathbf{x}}^i_k+\mu_2\widetilde{\mathbf{x}}^i_{k+1}\rangle_{\mathbb{R}^3}=\\ &= \mu_1^2\,\widetilde{g}^i_{k,k}+2\,\mu_1\,\mu_2\,\widetilde{g}^i_{k,k+1}+\mu_2^2\,\widetilde{g}^i_{k+1,k+1},\end{aligned}$$

where $\widetilde{g}^i_{k,k}=\langle\widetilde{\mathbf{x}}^i_k,\widetilde{\mathbf{x}}^i_k\rangle_{\mathbb{R}^3}$ and $\widetilde{g}^i_{k,k+1}=\langle\widetilde{\mathbf{x}}^i_k,\widetilde{\mathbf{x}}^i_{k+1}\rangle_{\mathbb{R}^3}$. We also denote $O^i_k:=\operatorname{sign}\big(\det(\widetilde{\mathbf{x}}^i_k,\ \widetilde{\mathbf{x}}^i_{k+1},\ \mathbf{N}^i)\big)$. The quantity O^i_k indicates the *orientation* of the triplet $(\widetilde{\mathbf{x}}^i_k,\ \widetilde{\mathbf{x}}^i_{k+1},\ \mathbf{N}^i)$.

Note that $\widetilde{I}^i(\cdot)$ is a quadratic form in each triangle $\triangle^i_k$ with a C^0 connection between adjacent triangles. Also note that the quantities $\widetilde{g}^i_{k,m}$ and O^i_k give a full parameterization of the tangent plane. Furthermore, the invariance of the first discrete form to different parameterizations can be easily verified. The first discrete form can be described by the *lengths* of the projected edges and the signed *angles* between every two adjacent projected edges.

The first discrete form lacks information in the normal direction of the 1-ring neighborhood of a vertex. We define the *second discrete form* as a piecewise *linear* form which is actually the height function of the 1-ring neighborhood of a vertex above the tangent plane:

$$\widetilde{II}^i(\cdot):\bigcup_{k=1}^{d_i-1}\triangle^i_k\longrightarrow\mathbb{R}.$$

Let $\mu=\mu_1\,\mathbf{x}^i_k+\mu_2\,\mathbf{x}^i_{k+1}\in\triangle^i_k$. Then,

$$\widetilde{II}^i(\mu):=\mu_1\,\langle\hat{\mathbf{x}}^i_k,\ \mathbf{N}^i\rangle_{\mathbb{R}^3}+\mu_2\,\langle\hat{\mathbf{x}}^i_{k+1},\ \mathbf{N}^i\rangle_{\mathbb{R}^3}=\mu_1\,\widetilde{L}^i_k+\mu_2\,\widetilde{L}^i_{k+1},$$

where we introduce the coefficients $\widetilde{L}^i_k=\langle\hat{\mathbf{x}}^i_k,\ \mathbf{N}^i\rangle_{\mathbb{R}^3}$.

The discrete forms at a vertex define the geometry of its 1-ring neighborhood up to a rigid transformation, as explained in the following lemma.

Lemma 2.1. *Given the discrete form coefficients and the orientation bits at vertex i, the 1-ring neighborhood of i is defined up to a rigid transformation. Fixing the position of vertex i at some*

point $\hat{\mathbf{x}}^i \in \mathbb{R}^3$, and fixing the direction of one edge emanating from vertex i (or its projection onto the tangent plane) and the normal $\mathbf{N}^i$, uniquely defines all the rest of the 1-ring vertex positions.

Proof. Assume w.l.o.g. that we fix the first edge $\widetilde{\mathbf{x}}_1^i$ in the tangent plane. Denote by $\mathbf{n}$ the unit length vector which is orthogonal to $\mathbf{N}^i$ and to $\widetilde{\mathbf{x}}_1^i$ and such that the ordered triplet $(\widetilde{\mathbf{x}}_1^i,\ \mathbf{n},\ \mathbf{N}^i)$ forms a right-hand orthogonal basis. Then,

$$\begin{aligned}\hat{\mathbf{x}}_2^i &= \langle \widetilde{\mathbf{x}}_2^i, \frac{\widetilde{\mathbf{x}}_1^i}{\|\widetilde{\mathbf{x}}_1^i\|}\rangle \frac{\widetilde{\mathbf{x}}_1^i}{\|\widetilde{\mathbf{x}}_1^i\|} + \langle \widetilde{\mathbf{x}}_2^i, \mathbf{n}\rangle \mathbf{n} + \langle \hat{\mathbf{x}}_2^i, \mathbf{N}^i\rangle \mathbf{N}^i = \\ &= \frac{\widetilde{g}_{12}}{\widetilde{g}_{11}} \widetilde{\mathbf{x}}_1^i + O_1^i \sqrt{\frac{\Delta}{\widetilde{g}_{11}\widetilde{g}_{22}}}\, \mathbf{n} + \widetilde{L}_2 \mathbf{N}^i,\end{aligned}$$

where $\Delta = \widetilde{g}_{11}\widetilde{g}_{22} - \widetilde{g}_{12}^2$.
In the same manner one can compute $\hat{\mathbf{x}}_3, ..., \hat{\mathbf{x}}_{d_i}$. □

Note that the discrete forms are not affected by the choice of the local parameterization because we compute their coefficients directly from the mesh. However, the parametric definition can be used to prove convergence of the discrete forms to continuous differential properties (see [Lipman 2004]). The main strength of these definitions is that they allow us to define discrete linear surface equations that represent the mesh by local quantities which are invariant under rigid transformations, as shown in the next section. It should be emphasized that the discrete forms are not scale- or shear-invariant.

3 Discrete surface equations

We use the definitions from Section 2 to introduce a surface representation. We define a *discrete frame* at vertex $i \in V$ as the triplet $(\mathbf{b}_1^i,\ \mathbf{b}_2^i,\ \mathbf{N}^i)$, where $\mathbf{b}_1^i \in T_iM$ is a unit vector parallel to $\widetilde{\mathbf{x}}_1^i$, $\mathbf{b}_2^i \in T_iM$ is a unit vector orthogonal to $\mathbf{b}_1^i$, such that the triplet $(\mathbf{b}_1^i,\ \mathbf{b}_2^i,\ \mathbf{N}^i)$ forms a right-hand orthonormal basis. Note that the choice of the vector $\widetilde{\mathbf{x}}_1^i$ is arbitrary.

Assume $(i,j) \in E$. We define the difference operator δ on the discrete frame vectors:

$$\begin{aligned}\delta_j(\mathbf{b}_1^i) &= \mathbf{b}_1^j - \mathbf{b}_1^i \\ \delta_j(\mathbf{b}_2^i) &= \mathbf{b}_2^j - \mathbf{b}_2^i \\ \delta_j(\mathbf{N}^i) &= \mathbf{N}^j - \mathbf{N}^i.\end{aligned}$$

Finally, we lay out the *discrete surface equations*:
$\forall\, i,j \in V$ s.t. $(i,j) \in E$

$$\begin{aligned}\delta_j(\mathbf{b}_1^i) &= \Gamma_{j,1}^{i,1}\mathbf{b}_1^i + \Gamma_{j,1}^{i,2}\mathbf{b}_2^i + A_{j,1}^i \mathbf{N}^i \\ \delta_j(\mathbf{b}_2^i) &= \Gamma_{j,2}^{i,1}\mathbf{b}_1^i + \Gamma_{j,2}^{i,2}\mathbf{b}_2^i + A_{j,2}^i \mathbf{N}^i \\ \delta_j(\mathbf{N}^i) &= \Gamma_{j,3}^{i,1}\mathbf{b}_1^i + \Gamma_{j,3}^{i,2}\mathbf{b}_2^i + A_{j,3}^i \mathbf{N}^i.\end{aligned} \tag{1}$$

The above discrete surface equations encode the difference between adjacent discrete frames corresponding to adjacent vertices of the mesh. The difference is encoded in the discrete frame of one of the vertices. The key observation is that the quantities $\Gamma_{j,m}^{i,l}$ and $A_{j,m}^i$ can be expressed by the coefficients of the discrete forms only.

Theorem 3.1. *The coefficients $\Gamma_{j,m}^{i,l}$ and $A_{j,m}^i$ of the discrete surface equations can be expressed by the discrete forms.*

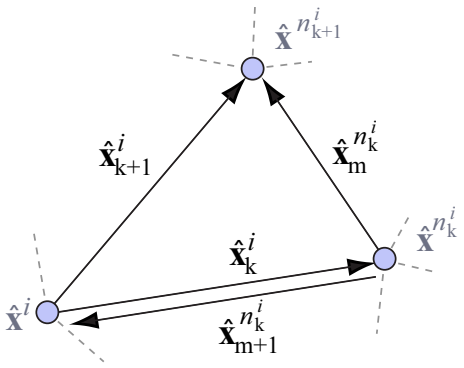


Figure 2: Illustration for Theorem 3.1. The vertex notations are in gray and the edge vector notations are black.

Proof. Note that equations (1) can be written in the following equivalent way:

$$\begin{aligned}\mathbf{b}_1^j &= \left(\Gamma_{j,1}^{i,1}+1\right)\mathbf{b}_1^i + \Gamma_{j,1}^{i,2}\mathbf{b}_2^i + A_{j,1}^i\mathbf{N}^i \\ \mathbf{b}_2^j &= \Gamma_{j,2}^{i,1}\mathbf{b}_1^i + \left(\Gamma_{j,2}^{i,2}+1\right)\mathbf{b}_2^i + A_{j,2}^i\mathbf{N}^i \\ \mathbf{N}^j &= \Gamma_{j,3}^{i,1}\mathbf{b}_1^i + \Gamma_{j,3}^{i,2}\mathbf{b}_2^i + \left(A_{j,3}^i+1\right)\mathbf{N}^i.\end{aligned} \tag{2}$$

These equations simply encode a discrete frame in the basis of an adjacent discrete frame. Thus, to calculate the coefficients of the above equations, one should represent the discrete frame at vertex j in the coordinates of the discrete frame at vertex i. For convenience, let us take the discrete frame at vertex i as the coordinate vectors, that is, $\mathbf{b}_1^i = (1,0,0),\ \mathbf{b}_2^i = (0,1,0),\ \mathbf{N}^i = (0,0,1)$. Note that we are *not* interested to find the actual discrete frame at j in the *global* coordinate system, but only its representation with respect to the frame at i. Denote the index of the kth neighbor of vertex i by n_k^i. Assume $j = n_k^i$. First let us find the normal $\mathbf{N}^{n_k^i}$. Note that

$$\hat{\mathbf{x}}_k^i = \hat{\mathbf{x}}^{n_k^i} - \hat{\mathbf{x}}^i = \widetilde{\mathbf{x}}_k^i + \widetilde{L}_k^i \mathbf{N}^i.$$

Since $$\hat{\mathbf{x}}_{k+1}^i = \hat{\mathbf{x}}^{n_{k+1}^i} - \hat{\mathbf{x}}^i = \widetilde{\mathbf{x}}_{k+1}^i + \widetilde{L}_{k+1}^i \mathbf{N}^i,$$

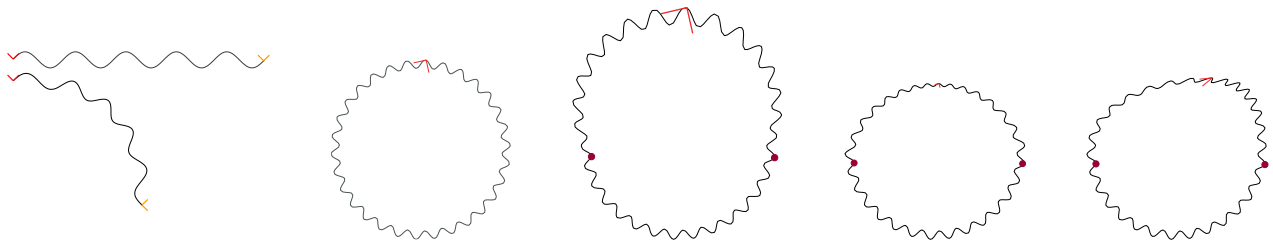
we also have

$$\hat{\mathbf{x}}^{n_{k+1}^i} - \hat{\mathbf{x}}^{n_k^i} = \widetilde{\mathbf{x}}_{k+1}^i + \widetilde{L}_{k+1}^i \mathbf{N}^i - \widetilde{\mathbf{x}}_k^i - \widetilde{L}_k^i \mathbf{N}^i.$$

Next, note that $\hat{\mathbf{x}}^{n_{k+1}^i} - \hat{\mathbf{x}}^{n_k^i}$ and $\hat{\mathbf{x}}^i - \hat{\mathbf{x}}^{n_k^i}$ are the vectors $\hat{\mathbf{x}}_m^{n_k^i}$ and $\hat{\mathbf{x}}_{m+1}^{n_k^i}$ for some $1 \le m \le d_{n_k^i}$ (see Figure 2), so finding $\mathbf{N}^{n_k^i}$ reduces to solving the following linear 2×2 system:

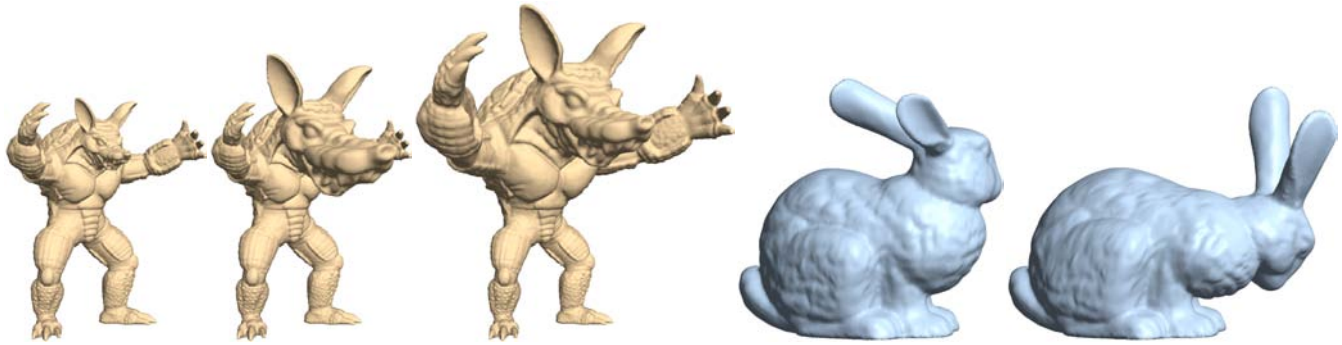
$$\begin{aligned}\langle \hat{\mathbf{x}}_m^{n_k^i}, \mathbf{N}^{n_k^i}\rangle &= \langle \widetilde{\mathbf{x}}_{k+1}^i - \widetilde{\mathbf{x}}_k^i, \mathbf{N}^{n_k^i}\rangle + \left(\widetilde{L}_{k+1}^i - \widetilde{L}_k^i\right)\langle \mathbf{N}^i,\ \mathbf{N}^{n_k^i}\rangle \\ \langle \hat{\mathbf{x}}_{m+1}^{n_k^i},\ \mathbf{N}^{n_k^i}\rangle &= -\langle \widetilde{\mathbf{x}}_k^i,\ \mathbf{N}^{n_k^i}\rangle - \widetilde{L}_k^i \langle \mathbf{N}^i,\ \mathbf{N}^{n_k^i}\rangle.\end{aligned} \tag{3}$$

The left-hand sides of the above equations are the second discrete form coefficients at vertex n_k^i: $\widetilde{L}_m^{n_k^i}$ and $\widetilde{L}_{m+1}^{n_k^i}$. By writing Eq. 3 in the discrete frame at vertex i we get an under-determined system for $\mathbf{N}^{n_k^i}$ in that basis. Taking a unit length solution to this system results in the coordinate vector of $\mathbf{N}^{n_k^i}$ with respect to the discrete frame at vertex i (the choice between the two possible solutions of norm 1 is determined by the orientation bits). Next, since we have the vertex position $\hat{\mathbf{x}}^{n_k^i}$, the normal $\mathbf{N}^{n_k^i}$ and one edge $\hat{\mathbf{x}}_m^{n_k^i}$, all expressed in the coordinates of the discrete frame of i, the whole 1-ring of vertex n_k^i is determined by Lemma 2.1. By the definition of the discrete frame, it is set by the 1-ring neighborhood of a vertex and its normal, therefore we have the representation of the discrete frame of n_k^i in the coordinates of the frame of i. □



(a) (b) (c) (d) (e)

Figure 3: Examples of basic editing operations. (a) Rotation of the handle discrete frame. The top curve is the original curve. We constrain the leftmost discrete frame (red) to remain the same and rotate the rightmost discrete frame by 90°. (b) The original circle model. (c) Uniform scaling ($\times 2$). (d) Uniform shrink constraint ($\times 0.3$). (e) Shear constraint. We constrained the discrete frames and the positions of the brown vertices to remain the same, and placed the editing constraint on the red discrete frames.



(a) (b) (c) (d) (e)

Figure 4: (a) The original *Armadillo* model (172974 vertices). (b) Scaling ($\times 3$) a few discrete frames on the nose, while fixing the body. (c) Scaling the same frames by 2.5 while fixing only the legs. (d–e) Bending the *Bunny*'s head while fixing the rear part. Note the preservation of the fur details.

The set of equations (1) forms an over-determined sparse linear system. To argue that the discrete forms indeed give rise to a representation of the mesh up to rigid motion, we point out the following two arguments.

Theorem 3.2. *Given an initial discrete frame at an arbitrary vertex i_0, $(\mathbf{b}_1^{i_0}, \mathbf{b}_2^{i_0}, \mathbf{N}^{i_0})$, such that the triplet is a right-hand orthonormal basis, and given that the first and second discrete forms with the orientation bits are taken from an existing mesh, there exists a* unique *solution to the set of equations (1) such that vertex i_0 has the initial discrete frame, and this unique solution constitutes the original surface discrete frames (rotated to match the initial discrete frame at vertex i_0).*

Theorem 3.3. *Given the solution to the discrete surface equations, namely, the discrete frames at each vertex, there exists a unique embedding of the vertices of the mesh in $\mathbb{R}^3$ up to a translation, such that the resulting mesh has the given discrete forms and discrete frames. In other words, the mesh is uniquely determined, up to translation, by its discrete frames and the coefficients of its discrete forms.*

For the proof of Theorem 3.2, note that the existence of a solution is obvious (the original discrete frames of the mesh rotated to fit the given initial condition). For uniqueness, note that by Eq. 2, fixing a single discrete frame uniquely defines all others.

To prove Theorem 3.3, we reconstruct the *geometry* of the mesh from the original discrete form coefficients and the discrete frames we got by solving Eq. 1. We construct the following difference equations:

$$\hat{\mathbf{x}}_k^i = \widetilde{\mathbf{x}}_k^i + \widetilde{L}_k^i \mathbf{N}^i, \qquad i \in V, \;\; k = 1, \ldots, d_i. \tag{4}$$

Since we have the discrete frame at vertex i and the first discrete form, we can find $\widetilde{\mathbf{x}}_1^i$ as $\widetilde{\mathbf{x}}_1^i = \mathbf{b}_1^i (\widetilde{g}_{1,1}^i)^{1/2}$. We also have $\mathbf{N}^i$ and thus we can find all $\widetilde{\mathbf{x}}_k^i$, $2 \le k \le d_i$ (see the proof of Lemma 2.1). Therefore the right-hand side of the above equations is known. Next, note that $\hat{\mathbf{x}}_k^i = \hat{\mathbf{x}}^j - \hat{\mathbf{x}}^i$ where j is the kth neighbor of vertex i. The unknowns $\hat{\mathbf{x}}^i$, $i \in V$, are the positions we are looking for. Combining the above with the proof of Lemma 2.1 and Eq. 4, we get:

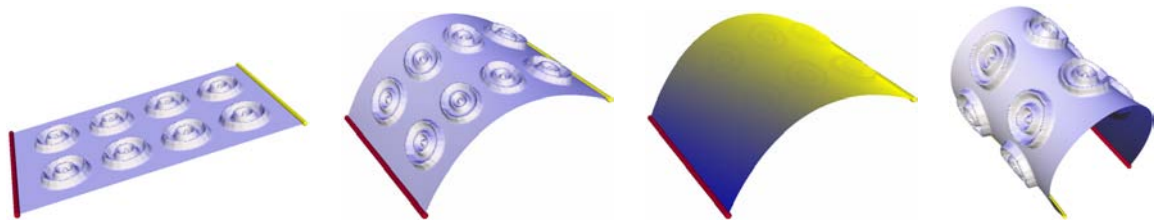
$$\hat{\mathbf{x}}^j - \hat{\mathbf{x}}^i = \widetilde{\mathbf{x}}_k^i + \widetilde{L}_k^i \mathbf{N}^i = \langle \widetilde{\mathbf{x}}_k^i, \mathbf{b}_1^i \rangle \, \mathbf{b}_1^i + \langle \widetilde{\mathbf{x}}_k^i, \mathbf{b}_2^i \rangle \, \mathbf{b}_2^i + \widetilde{L}_k \mathbf{N}^i, \qquad \forall (i,j) \in E \tag{5}$$

where the coefficients $\langle \widetilde{\mathbf{x}}_k^i, \mathbf{b}_k^i \rangle$ are computed from the original discrete forms. These equations obviously have a unique solution which is the original mesh, up to translation. This proves Theorem 3.3.

It should be noted that when the mesh is edited (Section 4), more constraints are added to the surface equations (1) and (5). Although it is clear that the reconstruction will produce a new triangular mesh, it is not guaranteed that self-intersection will not occur. The user can always define an extreme deformation constraint for which self-intersection would be inevitable.

3.1 Mesh representation

As proved above, one can represent the mesh (up to rigid transformation) using the first and second discrete form coefficients with orientation bits. Below we summarize how to construct this representation. Given a mesh in Cartesian coordinates, we orthogonally



(a) (b) (c) (d)

Figure 5: Smooth rotation of discrete frames. In (a), the original model is shown; the red part is fixed and the yellow part serves as a handle. (b) The result of constraining a 90° rotation about the yellow axis on the discrete frames of the handle. (c) The pseudocolor visualizes the gradual change of the discrete frames; the color coding corresponds to the Frobenius norm of the difference between the original discrete frame and the discrete frame after the editing operation. Note that regular shading due to the light source was disabled here, thus the smooth gradient of the Frobenius norm coloring shows that the rotations of the discrete frames vary smoothly on flat parts of the model as well as on the details. (d) Same rotation applied twice more.

project the edges emanating from each vertex i onto the tangent plane T_iM, which results in the following decomposition:

$$\hat{\mathbf{x}}_k^i = \hat{\mathbf{x}}_k^i - \langle \hat{\mathbf{x}}_k^i, \mathbf{N}^i \rangle \mathbf{N}^i + \langle \hat{\mathbf{x}}_k^i, \mathbf{N}^i \rangle \mathbf{N}^i = \widetilde{\mathbf{x}}_k^i + \widetilde{L}_k^i \mathbf{N}^i.$$

The orthogonal component represents the second discrete form, that is, $\widetilde{L}_k^i = \langle \hat{\mathbf{x}}_k^i, \mathbf{N}^i \rangle$. The first discrete form is represented by the lengths of the projected edges and angles between adjacent projected edges. This sums to $3d_i$ scalars for each vertex: $2d_i$ scalars for the first discrete form and d_i scalars for the second discrete form. These scalars are rigid motion invariant and describe only the differential properties around each vertex. In addition, we need the orientation bits O_k^i, as defined in Section 2.

Note that this differential representation is not intended to be compact. On the contrary, it contains redundant information, since the discrete surface equations (1) form an over-determined system, where at each vertex we hold enough information to reconstruct the mesh in *any* direction.

To reconstruct the mesh given its discrete form coefficients, an initial discrete frame and a position of one vertex in space:

- Construct the discrete surface equations (1):
  - For each vertex i and its k-th neighbor n_k^i:
    - Solve Eq. 3 to obtain the normal $N^{n_k^i}$ (Theorem 3.1) in the coordinates of the local frame of i;
    - Use $\hat{x}_k^i$ and $N^{n_k^i}$ to find the discrete frame of vertex n_k^i expressed in the frame of vertex i (Lemma 2.1);
    - Extract the coefficients of Eq. 1 for i and $j = n_k^i$.
- Add the given initial discrete frame as an additional equation;
- Solve the augmented linear system in the least-squares sense to obtain the discrete frames;
- Construct the geometry difference equations (5):
  - Use the discrete frames obtained above and the *original* discrete forms to calculate the right-hand side of Eq. 5;
- Add the given initial vertex position as an additional equation;
- Solve the augmented linear system in the least-squares sense to get the positions of the vertices.

In the next section, we describe an editing mechanism that uses the above steps.

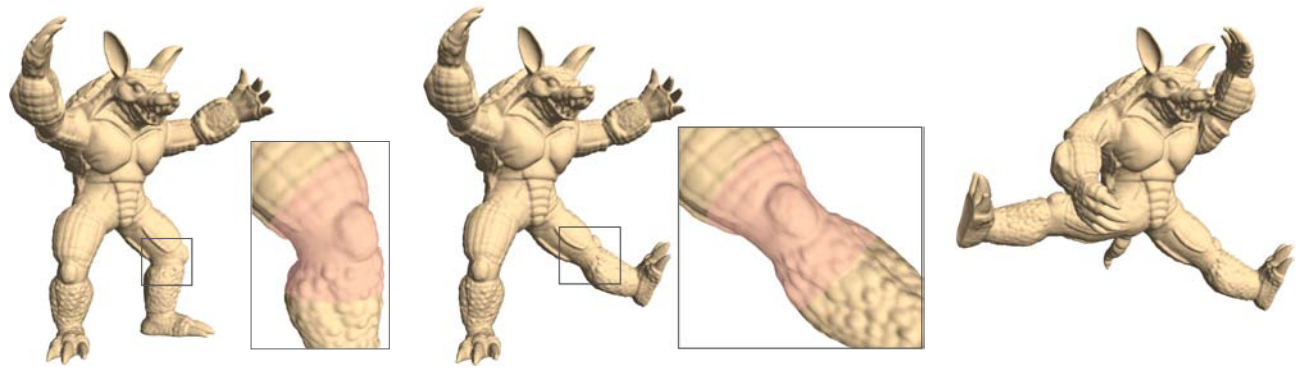
4 Mesh editing

Our interactive mesh editing mechanism is based on the surface representation and the reconstruction process of the geometry introduced in Section 3. The editing operation is applied by adding linear constraints to the linear systems defined by Eqs. 1 and 5. As described in Section 3, the reconstruction of the geometry of the mesh from the discrete forms consists of two main stages:

- Reconstructing the discrete frames at each vertex by solving the discrete surface equations.
- Reconstructing the geometry at each vertex from the discrete forms and the discrete frames.

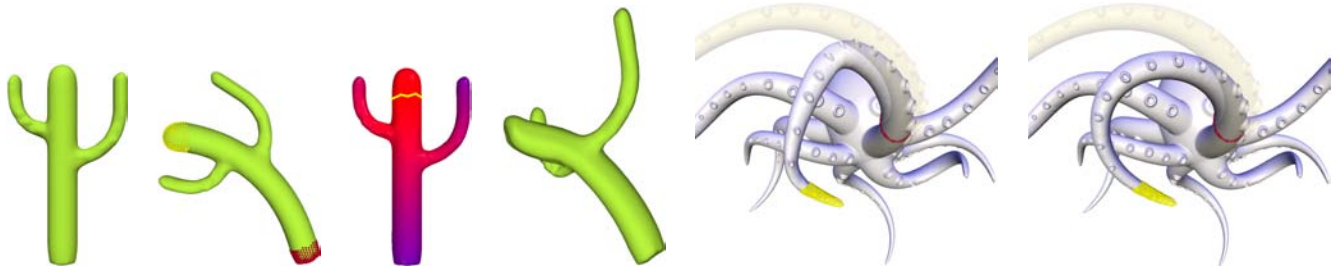
In the first stage, we solve the discrete surface equations (1), which form an over-determined sparse linear system. The coefficients of the equations can be computed in two equivalent ways: either as described in Section 3.1 using only the discrete forms, or directly from the original mesh and the vertex normals by Eq. 2. In this case, since the geometry of the *original* mesh is known, the latter way is somewhat simpler. Editing is applied by constraining more discrete frames and solving in the least-squares sense. When constraining more than one discrete frame, there is no guarantee of an exact solution; hence we aim at obtaining the mesh which is as close as possible to satisfying the prescribed differential relations between the discrete frames under the posed conditions. Theorem 3.2 guarantees a unique solution given an initial discrete frame at some vertex, hence, the corresponding system in (1) with the added initial condition has full rank, and thus there exists a unique least-squares solution to the over-determined editing system. In the second stage, some spatial constraints on the geometry are added to Eq. 5, and the system is solved to reconstruct the mesh geometry. As before, Theorem 3.3 guarantees a unique least-squares solution of the system with the added spatial conditions.

In our interactive editing system, we adopt the modeling metaphor established in previous works [Kobbelt et al. 1998; Botsch and Kobbelt 2004; Sorkine et al. 2004; Yu et al. 2004]. Namely, the user defines a region of interest (ROI) and a handle, which is some subset of the ROI vertices. The editing is performed by manipulating the handle and reconstructing the surface by applying the new constraints. The discrete frames and the positions of the vertices on the boundary of the ROI are constrained to remain the same, and surface reconstruction is performed on the ROI only, leaving the rest of the mesh unchanged. For speedup, the sparse matrices involved in the reconstruction are factored once per ROI (using sparse Cholesky decomposition [Toledo 2003]), and each time the constraints on the handle change, only an update of the right-hand side of the system and a solution by back-substitution is needed. Note



(a) (b) (c)

Figure 6: Examples of editing operators. (a) The original *Armadillo* model. (b) rotation operator applied to the knee. The red region denotes the free vertices; the vertices of the foot are used as the handle. The close-ups show the details around the knee from an additional angle. Note that the details are well-preserved after the rotation. (c) Combination of several edits.



(a) (b) (c) (d) (e) (f)

Figure 7: Comparison of our editing method to Poisson Editing and Laplacian Editing. The original *Cactus* model is shown in (a). The result of editing with our method is displayed in (b), where the red spheres mark the static anchors and the yellow spheres mark the handle vertices. (c) The "strength field" resulting from marking a handle curve in the Poisson Editing application. Red denotes maximum and blue denotes minimum values. (d) The result achieved with Poisson Editing when rotating and translating the handle in approximately the same manner as in (b). Note that since the strength field is weaker at the branches (which are geodesically far from the handle), the branches do not receive the same amount of rotation as the trunk. To compare to Laplacian Editing, we deformed the arm of the *Octopus* (the original surface is rendered in yellow). (e) shows the result of Laplacian Editing, whereas (f) is our result. It is evident that our new method handles large rotations better.

that we always use the original discrete forms of the mesh given the ROI, and thus the system matrices remain fixed. Our editing application enables the user to edit ROIs containing tens of thousands of vertices at interactive frame rates.

Different constraints on the handle result in various editing effects. The basic operation is a linear transformation A on the discrete frames of the handle. This can be done by manipulating an arcball-like control. For all the handle's vertices i we add the following constraints to the system:

$$\begin{aligned} \mathbf{b}_l^i &= A(\check{\mathbf{b}}_l^i), \qquad l = 1,2 \\ \mathbf{N}^i &= A(\check{\mathbf{N}}^i), \end{aligned}$$

where $(\check{\mathbf{b}}_1^i, \check{\mathbf{b}}_2^i, \check{\mathbf{N}}^i)$ denotes the discrete frame of vertex i in the original mesh. A simple example is shown in Figure 5, where A is a rotation. In 5(b) one can see that the surface bends according to the rotation of the handle, and the details are preserved and maintain their relative orientation with respect to the surface. This happens because our surface representation is rotation-invariant, and thus the local frames of the ROI are rotated in the reconstruction.

Figures 3 and 4(a–c) show other types of transformations A: uniform scale and shear. In all these cases the reconstruction process preserves the differential representation as much as possible under the constraints. More examples are shown in Figures 4(d–e), 6, 9, 13, where it can be seen that the details are preserved under the editing operations, including sharp features. The above examples contain transformation constraints on the local frames as well as translation constraints on the vertex positions. We apply the same transformation A and the same translation to all vertices of the handle. It should be noted that extreme rotation constraints (e.g., by 180 degrees) on the discrete frames may cause unintuitive results. However, less ambiguous rotation constraints can be easily performed; see Figure 10, where the top of the bar is rotated by 170 degrees.

Note that the solution of the augmented system for the discrete frames does not guarantee that the resulting discrete frames will be orthonormal triplets. On the contrary, with scaling and shearing editing operations A, the system produces scaled and sheared frames. When A is orthogonal, the frames tend to stay orthonormal in practice. In this case, it is possible to normalize each frame vector of the solution. It is important to mention that the orthogonality of the discrete frames is not a necessary condition for the discrete surface equations framework to work.

We compare the capabilities of our editing mechanism to other recent approaches in Figure 7. The Poisson editing system, proposed by Yu et al. [2004], propagates the transformation of the handle to the rest of the ROI using geodesic distances as weights. As a result, protruding features that are "far" from the handle get less

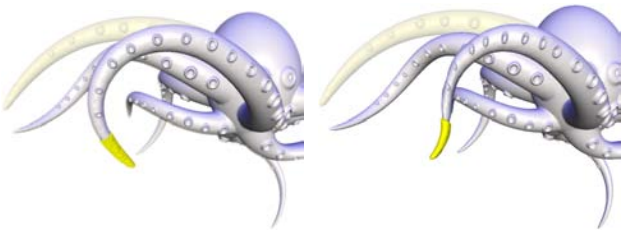


Figure 8: The positional constraints on the vertices and the transformation constraints on the local frames need to be compatible. The left image shows the result of compatible constraints, where the local frames are first appropriately rotated and then the handle is translated (the original mesh is shown in light yellow). The right image shows incompatible constraints: the handle is only translated. As a result, the local frames are not rotated appropriately and the deformation looks less natural.

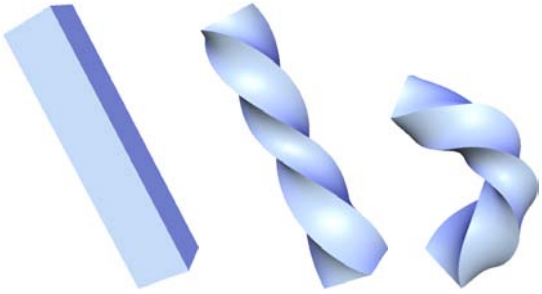


Figure 9: Rotating the top vertices on the bar while fixing the bottom; then bending the top. The editing preserves the sharp edges.

rotation, and the editing result might look unnatural. Observe the *Cactus* model in Figure 7(a–d), where the tips of the branches are further from the handle than their bases, which causes the branches to almost stay in place when the handle is rotated. In contrast, our rotation-invariant representation yields a plausible editing result, naturally rotating the branches. The Laplacian editing approach [Sorkine et al. 2004] handles local rotations in an implicit manner in the same system that solves for the vertex positions. However, to pose the editing in a linear way, 3D rotations must be linearized. This causes errors when the required rotation of the handle is large. See Figure 7(e–f), where a large rotation of the *Octopus* arm looks smooth with our approach, whereas the Laplacian editing result is "broken" into several parts of smaller rotation.

An important observation is that the linear transformation constraints are added to the discrete surface equations (1), while the positional constraints are added to the system (5), so that actually, the discrete frames are determined before the positional constraints are considered. The positional and the linear transformation constraints of the handle should be *compatible*. Figure 8 shows an example of compatible constraints and incompatible ones. In future work, we would like to consider solving the two sets of equations again to obtain a better correspondence between the linear and the positional constraints. In addition, it would be interesting to infer the transformation of the discrete frames from the translational movement of the handle.

By solving the augmented surface equations in the least-squares sense, we minimize the differences between the coefficients in the discrete surface equations (1) of the original mesh and the corresponding coefficients of the edited mesh. We argue that such a solution strives to keep the *curvatures* of the edited mesh as close as

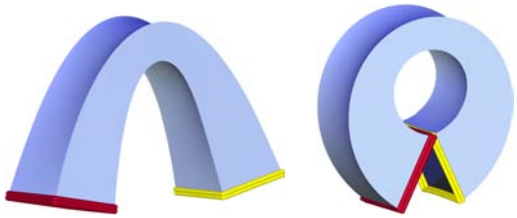


Figure 10: Example of extreme rotation. Left: the handle (in yellow) is rotated by 170°. The red vertices mark the static constraints. Right: rotation by 315° performed in three steps.

possible to the original surface curvatures. Indeed, in the smooth case, if we consider a frame field $\{(\mathbf{b}_1, \mathbf{b}_2, \mathbf{N})\}$, where $\mathbf{N}$ is the surface normal, a curve on the surface with arc-length parameter s yields the following equations [Guggenheimer 1963]:

$$\frac{d}{ds}\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\\ \mathbf{N}\end{pmatrix} = \begin{pmatrix}0 & k_g & k_n\\ -k_g & 0 & t_r\\ -k_n & -t_r & 0\end{pmatrix}\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\\ \mathbf{N}\end{pmatrix},$$

where k_g is the geodesic curvature, k_n is the normal curvature and t_r is the relative torsion. In our discrete case, the coefficients of the surface equations (1) approximate the curvature quantities, as in the matrix above. This can be shown by dividing both sides of Eq. 1 by the length of the corresponding edges. Hence, when the deformation constraint A does not change the edge lengths (when A is orthogonal), minimizing the difference between the coefficients in Eq. 1 approximately minimizes the curvature differences between the original and the edited mesh. Of course, when A is strongly distorting the edge lengths, the curvature error is large, which is unavoidable in this case.

4.1 Shape interpolation

Our rigid-motion invariant surface representation is readily suitable for interpolation between different shapes. It has been shown that a proper shape interpolation should minimize redundant distortions, or in other words, that it should be as rigid as possible [Cohen-Or et al. 1998; Alexa et al. 2000]. This can be achieved by applying a rigid-elastic decomposition to the transformation. The elastic component is defined as the residual of applying the rigid transformation first. Such decomposition minimizes the elastic component, and consequently, the redundant distortion. This effect comes for free with rigid-motion invariant representation, since the rigid part is factored out.

Assume M_1 and M_2 are two meshes with identical connectivity and different geometries, where the geometries are represented by the discrete forms. Simple linear interpolation between the discrete forms coefficients of M_1 and M_2 yields the "as-rigid-as-possible" effect. To preserve the property that the sum of angles in the tangent plane around each vertex is 2π, we interpolate the angles themselves, rather than their cosines ($\widetilde{g}_{12}$). Our method can be regarded as a generalization of the 2D interpolation technique of [Sederberg et al. 1993], which interpolates the edge lengths and the angles between consecutive edges of polygonal curves.

Examples of such shape interpolation are demonstrated in Figures 11 and 12, where we show a number of intermediate shapes for different parameters t. It is evident that linear interpolation between our rigid-invariant representations yields intuitive rotations of the source shape towards the target shape, whereas, for comparison, the linear interpolation clearly does not accommodate local rotations. Note that linear interpolation of relative coordinates, such as the

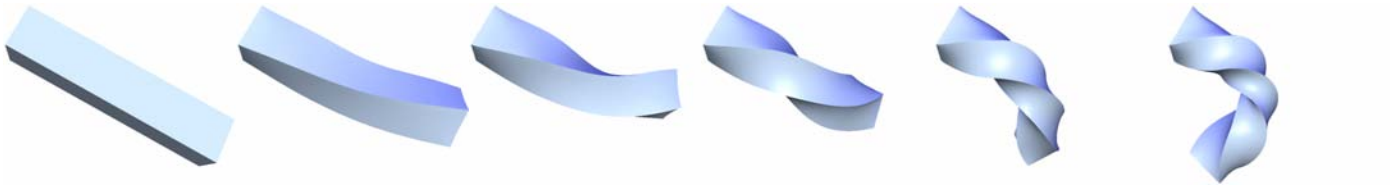


Figure 11: Linear shape interpolation sequence of the *Bar* mesh (1600 vertices) using our representation. Note the natural rotation of the in-between shapes.

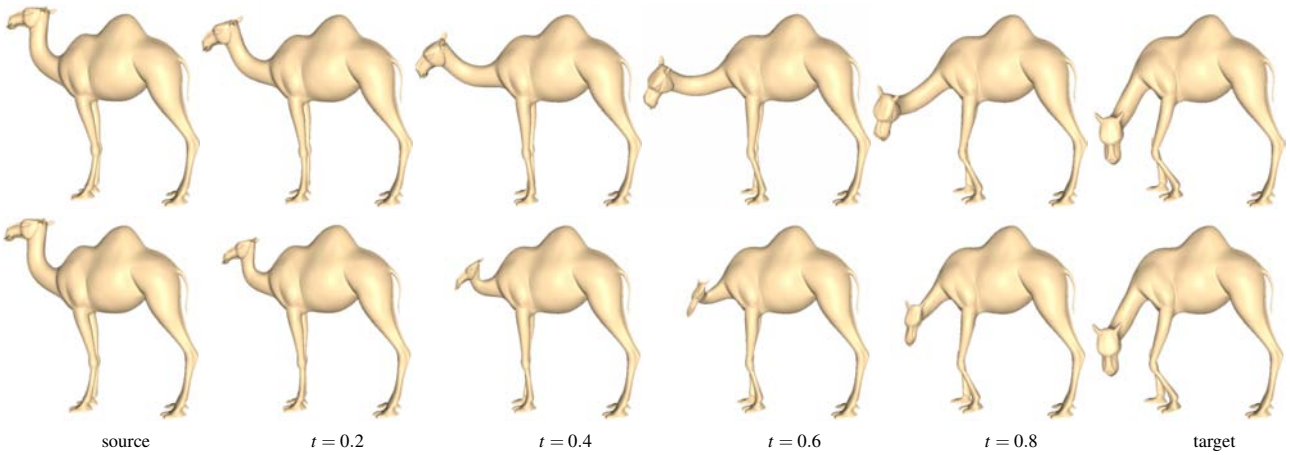


Figure 12: Shape interpolation sequence of the *Camel* mesh, 39074 vertices. The top row shows the results of linear interpolation of the discrete forms, and the bottom row shows linear interpolation of the Cartesian coordinates. Clearly, the interpolation using our rigid motion invariant shape representation (top row) yields natural rotations of the in-between shapes.

Laplacian coordinates [Alexa 2003; Sorkine et al. 2004], does not yield different results, since the coordinates are linearly dependent on the Cartesian coordinates. Furthermore, our shape interpolation does not involve any non-linear optimizations, neither in the interpolation nor in the reconstruction. We believe that our results have the same quality as the advanced morphing methods, such as [Alexa et al. 2000; Xu et al. 2005], where the first requires meshing the interior of the model, and the second involves non-linear interpolation of the Poisson mesh representation.

5 Discussion

The mesh representation presented in the previous sections has an interesting link to classical differential geometry in the following sense. Classical differential geometry defines two quadratic forms on the surface, that is, the first and second fundamental forms, which locally describe the tangential and normal components of the surface. Combining these two forms furnishes a good local approximation of the surface. Furthermore, Bonnet theorem ensures that locally, the surface can be *reconstructed* given the fundamental forms which hold some compatibility conditions (Gauss-Codazzi-Mainardi equations, see [Stoker 1989]). The Bonnet theorem is constructive and consists of two stages: first, a first-order linear PDE describing the changes of the local Frenet frames on the surface is solved, and second, the frames are integrated, which leads to a parameterization of the surface. Our algorithm has a very similar concept: we also write a linear system which describes the changes of the local frames and then "integrate" the changes to solve for the geometry. The reason we use a least-squares solution instead of incrementally "growing" a solution from an initial frame is that the latter approach equally distributes the error across the mesh.

6 Conclusions

We have presented a discrete framework on triangular meshes that furnishes a rigid motion invariant representation of the mesh. We obtain a linear surface reconstruction method by taking two steps: first, solving for the local frames of the surface, and then solving for the positions of the vertices. The reconstructed mesh can be regarded as the natural global shape defined by local descriptors defined at each vertex. Our framework enables detail-preserving surface editing and shape interpolation.

In future work, it might be interesting to experiment with other interactive tools that directly manipulate the local frames on the surface. We would also like to improve the co-existence of constraints on the local frames and the vertex positions, as mentioned in Section 4. Another interesting application could be to use our representation for *detail* editing, such as detail enhancement and other modifications.

Acknowledgements

We are grateful to Alon Lerner for helping with the video, to Kun Zhou for providing the Poisson Editing software, to Andrew Nealen for proofreading and to Gershon Elber and Leif Kobbelt for their valuable comments and suggestions. The *Bunny* and *Armadillo* are courtesy of of Stanford University, the *Octopus* is courtesy of Mark Pauly. This work was supported in part by grants from the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities) and the Israeli Ministry of Science.

Figure 13: Another example of interactive editing. We folded the *Gargoyle*'s wings and bent its head.

References

ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press/Addison-Wesley Publishing Co., 157–164.

ALEXA, M. 2003. Differential coordinates for local mesh morphing and deformation. *The Visual Computer 19*, 2, 105–114.

BENDELS, G. H., AND KLEIN, R. 2003. Mesh forging: editing of 3D-meshes using implicitly defined occluders. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Eurographics Association, 207–217.

BOTSCH, M., AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. In *Proceedings of ACM SIGGRAPH 2004*, ACM Press, 630–634.

CAZALS, F., AND POUGET, M. 2003. Estimating differential quantities using polynomial fitting of osculating jets. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Eurographics Association, 177–187.

COHEN-OR, D., LEVIN, D., AND SOLOMOVICI, A. 1998. Three-dimensional distance field metamorphosis. *ACM Trans. Graph. 17*, 2, 116–141.

COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted delaunay triangulations and normal cycle. In *Proceedings of the 19th annual symposium on computational geometry*, ACM Press, 312–321.

GUGGENHEIMER, H. 1963. *Differential Geometry*. McGraw–Hill, New York.

GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. In *Proceedings of ACM SIGGRAPH 99*, ACM Press/Addison-Wesley Publishing Co., 325–334.

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

LEE, S. 1999. Interactive multiresolution editing of arbitrary meshes. *Computer Graphics Forum (Proceedings of Eurographics 1999) 18*, 3, 73–82.

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.

LIPMAN, Y. 2004. Differential geometry of piecewise-linear surfaces. Tech. rep., Tel Aviv University, December.

MEEK, D. S., AND WALTON, D. J. 2000. On surface normal and Gaussian curvature approximations given data sampled from a smooth surface. *Computer Aided Geometric Design 17*, 6, 521–543.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *Proceedings of VisMath*.

O'NEILL, B. 1969. *Elementary Differential Geometry*. Academic Press, New York.

SEDERBERG, T. W., GAO, P., WANG, G., AND MU, H. 1993. 2-D shape blending: an intrinsic solution to the vertex path problem. In *Proceedings of ACM SIGGRAPH 93*, 15–18.

SHEFFER, A., AND KRAEVOY, V. 2004. Pyramid coordinates for morphing and deformation. In *Proceedings of 2nd International Symposium on 3D Data Processing, Visualization, and Transmission*, IEEE Computer Society Press, 68–75.

SORKINE, O., LIPMAN, Y., COHEN-OR, D., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Eurographics Association, 179–188.

STOKER, J. J. 1989. *Differential Geometry*. Wiley, New York.

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/`, Sept.

XU, D., ZHANG, H., WANG, Q., AND BAO, H. 2005. Poisson shape interpolation. In *ACM Symposium on Solid and Physical Modeling*, to appear.

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. In *Proceedings of ACM SIGGRAPH 2004*, ACM Press, 641–648.

ZAYER, R., RÖSSL, C., KARNI, Z., AND SEIDEL, H.-P. 2005. Harmonic guidance for surface deformation. In *Computer Graphics Forum, Proceedings of Eurographics 2005*, to appear.

ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1997. Interactive multiresolution mesh editing. In *Proceedings of ACM SIGGRAPH 97*, ACM Press/Addison-Wesley Publishing Co., 259–268.