Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Quasi-angle-preserving mesh deformation using the least-squares approach[*]

Gang XU[†1], Li-shan DENG[1], Wen-bing GE[2], Kin-chuen HUI[2], Guo-zhao WANG[3], Yi-gang WANG[†‡1]

([1]*Department of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, China*)
([2]*Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong, China*)
([3]*Department of Mathematics, Zhejiang University, Hangzhou 310027, China*)
[†]E-mail: xugangzju@gmail.com; yigang.wang@hdu.edu.cn

Received Mar. 21, 2014; Revision accepted Aug. 5, 2014; Crosschecked Aug. 19, 2014

Abstract: We propose an angle-based mesh representation, which is invariant under translation, rotation, and uniform scaling, to encode the geometric details of a triangular mesh. Angle-based mesh representation consists of angle quantities defined on the mesh, from which the mesh can be reconstructed uniquely up to translation, rotation, and uniform scaling. The reconstruction process requires solving three sparse linear systems: the first system encodes the length of edges between vertices on the mesh, the second system encodes the relationship of local frames between two adjacent vertices on the mesh, and the third system defines the position of the vertices via the edge length and the local frames. From this angle-based mesh representation, we propose a quasi-angle-preserving mesh deformation system with the least-squares approach via handle translation, rotation, and uniform scaling. Several detail-preserving mesh editing examples are presented to demonstrate the effectiveness of the proposed method.

Key words: Mesh deformation, Angle-based representation, Detail-preserving, Least-squares approach
doi:10.1631/jzus.C1400103 **Document code:** A **CLC number:** TP391

1 Introduction

How to modify and edit the shape of a triangle mesh is an important topic in numerical solving and geometric modeling (Zhou and Li, 2013). As a triangular mesh has been a popular representation for computational physics, mesh deformation has received extensive attention in recent years (Luke *et al.*, 2012). Mesh deformation methods based on differential coordinates (e.g., Laplacian coordinates and the gradient field) have been proposed recently (Alexa, 2003; Sorkine *et al.*, 2004; Lipman *et al.*, 2004; 2005; Zhou *et al.*, 2004; Yu *et al.*, 2005; Au *et al.*, 2006). These editing systems represent local features using differential coordinates defined in a global coordinate system. The simplest form of differential coordinates is the Laplacian coordinates. It uses the deviation of a vertex from the centroid of its neighbors to encode the local details of the mesh surface. From the theory of discrete differential geometry, various angles on the mesh surfaces are also important information to encode the local geometric details. For example, curvature metrics on a triangular mesh are often estimated from the angle information. However, as we know, there are few works on using angle information to encode the local details and develop mesh editing tools.

In this paper, we introduce an angle-based mesh representation, which is invariant under translation,

[‡] Corresponding author

[*] Project supported by the National Natural Science Foundation of China (Nos. 61472111, 61272300, and 51475309), the Defense Industrial Technology Development Program (No. A3920110002), the Open Project Program of the State Key Lab of CAD&CG, Zhejiang University (No. A1406), the Zhejiang Provincial Natural Science Foundation (No. Z1091077), the Direct Grant from the Chinese University of Hong Kong (No. 2050492), and the Research Grants Council of the Hong Kong Special Administration Region, China (No. 412913)

rotation, and uniform scaling. Here uniform scaling means that the same ratio is applied along x, y, and z axes. Angle-based mesh representation describes the mesh by its local angle information between edges. Using this representation, we develop quasi-angle-preserving mesh deformation techniques, which preserve the intrinsic geometry information of the surface as much as possible with the constraints of the modeling operations.

Reconstructing mesh geometry from locally defined quantities is a fundamental mechanism of our algorithm, which consists of three steps. First, we use the equations of the difference between the length of vertex to compute the scalar part of each vertex (the concept of 'length of vertex' will be defined in Definition 2). Then, we use the relative frame coefficient representation of the local frame to compute a global least-squares fitting with the user-specified local frames of the handle. Finally, we use the differential representation of the mesh to compute a global least-squares fitting constrained to the user-specified positions of the handle. The presented examples illustrate that the angle information on the mesh surface is well preserved.

2 Related work

1. Free-form deformation: Free-form deformation (FFD) has been widely employed as a shape modification tool in commercial software. The basic idea of FFD is to embed and parameterize the target object in a spatial control lattice. The deformed surface is obtained through the modification of the control points. The FFD method was first introduced by Sederberg and Parry (1986), and extended to other forms with different kinds of splines (Coquillart, 1990; Hsu *et al.*, 1992; MacCracken and Joy, 1996; Xu *et al.*, 2008). Botsch and Kobbelt (2004) introduced an intuitive free-form modeling framework that uses custom tailored basis functions to define interpolations between the handle regions and fixed outside regions. Curve-based methods (Lazarus *et al.*, 1994; Singh and Fiume, 1998; Nealen *et al.*, 2005; Xu *et al.*, 2013) directly attach mesh surfaces to curves for achieving deformation results.

2. Multi-resolution editing: By decomposing the geometric details into several levels, the multi-resolution approaches (Zorin *et al.*, 1997; Kobbelt *et al.*, 1998; Guskov *et al.*, 1999; Boier-Martin *et al.*, 2004) enable detail-preserving deformations, in which the deformation is formed as displacements in the local coordinate frame, and the coarse mesh is used as control lattices. In the editing process, the user transforms the details according to the changes in the local frames on the low-level mesh. Hence, the definition of the local frames and the accuracy of the detail-reconstruction method are important issues in multi-resolution editing.

3. Differential coordinates editing: Recently, the local features on mesh geometry were encoded by differential coordinates for the mesh editing framework (Alexa, 2003; Lipman *et al.*, 2004; 2005; Sorkine *et al.*, 2004; Zhou *et al.*, 2004; Yu *et al.*, 2005). By these methods, the users can directly specify the positions or normal directions on the mesh, called the handles, and the rest of the surface is reconstructed by solving a linear system to minimize the shape distortion. Poisson meshes (Yu *et al.*, 2005) manipulate gradients of the coordinate functions on the mesh using local transformation, and the mesh surface is reconstructed from the Poisson equation. Laplacian coordinates are another metric to represent surface details (Alexa, 2003; Lipman *et al.*, 2004; Sorkine *et al.*, 2004; Zhou *et al.*, 2004). Dual Laplacian coordinates are employed for mesh deformation (Au *et al.*, 2006) and mesh morphing (Hu *et al.*, 2007) to produce natural deformation results. Another surface representation is rotation-invariant coordinates (Lipman *et al.*, 2005), consisting of the tangential part and the normal part, which are invariant under rotation and translation in the first and second discrete forms, respectively. However, the above discrete forms are not scale- or shear-invariant.

In this paper, we propose a new mesh editing framework based on angle-based mesh representation, which consists of the angles between edges and the angles between the normals and edges and is invariant under rotation, translation, and uniform scaling. The new approach can produce more natural editing results than previous methods.

3 Angle-based mesh representation

3.1 Mesh representation with angle quantities

Angle-based mesh representation consists of angle quantities, which are invariant under translation, rotation, and uniform scaling, and contains enough information to reconstruct the mesh uniquely.

Let $G = (V, E)$ be a triangular mesh. V denotes the set of vertices on the mesh; that is, $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n\}$ describes the absolute Cartesian coordinates of the vertices in $\mathbb{R}^3$. E denotes the set of edges; that is, each edge connecting vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ can be represented as a pair (i, j) in E. We use $P_i = (V_i', E_i')$ to denote the geometry related to $\boldsymbol{v}_i$. V_i' denotes the set of vertices that are projected on the tangent plane of $\boldsymbol{v}_i$ by $\boldsymbol{v}_i$ and its 1-ring neighborhood. We use $\boldsymbol{v}_k^i$ to denote the kth neighbor of vertex $\boldsymbol{v}_i$, while $\boldsymbol{v}_k^{i'}$ to denote the kth neighbor of vertex $\boldsymbol{v}_i$ on its tangent plane.

Definition 1 The following notations are used to define the angle quantities (Fig. 1):

1. α_k^i, which are the angles between vectors $\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i$ and $\boldsymbol{v}_{k+1}^{i'} - \boldsymbol{v}_i$.

2. β_k^i, which are the angles between vectors $\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i$ and $\boldsymbol{v}_{k+1}^{i'} - \boldsymbol{v}_k^{i'}$.

3. γ_k^i, which are the angles between the normal $\boldsymbol{N}_i$ at vertex $\boldsymbol{v}_i$ and the vector $\boldsymbol{v}_k^i - \boldsymbol{v}_i$.

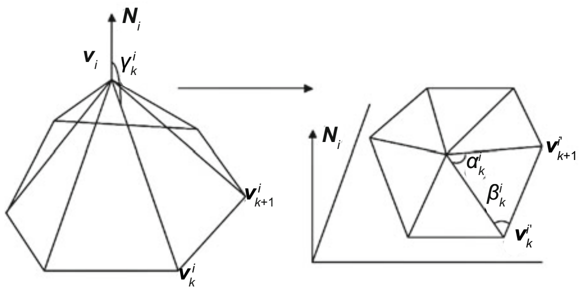


Fig. 1 Angle quantities of $\boldsymbol{v}_i$

Remark 1 The estimation of normal vector $\boldsymbol{N}_i$ at vertex $\boldsymbol{v}_i$ will affect the performance of the mesh-editing process. To compute the vertex normal of triangular meshes more accurately, we will adopt the weight-based method to estimate the normal vector. Suppose that the set of triangles that contain vertex $\boldsymbol{v}_i$ is denoted by F_i. If triangle f_k is incident to $\boldsymbol{v}_i$, then f_k is in F_i. The unit normal vector $\boldsymbol{N}_i$ at vertex $\boldsymbol{v}_i$ on the mesh is estimated in the following way:

$$\boldsymbol{N}_i = \frac{\sum_{f_k \in F_i} w_k \boldsymbol{N}_{f_k}}{\|\sum_{f_k \in F_i} w_k \boldsymbol{N}_{f_k}\|},$$

where $\boldsymbol{N}_{f_k}$ is the unit normal vector of triangle f_k, and the weights w_k are defined as in Chen and Wu (2004):

$$w_k = \frac{1}{\|\boldsymbol{g}_k - \boldsymbol{v}_i\|},$$

where $\boldsymbol{g}_k$ is the center of the triangle face f_k, determined as

$$\boldsymbol{g}_k = \sum_{\boldsymbol{v}_j \in f_k} \boldsymbol{v}_j / 3.$$

The angle quantities α_k^i, β_k^i, and γ_k^i describe the geometry of 1-ring neighborhood at vertex $\boldsymbol{v}_i$ up to translation, rotation, and uniform scaling, as explained in the following theorem:

Theorem 1 Given the angle quantities of vertex $\boldsymbol{v}_i$ and the normal at $\boldsymbol{v}_i$, the 1-ring neighborhood of vertex $\boldsymbol{v}_i$ can be computed by translation, rotation, and uniform scaling.

Proof Given the vertex $\boldsymbol{v}_i$, the direction of the first edge can be defined as

$$\boldsymbol{t}_i = \frac{\boldsymbol{v}_1^i - \boldsymbol{v}_i}{\|\boldsymbol{v}_1^i - \boldsymbol{v}_i\|},$$

the distance from $\boldsymbol{v}_i$ to its first neighborhood is denoted as l_i, the length of the kth edge on the tangent plane is denoted as $l_k^{i'}$, and $\theta_k^i = \pi - (\alpha_k^i + \beta_k^i)$. Then $l_1^{i'} = l_i \sin\gamma_1^i$. Since

$$\frac{l_{k+1}^{i'}}{l_k^{i'}} = \frac{\sin\beta_k^i}{\sin\theta_k^i} = \frac{\sin\beta_k^i}{\sin(\pi - (\alpha_k^i + \beta_k^i))} = \frac{\sin\beta_k^i}{\sin(\alpha_k^i + \beta_k^i)},$$

the relationship between $l_k^{i'}$ and $l_1^{i'}$ is computed as follows:

$$\begin{aligned}\frac{l_k^{i'}}{l_1^{i'}} &= \frac{l_k^i}{l_{k-1}^i}\frac{l_{k-1}^i}{l_{k-2}^i}\cdots\frac{l_2^i}{l_1^i} \\ &= \frac{\sin\beta_{k-1}^i}{\sin(\alpha_{k-1}^i + \beta_{k-1}^i)}\frac{\sin\beta_{k-2}^i}{\sin(\alpha_{k-2}^i + \beta_{k-2}^i)} \\ &\quad\cdots\frac{\sin\beta_1^i}{\sin(\alpha_1^i + \beta_1^i)} \\ &= \frac{\prod_{j=1}^{k-1}\sin\beta_j^i}{\prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)},\end{aligned}$$

which implies

$$l_k^{i'} = l_1^{i'}\frac{\prod_{j=1}^{k-1}\sin\beta_j^i}{\prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)}.$$

We obtain the distance from $\boldsymbol{v}_i$ to its kth neighborhood:

$$l_k^i = \frac{l_k^{i'}}{\sin\gamma_k^i} = l_i\frac{\sin\gamma_1^i\prod_{j=1}^{k-1}\sin\beta_j^i}{\sin\gamma_k^i\prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)}. \quad (1)$$

Since the length of $\boldsymbol{v}_k^i - \boldsymbol{v}_i$ is obtained, next we describe how to compute the direction of $\boldsymbol{v}_k^i - \boldsymbol{v}_i$. Set

$$\boldsymbol{n}_i = \frac{\boldsymbol{N}_i \times \boldsymbol{t}_i}{\|\boldsymbol{N}_i \times \boldsymbol{t}_i\|}$$

and $\boldsymbol{b}_i = \dfrac{\boldsymbol{v}_1^{i'} - \boldsymbol{v}_i}{\|\boldsymbol{v}_1^{i'} - \boldsymbol{v}_i\|} = \boldsymbol{n}_i \times \boldsymbol{N}_i$. $\{\boldsymbol{b}_i, \boldsymbol{n}_i, \boldsymbol{N}_i\}$ forms a right-hand orthogonal basis. Since the angle between vectors $\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i$ and $\boldsymbol{v}_1^{i'} - \boldsymbol{v}_i$ is $\sum_{j=1}^{k-1} \alpha_j^{i'}$,

$$\frac{\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i}{\|\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i\|} = \boldsymbol{b}_i \cos \sum_{j=1}^{k-1} \alpha_j^{i'} + \boldsymbol{n}_i \sin \sum_{j=1}^{k-1} \alpha_j^{i'}.$$

Then we obtain the unit direction of vector:

$$\frac{\boldsymbol{v}_k^{i} - \boldsymbol{v}_i}{\|\boldsymbol{v}_k^{i} - \boldsymbol{v}_i\|} = \frac{\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i}{\|\boldsymbol{v}_k^{i'} - \boldsymbol{v}_i\|} \sin \gamma_k^i - \boldsymbol{N}_i \cos \gamma_k^i.$$

Finally, we have

$$\begin{aligned}\boldsymbol{v}_k^i - \boldsymbol{v}_i &= l_k^i \frac{\boldsymbol{v}_k^{i} - \boldsymbol{v}_i}{\|\boldsymbol{v}_k^{i} - \boldsymbol{v}_i\|} \\ &= \frac{l_i \sin\gamma_1^i \prod_{j=1}^{k-1} \sin\beta_j^i}{\sin\gamma_k^i \prod_{j=1}^{k-1} \sin(\alpha_j^i + \beta_j^i)} \Bigg[\Bigg(\boldsymbol{b}_i \cos \sum_{j=1}^{k-1} \alpha_j^{i'} \\ &\quad + \boldsymbol{n}_i \sin \sum_{j=1}^{k-1} \alpha_j^{i'} \Bigg) \sin\gamma_k^i - \boldsymbol{N}_i \cos\gamma_k^i \Bigg]. \end{aligned} \tag{2}$$

As suggested by the reviewer, the proof of Theorem 1 can also be given by cone construction (Fig. 2). Starting from vertex $\boldsymbol{v}_k^i$, we can first construct a cone C_k^i with central axes $\boldsymbol{v}_k^i - \boldsymbol{v}_i$ and apex angle $2\alpha_k^i$; second, we construct a cone C_{k+1}^N with central axes $-\boldsymbol{N}_i$ and apex angle $2(\pi - \gamma_{k+1}^i)$. Then the unit vector of $\boldsymbol{v}_{k+1}^i - \boldsymbol{v}_i$ can be constructed as the intersection of cones C_k^i and C_{k+1}^N. Finally vertex $\boldsymbol{v}_{k+1}^i$ can be constructed by a uniform-scaling factor l_{k+1}^i, which can be computed according to Eq. (1).

Note that in Eq. (2), $\boldsymbol{v}_k^i$ is related to l_i, which is not uniform-scaling invariant. Next, we will describe that l_i can be represented in a uniform-scaling invariant way.

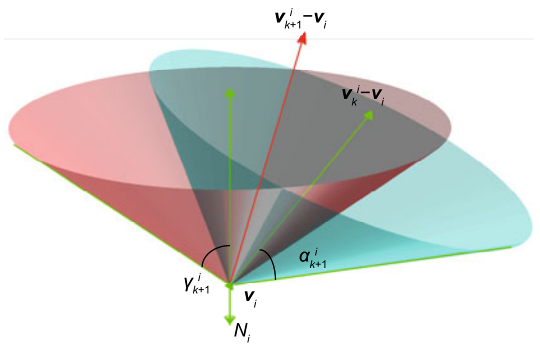

$\boldsymbol{v}_{k+1}^i - \boldsymbol{v}_i$
$\boldsymbol{v}_k^i - \boldsymbol{v}_i$
α_{k+1}^i
γ_{k+1}^i
$\boldsymbol{v}_i$
N_i

Fig. 2 Proof of Theorem 1 by cone construction

Definition 2 The length of $\boldsymbol{v}_i$ is defined as the distance from vertex $\boldsymbol{v}_i$ to its first neighborhood.

Assume that the edge $(i, j) \in E$ connects vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$. We define the difference between the lengths of vertices:

$$l_j = \kappa_j^i l_i. \tag{3}$$

Eq. (3) shows that the difference between adjacent lengths can be encoded in the length of one of the vertices. The key observation is that κ_j^i can be expressed by the coefficients of the angle quantities.

Theorem 2 κ_j^i can be expressed by angle quantities α_k^i, β_k^i, and γ_k^i.

Proof Assume that vertex $\boldsymbol{v}_j$ is the mth neighborhood of $\boldsymbol{v}_i$ and $\boldsymbol{v}_i$ is the nth neighborhood of $\boldsymbol{v}_j$. According to Eq. (2), we have

$$\|\boldsymbol{v}_j - \boldsymbol{v}_i\| = \|\boldsymbol{v}_m^i - \boldsymbol{v}_i\| = l_i \frac{\sin\gamma_1^i \prod_{t=1}^{m-1} \sin\beta_t^i}{\sin\gamma_m^i \prod_{t=1}^{m-1} \sin(\alpha_t^i + \beta_t^i)}$$

and

$$\|\boldsymbol{v}_i - \boldsymbol{v}_j\| = \|\boldsymbol{v}_n^j - \boldsymbol{v}_j\| = l_j \frac{\sin\gamma_1^j \prod_{t=1}^{n-1} \sin\beta_t^i}{\sin\gamma_n^j \prod_{t=1}^{n-1} \sin(\alpha_t^i + \beta_t^i)}.$$

Since $\|\boldsymbol{v}_j - \boldsymbol{v}_i\| = \|\boldsymbol{v}_i - \boldsymbol{v}_j\|$, we have

$$\begin{aligned} & l_j \frac{\sin\gamma_1^j \prod_{t=1}^{n-1} \sin\beta_t^i}{\sin\gamma_n^j \prod_{t=1}^{n-1} \sin(\alpha_t^i + \beta_t^i)} \\ =& l_i \frac{\sin\gamma_1^i \prod_{t=1}^{m-1} \sin\beta_t^i}{\sin\gamma_m^i \prod_{t=1}^{m-1} \sin(\alpha_t^i + \beta_t^i)}, \end{aligned}$$

which implies

$$l_i = \frac{\dfrac{\sin\gamma_1^j \prod_{t=1}^{n-1} \sin\beta_t^i}{\sin\gamma_n^j \prod_{t=1}^{n-1} \sin(\alpha_t^i + \beta_t^i)}}{\dfrac{\sin\gamma_1^i \prod_{t=1}^{m-1} \sin\beta_t^i}{\sin\gamma_m^i \prod_{t=1}^{m-1} \sin(\alpha_t^i + \beta_t^i)}} \cdot l_j. \tag{4}$$

Note that the main advantage of this representation is that the mesh can be represented by local quantities which are invariant under translation, rotation, and uniform scaling.

3.2 Frame representation

Since the 1-ring neighborhood of vertex $\boldsymbol{v}_i$ can be computed based on its angle quantities and local frame, in this subsection we present a method to represent the local frames which are invariant under translation, rotation, and uniform scaling.

Definition 3 Given the frame $\boldsymbol{F}_i = (\boldsymbol{b}_i, \boldsymbol{n}_i, \boldsymbol{N}_i)$ at vertex $\boldsymbol{v}_i$ and frames $\boldsymbol{F}_j^i = (\boldsymbol{b}_j^i, \boldsymbol{n}_j^i, \boldsymbol{N}_j^i)$ for its 1-ring neighborhood,

$$\begin{cases} \boldsymbol{b}_j^i = \lambda_{j11}^i \boldsymbol{b}_i + \lambda_{j12}^i \boldsymbol{n}_i + \lambda_{j13}^i \boldsymbol{N}_i, \\ \boldsymbol{n}_j^i = \lambda_{j21}^i \boldsymbol{b}_i + \lambda_{j22}^i \boldsymbol{n}_i + \lambda_{j23}^i \boldsymbol{N}_i, \\ \boldsymbol{N}_j^i = \lambda_{j31}^i \boldsymbol{b}_i + \lambda_{j32}^i \boldsymbol{n}_i + \lambda_{j33}^i \boldsymbol{N}_i, \end{cases} \tag{5}$$

then λ_{jmn}^i $(m, n = 1, 2, 3)$ are called relative frame coefficients between vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_j^i$.

Theorem 3 The frames of 1-ring neighborhood of vertex $\boldsymbol{v}_i$ can be represented by $\boldsymbol{F}_i$ and angle quantities α_k^i, β_k^i, and γ_k^i on the mesh.

Proof Suppose that the first neighborhood of vertex $\boldsymbol{v}_i$ is vertex $\boldsymbol{v}_j$ and that $\boldsymbol{v}_i$ is the kth neighborhood of $\boldsymbol{v}_j$. Assume that the lengths of the first edges of vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are l_i and l_j, respectively. From Eq. (2), we obtain

$$\boldsymbol{v}_k^j - \boldsymbol{v}_j = -(\boldsymbol{v}_1^i - \boldsymbol{v}_i) = -(l_i \sin\gamma_i \boldsymbol{b}_i + l_i \cos\gamma_i \boldsymbol{N}_i).$$

Let

$$\begin{aligned} \boldsymbol{e} =& \frac{l_i \sin\beta_i^1}{\sin(\alpha_i^1 + \beta_i^1)} \Big(2\cos\alpha_i^1 \cos\gamma_i^1 + \tan\gamma_i^2 \sin\gamma_i^1 \\ & - \frac{\cos\gamma_i^1 \sin(\alpha_i^1 + \beta_i^1)}{\sin\beta_i^1} \Big) \boldsymbol{N}_i \\ & + l_i \left(\frac{\cos\alpha_i^1 \sin\beta_i^1 \sin\gamma_i^1}{\sin(\alpha_i^1 + \beta_i^1)} - \sin\gamma_i^1 \right) \boldsymbol{b}_i \\ & + \frac{l_i \sin\alpha_i^1 \sin\beta_i^1 \sin\gamma_i^1}{\sin(\alpha_i^1 + \beta_i^1)} \boldsymbol{n}_i. \end{aligned}$$

We obtain

$$\boldsymbol{v}_{k+1}^j - \boldsymbol{v}_j = (\boldsymbol{v}_{k+1}^j - \boldsymbol{v}_i) - (\boldsymbol{v}_j - \boldsymbol{v}_i) = \boldsymbol{e}.$$

Finally, we solve the following equations to obtain $\boldsymbol{N}_j$:

$$\begin{cases} \langle \boldsymbol{N}_j, -\sin\gamma_i^1 \boldsymbol{b}_i - \cos\gamma_i^1 \boldsymbol{N}_i \rangle = \cos\gamma_j^k, \\ \langle \boldsymbol{N}_j, \boldsymbol{e} \rangle = \dfrac{l_j \sin\gamma_j^1 \prod_{t=0}^k \sin\beta_t}{\prod_{t=1}^k \sin(\alpha_t + \beta_t)} \cos\gamma_j^{k+1}, \\ \|\boldsymbol{N}_j\| = 1. \end{cases} \tag{6}$$

Other normals of the 1-ring neighborhood can be computed in the same way.

Theorem 4 Given the angle quantities, relative frame coefficients, the length of the first edge taken from an existing mesh, and an arbitrary local frame $\boldsymbol{F}_i$ at vertex $\boldsymbol{v}_i$, the mesh is uniquely determined up to translation.

Proof The existence of the mesh is obvious: we just translate and rotate the existing mesh to make the frame at vertex $\boldsymbol{v}_i$ match $\boldsymbol{F}_i$. For uniqueness, note that Eq. (6) determines the frames of 1-ring neighborhood of vertex $\boldsymbol{v}_i$. According to Eq. (2), the positions of its 1-ring neighborhood are determined. Similarly, the positions at each vertex are determined while the neighborhood is growing.

If we denote frame $\boldsymbol{F}_i$ as a vector $[\boldsymbol{b}_i, \boldsymbol{n}_i, \boldsymbol{N}_i]$ and denote all frames for the mesh as a vector $[\boldsymbol{F}_1, \boldsymbol{F}_2, \ldots, \boldsymbol{F}_n]^{\mathrm{T}}$, the local frames at each vertex of the mesh can be reconstructed by solving the system $\boldsymbol{Mx} = \boldsymbol{0}$. $\boldsymbol{M}$ is a $3m \times 3n$ matrix, where $n = |V|$ and $m = \sum_{i=1}^n d_i$ (d_i is the number of neighborhoods of vertex $\boldsymbol{v}_i$):

$$M_{ij} = \begin{cases} \lambda_{rmn}^s, & 3(s-1) < j \le 3s, \\ -1, & (i-j) \bmod 3 = 0, \\ & \lfloor j/3 \rfloor \text{ is the } r\text{th neighbor of } s, \\ 0, & \text{otherwise}, \end{cases}$$

where s must satisfy $3\sum_{t=1}^{s-1} d_t \le i < 3\sum_{t=1}^{s} d_t$, $r = i - 3\sum_{t=1}^{s-1} d_t, m = i \bmod 3, n = j \bmod 3$. Relative frame coefficient representation for the frames is invariant under linear transformation. $\boldsymbol{M}$ has rank $n-3$, which means that the nonzero solution of $\boldsymbol{F}$ can be recovered by fixing one frame and solving a linear system. If the number of neighborhoods for a vertex is small, then $\boldsymbol{M}$ is a sparse matrix, and the local frames can be efficiently computed.

Note that this representation contains redundant information, since Eq. (2) forms an overdetermined system and the relative frame coefficients can be expressed by angle quantities.

4 Quasi-angle-preserving mesh deformation

The mesh editing process consists of the following steps. First, the user defines a region of interest (ROI) for editing. Second, the user selects some triangles of the ROI as a handle, which acts as a control point to obtain the desired deformation results (Fig. 3). In this step, as presented in Lipman *et al.* (2004), the user can optionally define the stationary anchors, which support the transition between the ROI and the untouched part of the mesh. After the specification of the ROI, stationary anchors, and handle vertex, the mesh vertices can be classified into

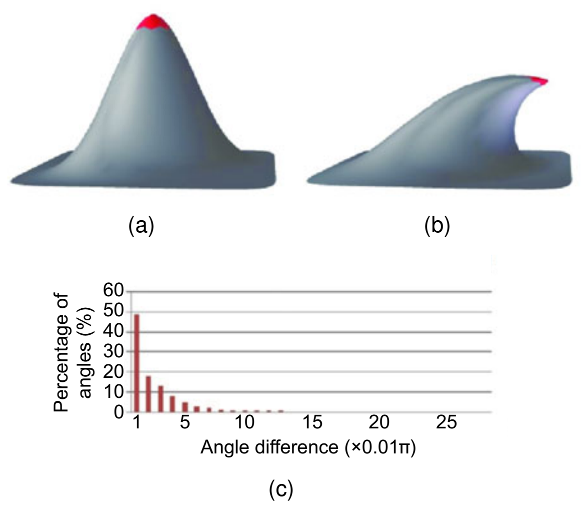


Fig. 3 Hill deformation: (a) original mesh; (b) deformed mesh; (c) angle difference. References to color refer to the online version of this figure

two groups: the modified vertices on the ROI, and the rest of the mesh, which stays fixed during the deformation. Then, in the third step, the user moves the handle to a new position. The user can rotate the triangles and translate the triangles to a new place. The scalar factor can be given as a parameter. Finally, the surface is reconstructed according to the relocation of the handle. The user can repeat the selecting and handle relocating steps for the current ROI until the desired editing result is achieved.

From this algorithm, once the user defines an ROI, we can obtain angle quantities, relative frame coefficients for each vertex in ROI, and the κ_j^i for each vertex and its neighborhoods. When transforming the handle to a new position, we constrain the new local frame and the position of the new vertex and reconstruct the deformed mesh in the following steps:

1. Compute l_i for each vertex $\boldsymbol{v}_i$ in ROI according to the transformation of the handle.

2. Construct Eq. (6) with the constrained local frame as an additional equation to obtain the local frame at each vertex in ROI.

3. Construct Eq. (2) with the constrained handle position as an additional equation to obtain the deformed geometry at each vertex in ROI.

The deformed mesh satisfying the above constraints will be reconstructed in a least-squares sense as the Laplacian mesh editing method (Lipman *et al.*, 2004). This results in three error functionals, including the differential length error functional, differential frame error functional, and differential coordinate error functional.

In the first step, according to Eq. (4), we can obtain a differential length error functional which is defined as follows:

$$E(L) = \sum_{i=1}^{n}\sum_{j=1}^{d_i}(l_i - \kappa_j^i l_j)^2 + \sum_{i=1}^{m} w_i (l_i - l_{i'})^2,$$

where $i < j$ $(i = 1, 2, \ldots, m)$, $l_{i'}$ are the constraint lengths, and $w_i > 0$ are the weights that we assign to the constraints.

In the second step, according to Eq. (6), the differential frame error functional is defined as follows:

$$\begin{aligned} E(F) = \sum_{i=1}^{n}\Bigg[&\sum_{j=1}^{d_i}(\lambda_{j11}^i \boldsymbol{b}_i + \lambda_{j12}^i \boldsymbol{n}_i + \lambda_{j13}^i \boldsymbol{N}_i - \boldsymbol{b}_j)^2 \\ &+ \sum_{j=1}^{d_i}(\lambda_{j21}^i \boldsymbol{b}_i + \lambda_{j22}^i \boldsymbol{n}_i + \lambda_{j23}^i \boldsymbol{N}_i - \boldsymbol{n}_j)^2 \\ &+ \sum_{j=1}^{d_i}(\lambda_{j31}^i \boldsymbol{b}_i + \lambda_{j32}^i \boldsymbol{n}_i + \lambda_{j33}^i \boldsymbol{N}_i - \boldsymbol{N}_j)^2\Bigg] \\ &+ \sum_{i=1}^{m} w_i \left[(\boldsymbol{b}_i - \boldsymbol{b}_i')^2 + (\boldsymbol{n}_i - \boldsymbol{n}_i')^2 + (\boldsymbol{N}_i - \boldsymbol{N}_i')^2\right], \end{aligned}$$

where $\boldsymbol{F}_i = (\boldsymbol{b}_i',\ \boldsymbol{n}_i', \boldsymbol{N}_i')$ $(i = 1, 2, \ldots, m)$ are the constraint frames and $w_i > 0$ are the weights that we assign to the constraints.

Since the frames for each vertex have been computed, according to Eq. (2), the differential coordinate error functional is defined as follows:

$$\begin{aligned} E(V) = \sum_{i=1}^{n}\Bigg\{\sum_{j=1}^{d_i}\Big[&(\boldsymbol{v}_i^j - \boldsymbol{v}_i) - (C_{i1}^j \boldsymbol{b}_i + C_{i2}^j \boldsymbol{n}_i \\ &+ C_{i3}^j \boldsymbol{N}_i)\Big]\Bigg\}^2 + \sum_{i=1}^{m} w_i (\boldsymbol{v}_i - \boldsymbol{v}_i')^2, \end{aligned}$$

where $\boldsymbol{v}_i'$ $(i = 1, 2, \ldots, m)$ are constrained vertices,

$$C_{i1}^j = \frac{l_i \sin\gamma_1^i \prod_{j=1}^{k-1}\sin\beta_j^i}{\sin\gamma_k^i \prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)} \cos\sum_{j=1}^{k-1}\alpha_j^{i'} \sin\gamma_k^i,$$

$$C_{i2}^j = \frac{l_i \sin\gamma_1^i \prod_{j=1}^{k-1}\sin\beta_j^i}{\sin\gamma_k^i \prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)} \sin\sum_{j=1}^{k-1}\alpha_j^{i'} \sin\gamma_k^i,$$

$$C_{i3}^j = -\frac{l_i \sin\gamma_1^i \prod_{j=1}^{k-1}\sin\beta_j^i}{\sin\gamma_k^i \prod_{j=1}^{k-1}\sin(\alpha_j^i + \beta_j^i)} \cos\gamma_k^i,$$

and $w_i > 0$ are the weights that we assign to the constraints.

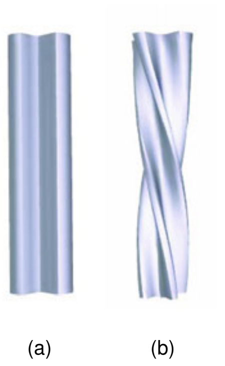
5 Implementation and analysis

The algorithm presented in this study has been implemented and tested on a computer with a 2.4 GHz Pentium IV CPU. The computational kernel of our algorithm is a sparse linear solver for the least-squares problem $\min\|\boldsymbol{Ax}-\boldsymbol{c}\|$ over the deformed mesh. This problem can be solved fast enough to guarantee interactive editing. We adopt a direct solver (Toledo, 2003) which first computes a sparse triangular factorization of the normal equations and then finds the minimizer by back-substitution. The factorization is the most time-consuming operation, but it needs to be done only once during the deformation. Solving the linear system by back-substitution is quite fast and enables us to reconstruct the surface interactively, following the user's manipulation of the handle.

In Fig. 3, after specifying the red handle regions and the whole mesh as ROI (Fig. 3a), the user translates and rotates the handle regions to a new position. Our quasi-angle-preserving mesh deformation method is used to propagate and damp the handle's transformation naturally (Fig. 3b). The angle difference between undeformed and deformed meshes is shown in Fig. 3c, where the horizontal axis describes the difference quantities of corresponding angles between undeformed and deformed meshes, and the vertical axis describes the percentage of angles with the same difference quantity. Figs. 4 and 5 demonstrate another two examples. The legs and head of the Dino model are deformed naturally.

In Figs. 6 and 7, the shape is deformed with and without the top axis as the vertex constraint, respectively. From top to bottom, the angles for rotation are π and 2π, respectively. The corresponding error charts show that more constraints will cause more distortion of the angles.

To show the performance of our method, we compare it with the method using linear rotation coordinates (Lipman *et al.*, 2005) (Fig. 8). For the Rabbit example, after specifying two ears of the original Rabbit model as the support regions and the red handle regions (Fig. 8g), the user rotates the handle regions. Propagating this rotation based on reconstruction of the local frames gives an intuitive solution of a smooth interpolation. Compared with linear rotation coordinates (right), our results (middle) are more natural.



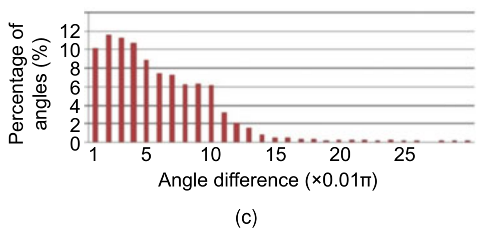
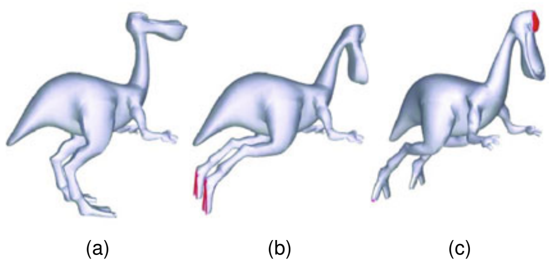


Fig. 4 Pipe deformation: (a) original mesh; (b) deformed mesh; (c) angle difference



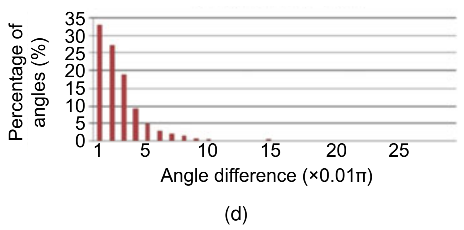
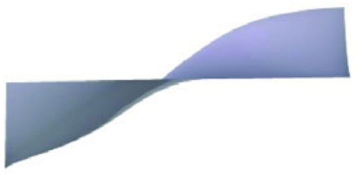


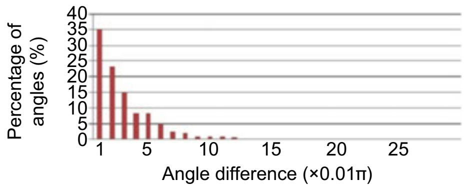
Fig. 5 Dino deformation: (a) original mesh; (b) deformed mesh; (c) deformed mesh from another viewpoint; (d) angle difference. References to color refer to the online version of this figure

We also evaluate the efficiency of the proposed technique (Table 1). The computation time depends heavily on the number of vertices of the original model.

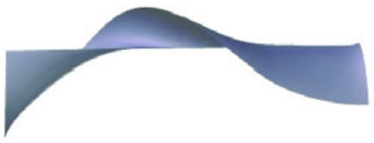
(a)



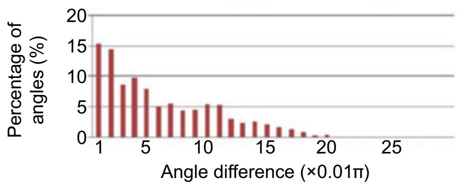
(b)



(c)



(d)

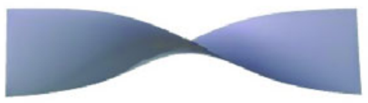


(e)

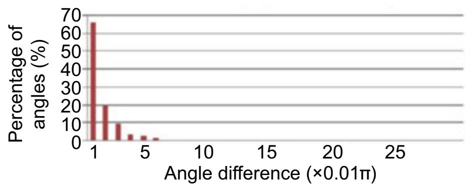
Fig. 6 Shape deformation with axial constraints: (a) original mesh; (b) deformed mesh with rotation angle π; (c) angle difference between the original mesh and deformed mesh in (b); (d) deformed mesh with rotation angle 2π; (e) angle difference between the original mesh and deformed mesh in (d)

Table 1 Model size and computation time

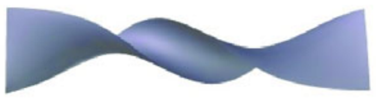
Model	Number of vertices	Computation time (s)
Hill	628	0.23
Pipe	2217	2.52
Dino	28 150	7.89
Rectangular	367	0.16
Handle of teapot	348	0.14
Mouth of teapot	359	0.18
Rabbit	36 827	9.88
Armadillo	82 672	15.26



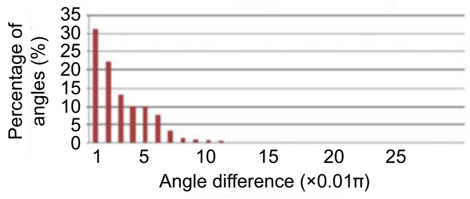
(a)



(b)



(c)



(d)

Fig. 7 Shape deformation without axial constraints: (a) deformed mesh with rotation angle π; (b) angle difference between the original mesh and deformed mesh in (a); (c) deformed mesh with rotation angle 2π; (d) angle difference between the original mesh and deformed mesh in (c)

6 Conclusions and future work

In this paper, we propose an angle-based mesh representation, which is invariant under translation, rotation, and uniform scaling. Using this mesh representation, we develop a novel quasi-angle-preserving mesh deformation method by translating, rotating, and uniform scaling of the handle. Several examples show the effectiveness of our algorithm.

There are several issues to be investigated. First, the frames and vertices are deformed separately and the translation of the handle will not affect the local frames of the mesh. How to solve this problem will be investigated in the future. Second, self-intersections may appear during the editing process. How to avoid self-intersections is another open problem. In addition, the sensitivity of the proposed method with respect to the choice of first neighborhood should be studied in the future. How to apply

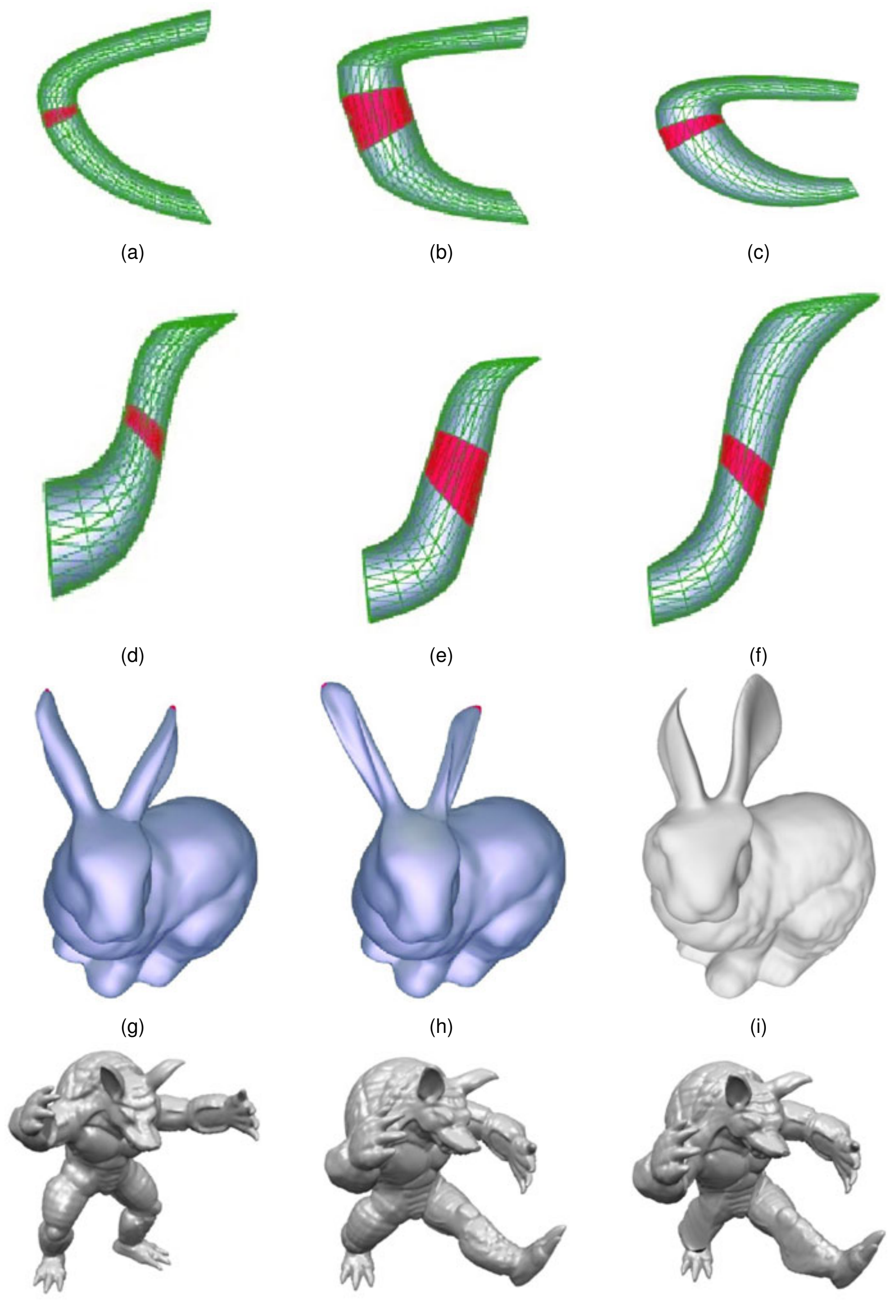


Fig. 8 Comparison between the proposed deformation method (middle) and the method using linear rotation-invariant coordinates proposed by Lipman *et al.* (2005) (right). The left column gives the original models. References to color refer to the online version of this figure

this new mesh representation to other kinds of mesh processing operations, such as mesh watermarking, is also a future direction.

References

Alexa, M., 2003. Differential coordinates for local mesh morphing and deformation. *Vis. Comput.*, **19**(2):105-114.

Au, O., Tai, C., Liu, L., *et al.*, 2006. Dual Laplacian editing for meshes. *IEEE Trans. Visual. Comput. Graph.*, **12**(3):386-395. [doi:10.1109/TVCG.2006.47]

Boier-Martin, I., Ronfard, R., Bernardini, F., 2004. Detail-preserving variational surface design with multiresolution constraints. Proc. Shape Modeling Int., p.119-128.

Botsch, M., Kobbelt, L., 2004. An intuitive framework for real-time freeform modeling. SIGGRAPH, p.630-634.

Chen, S.G., Wu, J.Y., 2004. Estimating normal vectors and curvatures by centroid weights. *Comput. Aid. Geom. Des.*, **21**(5):447-458. [doi:10.1016/j.cagd.2004.02.003]

Coquillart, S., 1990. Extended free-form deformation: a sculpting tool for 3D geometric modeling. SIGGRAPH, p.187-196.

Guskov, I., Sweldens, W., Schroder, P., 1999. Multiresolution signal processing for meshes. SIGGRAPH, p.325-334.

Hsu, W.H., Hughes, J.F., Kaufman, H., 1992. Direct manipulation of free-form deformations. SIGGRAPH, p.177-184.

Hu, J.W., Liu, L.G., Wang, G.Z., 2007. Dual Laplacian morphing for triangular meshes. *Comput. Anim. Virt. Worlds*, **18**(4-5):271-277. [doi:10.1002/cav.182]

Kobbelt, L., Campagna, S., Vorsatz, J., *et al.*, 1998. Interactive multiresolution modeling on arbitrary meshes. SIGGRAPH, p.105-114.

Lazarus, F., Coquillart, S., Jancne, P., 1994. Axial deformations: an intuitive deformation technique. *Comput.-Aid. Des.*, **26**(8):607-613. [doi:10.1016/0010-4485(94)90103-1]

Lipman, Y., Sorkine, O., Cohen-Or, D., *et al.*, 2004. Differential coordinates for interactive mesh editing. Proc. Shape Modeling Int., p.181-190.

Lipman, Y., Sorkine, O., Levin, D., *et al.*, 2005. Linear rotation-invariant coordinates for meshes. SIGGRAPH, p.479-487.

Luke, E., Collins, E., Blades, E., 2012. A fast mesh deformation method using explicit interpolation. *J. Comput. Phys.*, **231**(2):586-601. [doi:10.1016/j.jcp.2011.09.021]

MacCracken, R., Joy, K., 1996. Free-form deformations with lattices of arbitrary topology. SIGGRAPH, p.181-188.

Nealen, A., Sorkine, O., Alexa, M., *et al.*, 2005. A sketch-based interface for detail-preserving mesh editing. SIGGRAPH, p.1142-1147.

Sederberg, T.W., Parry, S.R., 1986. Free-form deformation of solid geometric models. SIGGRAPH, p.151-160.

Singh, K., Fiume, E., 1998. Wires: a geometric deformation technique. SIGGRAPH, p.405-414.

Sorkine, O., Lipman, Y., Cohen-Or, D., *et al.*, 2004. Laplacian surface editing. ACM SIGGRAPH/Eurographics on Geometry, p.179-188.

Toledo, S., 2003. TAUCS: a Library of Sparse Linear Solvers, Version 2.2.

Xu, G., Wang, G.Z., Chen, X.D., 2008. Free form deformation with rational DMS-spline volumes. *J. Comput. Sci. Technol.*, **23**(5):862-873. [doi:10.1007/s11390-008-9182-3]

Xu, G., Hui, K.C., Ge, W.B., *et al.*, 2013. Direct manipulation of free-form deformation using curve-pairs. *Comput.-Aid. Des.*, **45**(3):605-614. [doi:10.1016/j.cad.2012.09.004]

Yu, Y., Zhou, K., Xu, D., *et al.*, 2005. Mesh editing with Poisson-based gradient field manipulation. SIGGRAPH, p.496-503.

Zhou, K., Huang, J., Snyder, J., *et al.*, 2004. Large mesh deformation using the volumetric graph Laplacian. SIGGRAPH, p.641-648.

Zhou, X., Li, S., 2013. A new mesh deformation method based on disk relaxation algorithm with pre-displacement and post-smoothing. *J. Comput. Phys.*, **235**:199-215. [doi:10.1016/j.jcp.2012.10.024]

Zorin, D., Schroder, P., Sweldens, W., 1997. Interactive multiresolution mesh editing. SIGGRAPH, p.259-268. [doi:10.1145/258734.258863]