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

Scale-aware shape manipulation*

Zheng LIU†1, Wei-ming WANG2, Xiu-ping LIU2, Li-gang LIU1
(1School of Mathematical Sciences, University of Science and Technology of China, Hefei 230026, China)
(2School of Mathematical Sciences, Dalian University of Technology, Dalian 116023, China)
†E-mail: Liu.Zheng.jojo@gmail.com

Received Apr. 1, 2014; Revision accepted July 24, 2014; Crosschecked Aug. 19, 2014

Abstract: A novel representation of a triangular mesh surface using a set of scale-invariant measures is proposed. The measures consist of angles of the triangles (triangle angles) and dihedral angles along the edges (edge angles) which are scale and rigidity independent. The vertex coordinates for a mesh give its scale-invariant measures, unique up to scale, rotation, and translation. Based on the representation of mesh using scale-invariant measures, a two-step iterative deformation algorithm is proposed, which can arbitrarily edit the mesh through simple handles interaction. The algorithm can explicitly preserve the local geometric details as much as possible in different scales even under severe editing operations including rotation, scaling, and shearing. The efficiency and robustness of the proposed algorithm are demonstrated by examples.

Key words: Differential coordinates, Scale-invariant measures, Surface deformation
doi:10.1631/jzus.C1400122 **Document code:** A **CLC number:** TP391.41

1 Introduction

Shape manipulation has wide applications in computer graphics and animation. A variety of techniques have been developed to transform an original shape into a new one under a certain number of constraints. These techniques can be used to develop an efficient deformation tool to provide physically plausible and aesthetically pleasing surface deformation results, which in particular requires its local geometric details to be preserved as much as possible. Any smooth deformation can be decomposed into three modes, namely rotation, scaling, and shearing. We want to find an efficient local encoding of geometric details which can facilitate intuitive surface manipulation and deformation in these three modes.

In this paper, we introduce a novel differential representation of the 3D triangular mesh surface using a set of angle measures (triangle angles and edge angles) which are invariant to rigid and isotropic scale transformations. The representation determines a unique surface up to global similarity if the measurements are from the existing mesh. The representation is designed specifically to be shape preserving in different scales for arbitrary deformation.

As we know, the differential representations used in existing works are often absolute length measures, such as edge lengths (Sorkine and Alexa, 2007), local frames (Lipman *et al.*, 2005; Paries *et al.*, 2007; Kircher and Garland, 2008; Wang *et al.*, 2012), and differential coordinates (Sheffer and Kraevoy, 2004; Sorkine *et al.*, 2004; Yu *et al.*, 2004), which cannot efficiently handle the scaling deformations. Furthermore, some cannot preserve geometric details in shearing deformations. The reason is that the absolute length measures can hardly be preserved under such deformation operations. In contrast, our scale-invariant measures encode only the relative angles. Due to scale invariance, our method aims to find a discrete approximation for a conformal immersion of the source mesh with additional edge angle constraints. We believe that scale variation and

* Project supported by the National Natural Science Foundation of China (No. 61222206) and the One Hundred Talent Project of the Chinese Academy of Sciences, China

conformal deformation are desirable in digital geometry processing, because they do not exhibit shear and therefore preserve texture fidelity as well as the quality of the mesh itself (Paries *et al.*, 2007; Crane *et al.*, 2011).

The angle measurements (triangle angles and edge angles) are then used for a two-step iterative deformation algorithm which can efficiently preserve geometric details in arbitrary deformation operations through some point and orientation constraints. The basic idea is to minimize the deviation of each vertex 1-ring neighborhood in the ℓ_2-sense, ignoring rotation, translation, and scaling. The vertex and its 1-ring neighborhoods are encoded using triangle angles and edge angles between consecutive edges. The energy is minimized using an alternating scheme, fixing either face normals or vertex coordinates during each iteration.

Our work endeavors to preserve the local geometric details under the large-scale editing operations using our scale-invariant measures. We can easily note the characteristics of our algorithm in Figs. 9 and 11, and find that our algorithm is more effective than the other algorithms (Sorkine *et al.*, 2004; Lipman *et al.*, 2005; Paries *et al.*, 2007; Sorkine and Alexa, 2007) to preserve the local features. We can obtain an optimal rigidity approximation of the deformed shape by rescaling the scale factors to the same one, which provides an option to rescale the local shapes. We can also use the scale factors to magnify the interesting regions to different degrees without losing the detail features.

2 Related work

Deformation is an important part in shape manipulation. The deformation approaches can be classified as surface based methods, space based methods, volumetric deformation, point cloud deformation, etc. It is beyond our scope to review all existing categories of methods, and we will restrict ourselves to surface based and space based methods, which are two main types of the methods. As our contribution fits squarely in the surface based category, here we concentrate on surface based methods, although we mention some relevant space based approaches as well.

Early work in shape deformation approaches focuses on space based methods, and some surveys are well summarized in the literature (Milliron *et al.*, 2002; Gain and Bechmann, 2008). Some recent space methods were introduced by Jacobson *et al.* (2011), Levi and Levin (2014), etc. Recently, the high-resolution meshes which are produced by 3D scanning devices of real-world objects became a prominent representation for 3D models. Having abundant local geometric details on various scales is one distinct property of these meshes. The surface based methods are particularly suited for this type of representation of models and attract widespread attention of researchers. A common interactive mode is to specify a number of original and target vertices and compute the remaining vertex positions by a variational approach. Detail preservation is the central goal of these approaches.

Multiresolution methods: Multiresolution methods preserve the geometric details by decomposing the mesh into a series of hierarchical simplified levels with different detail coefficients. The details are defined by successive differences levels and encoded to local frames of the lower level (Zorin *et al.*, 1997; Kobbelt *et al.*, 1998; Botsch and Kobbelt, 2004). These methods often apply deformation on low levels of the model, and then reconstruct the high level details.

Physics-based methods: An alternative approach is to develop algorithms based on some physics-based energies, e.g., by using continuum mechanics and elasticity theory (Hu *et al.*, 2004; Bao *et al.*, 2005; Chao *et al.*, 2010). For example, Chao *et al.* (2010) defined an elastic energy based on the distance between the deformation gradient and the rotation group, including an additional thin-shell bending term if necessary, and gave a good analysis of the relationship between the as-rigid-as-possible methods and standard elastic energy minimization.

Differential representation methods: Using the differential representation based methods to preserve the details has gained significant popularity in recent years. Commonly used differential representations often extract different local geometric properties, such as Laplacian coordinates (Sorkine *et al.*, 2004), pyramid coordinates (Sheffer and Kraevoy, 2004), gradient fields (Yu *et al.*, 2004), local frames (Lipman *et al.*, 2005; Paries *et al.*, 2007; Kircher and Garland, 2008; Wang *et al.*, 2012), and dual Laplacian coordinates (Au *et al.*, 2006) (refer to Botsch and Sorkine (2008) for a survey). The deformation technique that we derive from our discrete surface representation can be seen as differential coordinates methods. These methods are characterized by applying the change in local geometric properties to reconstruct the whole mesh by inputting some handle constraints.

Comment on metric: Our approach uses iteration to solve the main optimization problem, which is similar to local frames (Paries *et al.*, 2007), pyramid coordinates (Sheffer and Kraevoy, 2004), and dual Laplacian coordinates (Au *et al.*, 2006), but with a completely different differential representation which consists of only the angular metrics.

The edge lengths and edge angles have been used as the local shape descriptor in the literature. Grinspun *et al.* (2003) used edge angles on some cloth energies in physics based simulation. Winkler *et al.* (2010) and Fröhlich and Botsch (2011) proposed methods based on interpolating the edge lengths and edge angles, in some physics-based energies coupled with a nonlinear reconstruction method. Recently, Wang *et al.* (2012) used edge lengths and edge angles to reconstruct the mesh only through a sparse linear solver; Zhang *et al.* (2012) used metric design through Ricci flow to the deformed surface; Gao *et al.* (2012) used l_p norms for shape deformation. We note that the absolute quantities such as edge lengths can hardly be preserved under the large-scale editing operations. In contrast, we encode the relative values of angular measures and strive to preserve the local details on different scales.

Comment on deformation constraints: Lipman *et al.* (2005) and Wang *et al.* (2012) proposed a linear and rotation-invariant method to handle rotational deformations, based on the fundamental theorem of surfaces in continuous setting. However, the linear method cannot avoid artifacts when rotational constraints are required, due to the fact that deformation energies should be invariant to rotation. This is a nonlinear function of shape geometry (Botsch and Sorkine, 2008). Attempts to linearize the rotational constraints show some problems. For example, the method is insensitive to translational constraints, and the local frames may degenerate and become inconsistent with the reconstructed geometry (Paries *et al.*, 2007). The nonlinear iterative method (Sheffer and Kraevoy, 2004; Au *et al.*, 2006; Paries *et al.*, 2007) can efficiently cope with the above problems. Compared to these methods our method generates less edge angle distortion.

Deformations with large shearing constraints may result in severe artifacts. Here our method can efficiently reduce angular distortion and preserve the local details as much as possible.

Conformal deformations permit efficient manipulation of a surface with scaling constraints. The goal of our method is to allow scale invariance of each vertex 1-ring in the ℓ_2-sense. In 2D space, our idea equals the as-similar-as-possible method (Igarashi *et al.*, 2005) and relates to the conformal shape editing methods (Lipman *et al.*, 2008; Weber and Gotsman, 2010; Chen *et al.*, 2013). In 3D space, the quasi-conformal shape editing methods (Paries *et al.*, 2007; Crane *et al.*, 2011; Lipman, 2012) can permit efficient manipulation of surface geometry up to scale. Our method strives to achieve the same goal, but in sharp contrast, we provide additional edge angle constraints. Crane *et al.* (2011) produced conformal deformation results by manipulating mean curvature and boundary data which can introduce less conformal distortion than our method. However, changing the mean curvature will lead to significant edge angle distortion. Paries *et al.* (2007)'s method is similar to ours. It seeks conformal deformations and encodes local frames as quaternions at vertices to preserve geometric details. It restricts local frames to be orthogonal, and not orthonormal, which can isotropically scale the shape. However, they have not considered the edge angle constraints which can preserve the mean curvature of the surface and produce larger angular distortion than our method.

Essentially, our method preserves edge angles between adjacent triangles which correspond to an implicit preservation of mean curvature (Grinspun *et al.*, 2003), and preserves corner angles which correspond to an implicit preservation of Gaussian curvature. Also, our differential representation based method is conceptually simple, which leads to an efficient algorithm.

3 Scale-invariant measures

Scale-invariant measures are designed to capture the shape of the mesh around each vertex and measure the set of angles uniquely relating a vertex to its 1-ring neighborhoods.

3.1 Notations and definitions

Denote a 2-manifold triangular mesh as $M = \{V, E, F\}$, where V, E, and F denote the sets of its vertices, edges, and triangles, respectively. Let $v \in V$ be a vertex of M with valence k and $v_1, v_2, \cdots, v_k$ be its 1-ring neighborhood vertices (Fig. 1). $C_v = \{v, v_1, v_2, \cdots, v_k\}$ is called the 1-ring cell of v. Denote the edges $e_i = vv_i$ and triangles $f_i = \triangle vv_iv_{i+1}$, $i = 1, 2, \cdots, k$ (here $v_{k+1} = v_1$). We define the triangle angles $\theta_i = \angle(e_i, e_{i+1})$, and the edge angles $\phi_i = \angle(f_{i-1}, f_i)$. We use the bold form to represent the corresponding coordinates throughout the paper if there is no ambiguity.

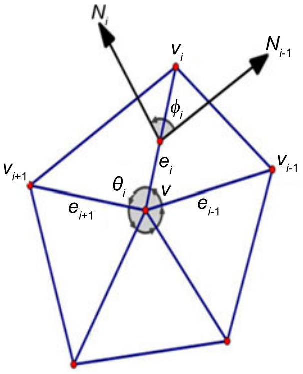

Fig. 1 Scale-invariant measures defined at the 1-ring cell of vertex v in a 3D triangular mesh

Then $\Omega_v = \{\Theta_i, \Phi_i\}$, where $\Theta_i = \{\theta_i, i = 1, 2, \cdots, k\}$ and $\Phi_i = \{\phi_i, i = 1, 2, \cdots, k\}$ are defined as scale-invariant measures of v. We can see that the measures are invariant under rigid and isotropic scaling transformation.

We define the scale-invariant measures of mesh M as $\Omega = \{\Omega_v, v \in V\}$. On the one hand, Ω is uniquely defined for a given mesh M. On the other hand, given the scale-invariant measures Ω, if we fix one of the edges and one of the orientations of the triangle containing this edge, all the edges can be obtained from Ω. Then a mesh can be uniquely determined by edge lengths and edge angles as shown in Wang *et al.* (2012). It is thus seen that a triangular mesh M has its shape (and size) completely determined by its scale-invariant measures Ω up to a rigid transformation (and a specific scale). If the user inputs more handle constraints, the system is overconstrained. We solve it in least squares sense to reconstruct a deformed mesh according to the prescribed handle constraints.

3.2 Scale-aware equations

Let us discuss how to reconstruct the mesh M from Ω. Consider the triangle f_i of the cell C_v. Denote $\boldsymbol{N}_i$ as the normal of f_i. Then edge e_{i+1} can be obtained by rotating edge e_i by angle θ_i around the axis $\boldsymbol{N}_i$. That is, we have

$$\boldsymbol{e}_{i+1} = l_i \boldsymbol{R}_i \boldsymbol{e}_i, \tag{1}$$

where $\boldsymbol{R}_i$ represents the rotation matrix of rotating around $\boldsymbol{N}_i$ by angle θ_i which encodes rotation, and l_i encodes the scaling from e_i to e_{i+1}. To preserve the connectivity of the mesh, we define the length ratios $l_i = \|\boldsymbol{e}_{i+1}\|/\|\boldsymbol{e}_i\| = \sin\alpha/\sin\beta$, where α is the opposite angle of edge e_{i+1} and β is the opposite angle of edge e_i due to the law of sines.

If $\boldsymbol{N}_i$ is known, we can have a linear equation on $\boldsymbol{v}, \boldsymbol{v}_i, \boldsymbol{v}_{i+1}$ from Eq. (1) as

$$(l_i \boldsymbol{R}_i - \boldsymbol{I})\boldsymbol{v} - l_i \boldsymbol{R}_i \boldsymbol{v}_i + \boldsymbol{v}_{i+1} = \boldsymbol{0}, \tag{2}$$

where $\boldsymbol{I}$ is the identity matrix. Applying this to all vertex angles $\theta \in \Theta$ yields a large sparse linear system

$$\boldsymbol{G}(\Theta, \mathcal{N}(\boldsymbol{V}))\boldsymbol{V} = 0, \tag{3}$$

where $\mathcal{N}$ denotes all the triangle normals of M. We call Eq. (3) the scale-aware equations of M.

Moreover, as $\angle(\boldsymbol{N}_{i-1}, \boldsymbol{N}_i) = \phi_i$, applying this to all edges yields the following constraint equations on $\mathcal{N}$:

$$H(\Phi, \mathcal{N}(\boldsymbol{V})) = \{\angle(\boldsymbol{N}_e^{\mathrm{l}}, \boldsymbol{N}_e^{\mathrm{r}}) = \phi_e, e \in E\}, \tag{4}$$

where $\boldsymbol{N}_e^{\mathrm{l}}$ and $\boldsymbol{N}_e^{\mathrm{r}}$ denote the normals of the two triangles sharing e and ϕ_e is the edge angle between two triangles. We call Eq. (4) the scale-aware constraints of M.

Given Ω, the vertices $\boldsymbol{V}$ of M can be reconstructed by solving the scale-aware equations (3) under the scale-aware constraints (4). Note that V also appears in the coefficient matrix $\boldsymbol{G}$ in Eq. (3). Thus, it is nontrivial to solve such a nonlinear system with nonlinear constraints. We will propose a two-step method to reconstruct M from Ω in the next section.

4 Scale-aware shape manipulation

Solving V from the scale-aware equations (3) is essentially a chicken-and-egg problem: on the one hand, the reconstruction of the mesh requires the triangle normals; on the other hand, the computation of triangle normals depends on the mesh vertices. To solve this problem, we present a two-step iterative solution by separating the vertices $\boldsymbol{V}$ and the triangle normals $\mathcal{N}$ to compute $\boldsymbol{V}$ and $\mathcal{N}$ alternatively.

Vertex constraints: To accurately reconstruct the mesh we need to fix some (at least three) vertices so as to eliminate the extra degrees of freedom of the intrinsic equations. In the application of shape deformation, the user fixes some vertices of the mesh and manipulates some other vertices (called handles). The deformed mesh has to be reconstructed according to the user's inputs. We denote the input constraints of the user as

$$\boldsymbol{v}_j = \boldsymbol{v}_j{}^*,\ j \in C, \tag{5}$$

where C is the set of fixed vertices and handles specified by the user.

4.1 Energy function

We solve the following minimization problem:

$$\min_{\boldsymbol{V}} E(\boldsymbol{V}) = \|\boldsymbol{G}(\Theta, \mathcal{N}(\boldsymbol{V}))\boldsymbol{V}\|^2 \quad \text{s.t. } \{\text{Eqs. (4) and (5)}\}. \tag{6}$$

Starting from an initial guess of the vertices, our algorithm alternatively updates the triangle normals $\mathcal{N}$ and the vertex positions $\boldsymbol{V}$ in two steps in an iterative manner (Fig. 2):

Step one: In this step, we are given an intermediate $\boldsymbol{V}$. The normals $\mathcal{N}$ are computed accordingly. We then update $\mathcal{N}$ to satisfy the scale-aware constraints (4).

Step two: In this step, we fix the normals $\mathcal{N}$. Then the vertices $\boldsymbol{V}$ are solved by solving the sparse linear system (3) under constraints (5).

4.1.1 Step one: updating normals

Suppose we have an initial guess which is generated by any available algorithm. Then we can compute its normal $\overline{\mathcal{N}}$ explicitly. However, $\overline{\mathcal{N}}$ may not satisfy the constraints (4). Thus, we need to update $\overline{\mathcal{N}}$ to satisfy Eq. (4).

Consider the Gaussian map $\widetilde{G}$, a graph on the unit sphere with each node representing the normal of a triangle, of M. Actually, $\widetilde{G}$ is a dual mesh on the unit sphere of M (Figs. 3a and 3b).

Now the problem is how to adjust the normals under the constraints (4). That is, given a set of

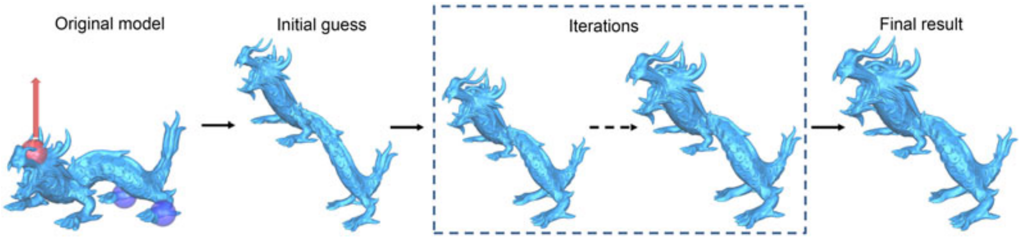

Fig. 2 Overview of our iterative algorithm. Given a Dragon model, the user fixes two points (in blue) on its legs and drags one point (in red) upwards. Starting from a severely distorted initial guess, our algorithm runs a two-step processing to update the mesh in an iterative manner. The final result explicitly preserves the local geometric details as much as possible in different scales. References to color refer to the online version of this figure

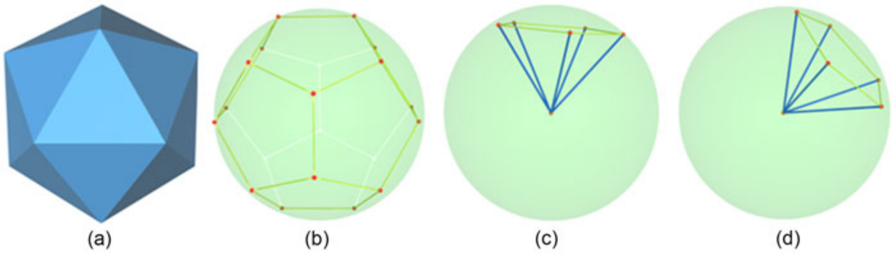

Fig. 3 Illustration of the triangular mesh (a) and its dual mesh on the unit sphere (b). (c) shows a pyramid consisting of nodes of a face in (b) and the origin of the sphere. The pyramid is deformed into another pyramid (d) with a rotation transformation which can be computed in the local phase of updating the normals in our algorithm

nodes on the unit sphere, how to adjust the positions of the nodes such that the angles (distances) between each node and its adjacent nodes are equal to the given edge angles $\varPhi$. We develop an iterative scheme to update the normals in an as-rigid-as-possible manner (Sorkine and Alexa, 2007).

Denote $N_v = \{\boldsymbol{N}_1, \boldsymbol{N}_2, \cdots, \boldsymbol{N}_k\}$ as the triangle normals of the 1-ring neighbors of vertex v. The pyramid consists of N_v and the sphere origin O (with edges $\tilde{e}_i = ON_i$, $i = 1, 2, \cdots, k$) is denoted by $\widetilde{P}_v$. Denote M' as the deformed mesh of M. The counterpart X of M in M' is denoted by X' in this section.

Local phase: For each pyramid $\widetilde{P}_v$, the optimal rotation $\widetilde{\boldsymbol{R}}_v$ from $\widetilde{P}_v$ to $\widetilde{P}_v'$ is computed by minimizing the following energy function:

$$E_{\widetilde{P_v}} = \sum_{i=1}^{k} w_i \|\tilde{\boldsymbol{e}}_i' - \widetilde{\boldsymbol{R}}_v \tilde{\boldsymbol{e}}_i\|^2, \tag{7}$$

where w_i are the cotangent weights for spatially consistent non-uniform meshes. This is solved as $\widetilde{\boldsymbol{R}}_v = \boldsymbol{V}_v \boldsymbol{U}_v^{\mathrm{T}}$ by the signed singular value decomposition (SVD) factorization of the cross-covariance matrix (Sorkine and Alexa, 2007):

$$\boldsymbol{S}_v = \sum_{i=1}^{k} w_i \tilde{\boldsymbol{e}}_i \tilde{\boldsymbol{e}}_i'^{\mathrm{T}} = \boldsymbol{U}_v \boldsymbol{D}_v \boldsymbol{V}_v^{\mathrm{T}}. \tag{8}$$

Global phase: After we apply the rotation $\widetilde{\boldsymbol{R}}_v$ to each pyramid $\widetilde{P}_v$ individually, the corresponding nodes of $\widetilde{G}$ may not coincide as shown in Fig. 4 (left). We then stitch them by minimizing the following energy function:

$$E_{\widetilde{G}} = \sum_{v \in V} E_{\widetilde{P}_v}. \tag{9}$$

Its derivative is

$$\begin{aligned}\frac{\partial E_{\widetilde{G}}}{\partial \tilde{\boldsymbol{e}}_i'} =& \|\tilde{\boldsymbol{e}}_i' - \widetilde{\boldsymbol{R}}_{v_1} \tilde{\boldsymbol{e}}_i\|^2 + \|\tilde{\boldsymbol{e}}_i' - \widetilde{\boldsymbol{R}}_{v_2} \tilde{\boldsymbol{e}}_i\|^2 \\ &+ \|\tilde{\boldsymbol{e}}_i' - \widetilde{\boldsymbol{R}}_{v_3} \tilde{\boldsymbol{e}}_i\|^2,\end{aligned} \tag{10}$$

where v_1, v_2, and v_3 are the adjacent nodes of node N_i (i.e., e_i) in $\widetilde{G}$. By setting $\frac{\partial E_{\widetilde{G}}}{\partial \tilde{\boldsymbol{e}}_i'} = 0$ we have

$$\boldsymbol{N}_i' = \frac{1}{3}(\widetilde{\boldsymbol{R}}_{v_1} \boldsymbol{N}_i + \widetilde{\boldsymbol{R}}_{v_2} \boldsymbol{N}_i + \widetilde{\boldsymbol{R}}_{v_3} \boldsymbol{N}_i). \tag{11}$$

$\boldsymbol{N}_i'$ is then normalized on the unit sphere.

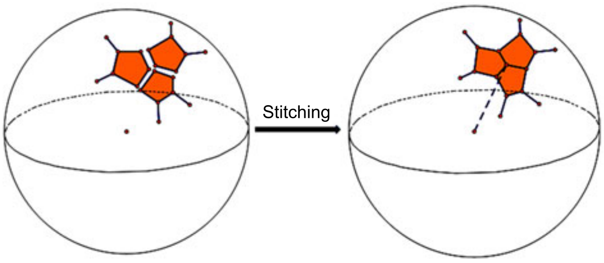

Fig. 4 Stitching the separate pyramids into a coherent mesh on the unit sphere

4.1.2 Step two: recovering vertices from normals

After $\mathcal{N}$ is updated, we fix $\mathcal{N}$ and recover $\boldsymbol{V}$ by solving the linear system $\boldsymbol{G}(\mathcal{L}, \varTheta, \mathcal{N})\boldsymbol{V} = 0$ with constraints (5). The constraints are regarded as soft constraints and $\boldsymbol{V}$ is obtained by solving the following minimization problem:

$$\min_{\boldsymbol{V}} E(\boldsymbol{V}) = \|\boldsymbol{G}\boldsymbol{V}\|^2 + \lambda \sum_{j \in C} \|\boldsymbol{v}_j - \boldsymbol{v}_j{}^*\|^2, \tag{12}$$

where λ is a weight (we set $\lambda = 100$ in our implementation).

4.2 Algorithm

Our algorithm is summarized in Algorithm 1. The deformed mesh is constructed by preserving the scale-invariant measures. That is, the result preserves the local similarity with the input mesh as much as possible. The optimal local scales are automatically obtained in different parts of the mesh.

Algorithm 1 Mesh reconstruction from the scale-aware variables

```
Require: Scale-aware variables $\varOmega$
Ensure: Mesh vertices $\boldsymbol{V}$
  Initialization: Generate an initial guess $\boldsymbol{V}$
  Repeat until convergence {
     Step one:
        Repeat until convergence {
        $\mathcal{N} \longleftarrow \boldsymbol{V}$;
        Update $\mathcal{N}$ under constraints (4)}
     Step two:
        $\boldsymbol{V} \longleftarrow$ Solve Eq. (3) under constraints (5)}
```

Advantages over previous methods: Our scale-aware approach can efficiently preserve the shape of local details and automatically determine the scale factors of the local details everywhere. When users stretch or squash the object in large scales, previous methods cannot preserve the local shape because they try to preserve the absolute measure of the length (rigidity), whereas our method tries to preserve the relative length similarity. Compared with the previous surface based conformal deformation method (Paries *et al.*, 2007), our method has a smaller angular distortion and better detail preservation results (Section 6); compared with the frame based methods (Lipman *et al.*, 2005; Paries *et al.*, 2007; Kircher and Garland, 2008; Wang *et al.*, 2012) which require manual scaling of some discrete frames, our scale factors are obtained automatically and our method has a smaller distortion.

5 Discussion

5.1 Convergence of the iterative algorithm

Fig. 5 shows the results of the iterations using our algorithm. We can see that the local details are better preserved as the algorithm is applied in more iterations. Furthermore, we quantitatively evaluate the convergence of our algorithm.

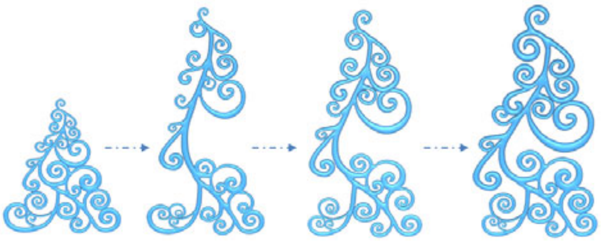

Fig. 5 The results of the iterations using our algorithm. From left to right: the input model; the intermediate result; the intermediate result; the final result. Carefully observe the difference in the branches: the latter ones better keep the similarity of the local shapes in the original model

Error metric: Our algorithm tries to preserve the scale-invariant measures Ω. To quantitatively evaluate our algorithm we respectively analyze the error metric on the angles as follows:

$$E_\theta = \sqrt{\sum_{\theta\in\Theta} (\theta_{\text{org}} - \theta_{\text{res}})^2/|\Theta|},$$

$$E_\phi = \sqrt{\sum_{\phi\in\Phi} (\phi_{\text{org}} - \phi_{\text{res}})^2/|\Phi|},$$

where θ_{org} and ϕ_{org} denote the original angles, θ_{res} and ϕ_{res} denote the result angles, and $|\cdot|$ represents the number of elements in the corresponding set.

Quantitative evaluation: We show the error curves of the example later in Fig. 11f. Fig. 6a shows the error curve of the triangle angles E_θ, which decreases quickly and finally converges. Fig. 6b shows the error curve of the edge angles E_ϕ, which always decreases in step one (normal updating step), lifts up in step two (vertex updating step), and finally converges to a stable status.

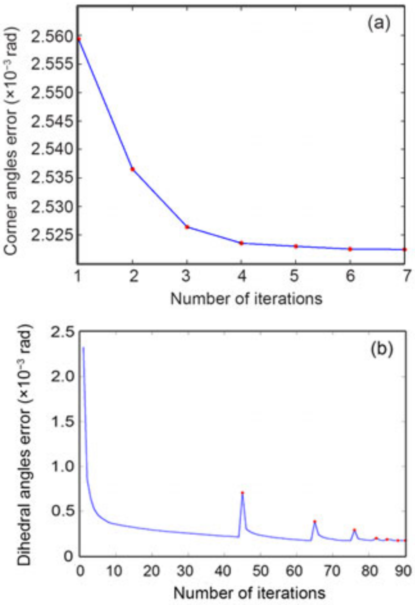

Fig. 6 The curves of angle errors of the example in Fig. 11f: (a) error curve of triangle angles (corner angles of triangles); (b) error curve of edge angles (dihedral angles of edges)

The convergence of the algorithm seriously impacts the final deformation result. In Fig. 5, we find that the final convergence result better preserves the local shapes in the branches of the tree model than the intermediate results. Although it is difficult to formally prove convergence of the above proposed algorithm, we observe in our experiments (Fig. 6) that the reconstruction converges after only a few iterations even for large handle transformations. Thus, we prove the convergence of the algorithm via numerical experiments.

5.2 Initialization

Our algorithm is insensitive to the choice of initialization. In our implementation, we all adopt the original Laplacian editing method (Sorkine *et al.*, 2004) as the initialization which should produce severe distortion in rotation, shearing, and scaling transformations. However, our algorithm can produce visually pleasing results from the highly distorted initialization even for large handle transformations.

5.3 Optimal rigidity approximation

Our scale-aware approach can well preserve the local shape by scaling the local details in an optimal manner. However, in some applications, the user wants to keep the rigidity of the local details. Starting from the result obtained by Algorithm 1, where the shapes of the triangles are well preserved, the rigidity of the triangles as well as its 1-ring neighborhoods can be easily estimated.

We extend our algorithm, called optimal rigidity approximation (ORA), to satisfy the requirements of preserving the rigidity in the results. Specifically, we scale each individual triangle to make it have the same size as the original one. Then we stitch all the scaled triangles into one coherent mesh (Igarashi *et al.*, 2005).

Note that the ORA results will increase some distortion of the edge angles and triangle angles, but produce global rigidity results.

Scale factor: We denote the area of triangle $t \in F$ as Area_t. The scale factor of t is defined as

$$\sigma_t = \frac{\text{Area}'_t}{\text{Area}_t}. \tag{13}$$

Scaling each triangle: We scale each triangle $t' = \{\boldsymbol{v}'^t_0, \boldsymbol{v}'^t_1, \boldsymbol{v}'^t_2\}$ according to its center $\boldsymbol{c}'_t = \frac{1}{3}(\boldsymbol{v}'^t_0 + \boldsymbol{v}'^t_1 + \boldsymbol{v}'^t_2)$ to a new triangle $\hat{t}_t = \{\hat{\boldsymbol{v}}^t_0, \hat{\boldsymbol{v}}^t_1, \hat{\boldsymbol{v}}^t_2\}$ as

$$\hat{\boldsymbol{v}}^t_j = \boldsymbol{c}'_t + \tau_t \boldsymbol{c}'_t \boldsymbol{v}'^t_j,\ j = 0, 1, 2, \tag{14}$$

where $\tau_t = \sqrt{1/\sigma_t} = \sqrt{\text{Area}_t/\text{Area}'_t}$ (Fig. 7).

Stitching the triangles: We stitch all the triangles into a coherent mesh by minimizing the following energy:

$$E_{\text{stitch}} = \sum_{t \in F} \sum_{j=0}^{2} \|\check{\boldsymbol{v}}^t_j \check{\boldsymbol{v}}^t_{j+1} - \hat{\boldsymbol{v}}^t_j \hat{\boldsymbol{v}}^t_{j+1}\|^2. \tag{15}$$

The above stitching scheme is actually an extension of that presented in Igarashi *et al.* (2005) from 2D mesh to 3D mesh. As Algorithm 1 obtains similar shapes of the triangles to the input mesh, the above scaling and stitching scheme reveals the rigidities of all triangles very well (Fig. 8).

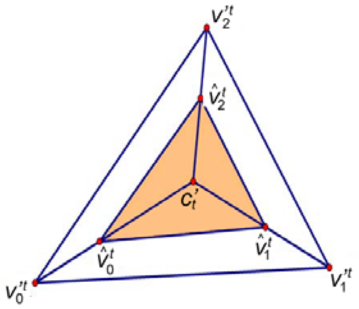

Fig. 7 Construction of an adjusted triangle from a deformed triangle by a scale factor to make the area of the adjusted triangle equal to that of the original one

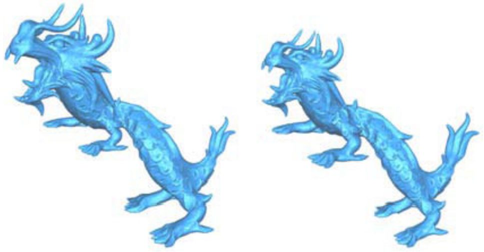

Fig. 8 The results of shape manipulation using our scale-aware manipulation method (left) and our optimal rigidity approximation method (right)

6 Experimental results

6.1 Illustration of 2D curve editing

Fig. 9 compares the editing results of a 2D Koch curve using various deformation approaches, including the original Laplacian editing approach (Sorkine *et al.*, 2004), the optimized Laplacian editing approach (Sorkine *et al.*, 2004), the linear rotation-invariant (LRI) approach (Lipman *et al.*, 2005), the as-rigid-as-possible (ARAP) approach (Sorkine and Alexa, 2007), and our approach. Large distortions in the local details occur in the results (Figs. 9b–9e). On the other hand, our method faithfully preserves the details of the original ones as much as possible with different scales, even including the sharp features (Fig. 9f), which shows the characteristic of our algorithm.

6.2 Mesh deformation

We compare our algorithm with other deformation methods in 3D deformation. To demonstrate the quality of the deformation results obtained by our approach, we apply the test examples from Botsch

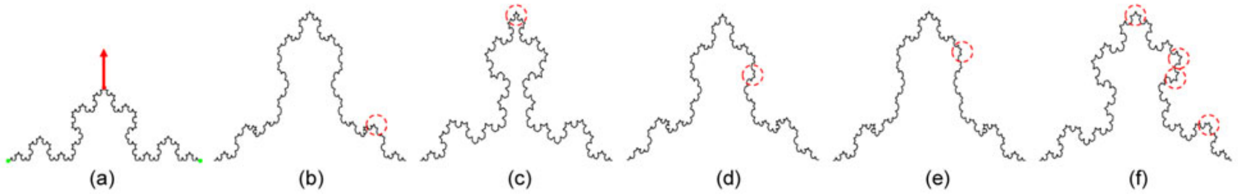

Fig. 9 Results of editing the 2D Koch curve using different approaches: (a) original 2D Koch curve (the two ends in green are fixed and the middle point in red is pulled upwards); (b) original Laplacian-based approach (Sorkine *et al.*, 2004); (c) implicit optimization Laplacian-based approach (Sorkine *et al.*, 2004); (d) linear rotation-invariant (LRI) approach (Lipman *et al.*, 2005); (e) as-rigid-as-possible (ARAP) approach (Sorkine and Alexa, 2007); (f) our scale-aware approach. References to color refer to the online version of this figure

and Sorkine (2008). The extreme examples are chosen to reveal the limitations of linear approaches; i.e., existing linear methods show gross artifacts on at least one of these examples.

From Fig. 10, we can see that our approach produces plausible results for all these examples and does not suffer from linearization artifacts. The quality of our results can compare to that of any nonlinear approach, such as Primo (Botsch *et al.*, 2006), ARAP (Sorkine and Alexa, 2007), and consistent local frames (Paries *et al.*, 2007).

Fig. 11 shows the deformation results when we stretch the Xmas tree model using different approaches. Our algorithm appropriately scales the local features to satisfy the input constraints of the user without obvious distortion. There are abundant methods that focus on rotation invariance only, but there are a couple of methods that address the similar problem to ours. To demonstrate the advantage of our representation, we compare it with the conformal deformation method (Paries *et al.*, 2007) which is quaternion based. Comparing Fig. 11e and Fig. 11f we can find that our method produces a better shape preservation and less distortion of edge angles than the conformal deformation method (Paries *et al.*, 2007).

As is evident in Figs. 10 and 11, our scale-aware method consistently gives the best value of edge angle distortion (E_ϕ), at a very small, even insignificant, penalty in triangle angle distortion (E_θ).

Our algorithm can be used to expand the focal region and reconstruct the deformation result. Fig. 12 shows the results of magnifying the local region by different scales. Fig. 13 shows the results of magnifying different local regions on the same scale.

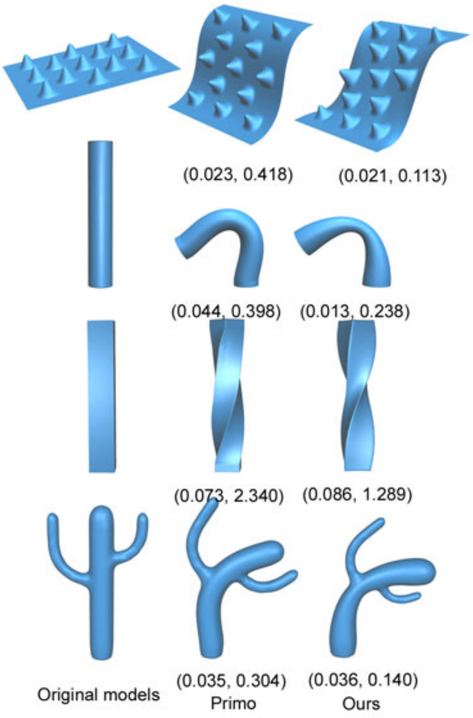

Fig. 10 Our results (right) and the results of Primo (Botsch *et al.*, 2006) (middle) for a pure translation, a 120° bending, a 135° twisting, and a 70° bending of different objects, which are some extreme examples shown in Botsch and Sorkine (2008). Numbers in brackets denote triangle angle distortion and edge angle distortion

The results show that the details of the local shapes can be well preserved in different scales.

Fig. 14 shows more results of deforming a Dinosaur model using our method.

6.3 Implementation details

We have implemented our algorithm using C++ on a notebook with an Intel dual core 2.10/2.10 GHz

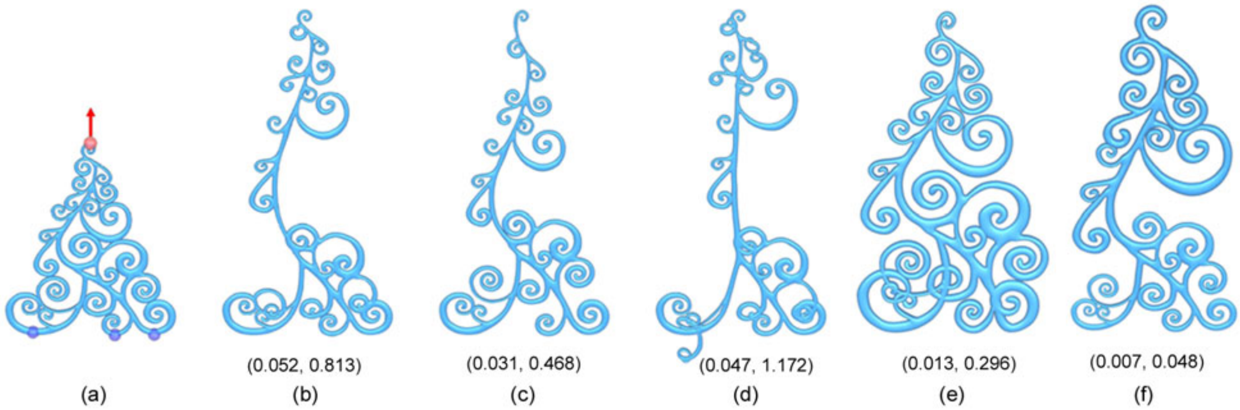

Fig. 11 Results of manipulating the Xmas tree model using different approaches: (a) Xmas tree model (the three points in blue are fixed and the point in red is pulled upwards); (b) Laplacian editing approach (Sorkine *et al.*, 2004); (c) linear rotation-invariant (LRI) approach (Lipman *et al.*, 2005); (d) as-rigid-as-possible (ARAP) approach (Sorkine and Alexa, 2007); (e) consistent local frames approach (Paries *et al.*, 2007); (f) our scale-aware approach. Numbers in brackets denote triangle angle distortion and edge angle distortion. Our algorithm is more effective for preserving the local features in different scales than the other algorithms under large-scale edit operation. References to color refer to the online version of this figure

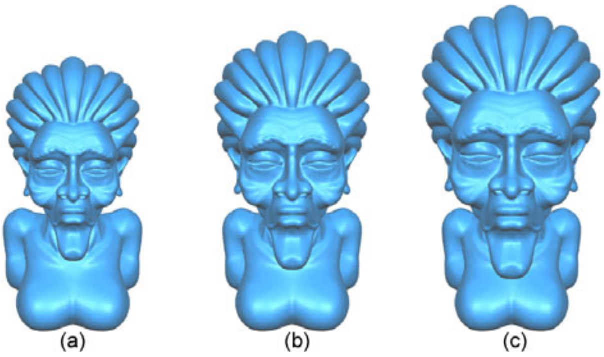

Fig. 12 Results of magnifying the same part of the Elder model with different scales: (a): original model; (b): scaling the head by a scale of 1.5; (c): scaling the head by a scale of 2

processor and 4 GB RAM. In step one, we use an iterative local/global method to update the normal. It usually takes 5–20 iterations to converge; thus, it is very fast and the runtime is comparable to that of the ARAP method (Sorkine and Alexa, 2007). The most time-consuming part of our algorithm is solving the sparse linear system (Eq. (3)) in step two. The factorization of the normal equation may take a longer time, but it is precomputed only once. At each iteration, only back-substitutions are performed to solve the system. We use the sparse Cholesky linear solver for matrix computation. In all our experiments, our algorithm usually takes 10–20 iterations to converge. Due to the two-step iterative process, our algorithm needs more running time than linear variational deformation methods such as the Laplacian editing method (Sorkine *et al.*, 2004) and linear rotation-invariant method (Lipman *et al.*, 2005). However, the angular measure errors are very low in all the cases in this study, which demonstrates the efficiency and accuracy of our algorithm. Table 1 gives the model statistic, the time required, and the error estimation of the deformation results, which are measured when iteration converges.

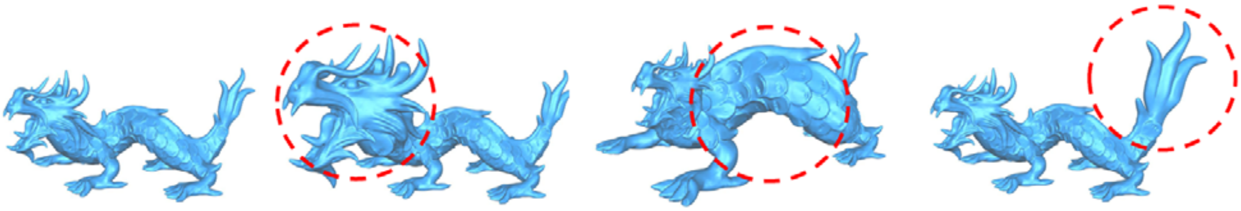

Fig. 13 Results of magnifying the different parts of the Dragon model with the same scale (×3). From left to right: original dragon model; scaling the head; scaling the body; scaling the tail

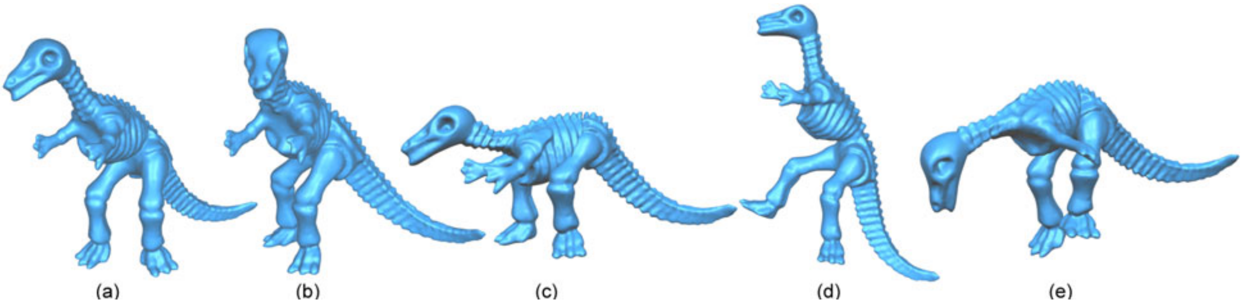

Fig. 14 Results of editing the Dino model (a) to a variety of different postures (b–e)

Table 1 Model statistics and performance

Model	Number of vertices	Runtime (s)	E_θ	E_ϕ
Bumpy plane	40 401	76	0.021	0.113
Cylinder	4802	8	0.013	0.238
Bar	2602	6	0.086	1.289
Cactus	5261	11	0.036	0.140
Xmas tree	8582	24	0.007	0.048
Dragon	19 974	42*	0.022*	0.113*
Dino	5420	13*	0.033*	0.311*
Elder	12 500	37*	0.004*	0.032*

* The average of all the examples

7 Conclusions

In this paper, we introduce a new set of scale-invariant measures for representing the 3D triangular mesh. The measures are invariant to rigid and isotropic scale transformations. Then we present a robust and efficient approach to edit the shape based on the measures. The shape of the local details is well preserved in the deformation results. The experimental results show that our algorithm is an effective tool for manipulating shapes and can generate visually pleasing results even under severe editing operations.

For our future work, we will speed up the algorithm using GPU implementation. It is also worthwhile to apply our scale-invariant measures in other applications such as surface morphing, detail enhancement, and coating.

References

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

Bao, Y.F., Guo, X.H., Qin, H., 2005. Physically based morphing of point-sampled surfaces. *Comput. Anim. Virt. Worlds*, **16**(3-4):509-518. [doi:10.1002/cav.100]

Botsch, M., Kobbelt, L., 2004. An intuitive framework for real-time freeform modeling. *ACM Trans. Graph.*, **23**(3):630-634. [doi:10.1145/1015706.1015772]

Botsch, M., Sorkine, O., 2008. On linear variational surface deformation methods. *IEEE Trans. Visual. Comput. Graph.*, **14**(1):213-230. [doi:10.1109/TVCG.2007.1054]

Botsch, M., Pauly, M., Gross, M.H., *et al.*, 2006. PriMo: coupled prisms for intuitive surface modeling. Proc. Symp. on Geometry Processing, p.11-20.

Chao, I., Pinkall, U., Sanan, P., *et al.*, 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, **29**(4):38:1-38:6. [doi:10.1145/1778765.1778775]

Chen, R.J., Weber, O., Keren, D., *et al.*, 2013. Planar shape interpolation with bounded distortion. *ACM Trans. Graph.*, **32**(4), Article 108. [doi:10.1145/2461912.2461983]

Crane, K., Pinkall, U., Schröder, P., 2011. Spin transformations of discrete surfaces. *ACM Trans. Graph.*, **30**(4), Article 104. [doi:10.1145/2010324.1964999]

Fröhlich, S., Botsch, M., 2011. Example-driven deformations based on discrete shells. *Comput. Graph. Forum*, **30**(8):2246-2257. [doi:10.1111/j.1467-8659.2011.01974.x]

Gain, J., Bechmann, D., 2008. A survey of spatial deformation from a user-centered perspective. *ACM Trans. Graph.*, **27**(4), Article 107. [doi:10.1145/1409625.1409629]

Gao, L., Zhang, G.X., Lai, Y.K., 2012. L_p shape deformation. *Sci. China Inform. Sci.*, **55**(5):983-993. [doi:10.1007/s11432-012-4574-y]

Grinspun, E., Hirani, A.N., Desbrun, M., *et al.*, 2003. Discrete shells. Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation, p.62-67.

Hu, S.M., Li, C.F., Zhang, H., 2004. Actual morphing: a physics-based approach to blending. Proc. 9th ACM Symp. on Solid Modeling and Applications, p.309-314.

Igarashi, T., Moscovich, T., Hughes, J.F., 2005. As-rigid-as-possible shape manipulation. *ACM Trans. Graph.*, **24**(3):1134-1141. [doi:10.1145/1073204.1073323]

Jacobson, A., Baran, I., Popović, J., *et al.*, 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, **30**(4), Article 78. [doi:10.1145/2010324.1964973]

Kircher, S., Garland, M., 2008. Free-form motion processing. *ACM Trans. Graph.*, **27**(2), Article 12. [doi:10.1145/1356682.1356685]

Kobbelt, L., Campagna, S., Vorsatz, J., *et al.*, 1998. Interactive multi-resolution modeling on arbitrary meshes. Proc. 25th Annual Conf. on Computer Graphics and Interactive Techniques, p.105-114. [doi:10. 1145/280814.280831]

Levi, Z., Levin, D., 2014. Shape deformation via interior RBF. *IEEE Trans. Visual. Comput. Graph.*, **20**(7):1062-1075. [doi:10.1109/TVCG.2013.255]

Lipman, Y., 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph.*, **31**(4), Article 108. [doi:10.1145/2185520.2185604]

Lipman, Y., Sorkine, O., Levin, D., *et al.*, 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph.*, **24**(3):479-487. [doi:10.1145/1073204.1073217]

Lipman, Y., Levin, D., Cohen-Or, D., 2008. Green coordinates. *ACM Trans. Graph.*, **27**(3), Article 78. [doi:10.1145/1360612.1360677]

Milliron, T., Jensen, R.J., Barzel, R., *et al.*, 2002. A framework for geometric warps and deformations. *ACM Trans. Graph.*, **21**(1):20-51. [doi:10.1145/504789.504791]

Paries, N., Degener, P., Klein, R., 2007. Simple and efficient mesh editing with consistent local frames. Proc. 15th Pacific Conf. on Computer Graphics and Applications, p.461-464. [doi:10.1109/PG.2007.43]

Sheffer, A., Kraevoy, V., 2004. Pyramid coordinates for morphing and deformation. Proc. 2nd Int. Symp. on 3D Data Processing, Visualization and Transmission, p.68-75. [doi:10.1109/3DPVT.2004.99]

Sorkine, O., Alexa, M., 2007. As-rigid-as-possible surface modeling. Symp. on Geometry Processing, p.109-116.

Sorkine, O., Cohen-Or, D., Lipman, Y., *et al.*, 2004. Laplacian surface editing. Proc. Eurographics/ACM SIGGRAPH Symp. on Geometry Processing, p.175-184. [doi:10.1145/1057432.1057456]

Wang, Y., Liu, B., Tong, Y., 2012. Linear surface reconstruction from discrete fundamental forms on triangle meshes. *Comput. Graph. Forum*, **31**(8):2277-2287. [doi:10.1111/j.1467-8659.2012.03153.x]

Weber, O., Gotsman, C., 2010. Controllable conformal maps for shape deformation and interpolation. *ACM Trans. Graph.*, **29**(4), Article 78. [doi:10.1145/1778765.1778815]

Winkler, T., Drieseberg, J., Alexa, M., *et al.*, 2010. Multiscale geometry interpolation. *Comput. Graph. Forum*, **29**(2):309-318. [doi:10.1111/j.1467-8659.2009.01600.x]

Yu, Y.Z., Zhou, K., Xu, D., *et al.*, 2004. Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graph.*, **23**(3):644-651. [doi:10.1145/1015706.1015774]

Zhang, M., Zeng, W., Xin, S.Q., *et al.*, 2012. Stable geodesic surface signatures. *Tsinghua Sci. Technol.*, **17**(4):471-480. [doi:10.1109/TST.2012.6297593]

Zorin, D., Schröder, P., Sweldens, W., 1997. Interactive multiresolution mesh editing. Proc. 24th Annual Conf. on Computer Graphics and Interactive Techniques, p.259-268. [doi:10.1145/258734.258863]