SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

May 2016, Vol. 59 052106:1–052106:9
doi: 10.1007/s11432-015-5361-3

Robust mesh deformation with salient features preservation

Yong ZHAO*, Shengjie LU, Hailong QIAN & Pengcheng YAO

School of Mathematical Sciences, Ocean University of China, Qingdao 266100, *China*

Received November 18, 2014; accepted February 4, 2015; published online February 17, 2016

Abstract Triangular meshes often contain a few salient features. Traditional deformation algorithms mainly preserve the local details and volume, thus producing unnatural results. This paper proposes a robust and effective algorithm to prevent the distortion of salient features. Firstly, the salient features can be automatically extracted through saliency-based clustering and aggregation. A nonlinear energy function is then minimized to make the salient features behave rigidly to retain the shape. Finally, for the robustness of the minimization, we generate a coarse solid subspace around the input mesh, and carry out the energy minimization in this subspace. Experiments show that our algorithm can preserve the salient features and obtain visual-pleasing results.

Keywords mesh deformation, salient features, rigid deformation, coarse solid subspace, nonlinear energy minimization

Citation Zhao Y, Lu S J, Qian H L, et al. Robust mesh deformation with salient features preservation. Sci China Inf Sci, 2016, 59(5): 052106, doi: 10.1007/s11432-015-5361-3

1 Introduction

Following the sound, image and video, the triangular mesh has become the fourth generation of digital media. In recent years, it is very difficult to robustly and effectively deform complex meshes in computer graphics. Mesh deformation has been extensively studied and applied to many fields, e.g., network game, cartoon animation and virtual simulation. Traditional algorithms mainly preserve the local details and volume, and uniformly distribute the deformation error over the whole model.

In fact, many triangular meshes contain salient features which imply global information. These features are more vulnerable than other relatively flat regions. Therefore, the deformation error should be non-uniformly distributed. As shown in Figure 1, the traditional deformation algorithm [1] cannot keep the important characteristics, and results in apparent visual artifacts. Our algorithm extracts the salient features and effectively eliminates the artifacts.

This paper aims to preserve the salient features. Guided by the heat kernel signature [2], a saliency-based clustering and aggregation method is introduced to extract the salient features. During the deformation, these features are constrained to undergo a rigid transformation to strictly preserve the shape. For the other regions, the differential coordinates deformation algorithm is adopted to preserve the local details in the least squares sense.

* Corresponding author (email: yongzhao.ouc@gmail.com)

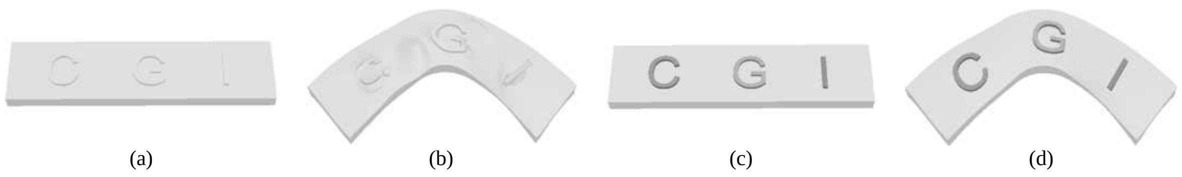

Figure 1 Comparison with the deformation algorithm [1]. (a) The CGI mesh; (b) result of algorithm [1]; (c) salient features (three letters); (d) our result.

Our deformation algorithm is a nonlinear energy minimization problem, which requires an iterative solver. To improve the convergence and stability, a coarse solid subspace is generated to envelop the input mesh, and the deformation energies are projected onto this subspace using the modified barycentric coordinates [3]. Because the subspace is very coarse, it is robust to perform the energy minimization in this subspace.

The rest of this paper is organized as follows. Section 2 briefly reviews the existing work about mesh deformation. Section 3 extracts the salient features, and derives the deformation energies. Section 4 generates the solid subspace, and develops a robust numerical solver. Section 5 discusses the experimental data and evaluates the algorithm performance. Section 6 concludes the paper and gives the future work.

2 Related work

Traditional deformation algorithms can be roughly divided into the following four categories: freeform deformation algorithms, skinning deformation algorithms, multi-resolution deformation algorithms and differential coordinates deformation algorithms.

Freeform deformation algorithms [3–11] envelop the input mesh with a coarse cage, and interpolate the deformation result after the cage is deformed. Skinning deformation algorithms [12–19] manipulate the input mesh through a skeleton. These algorithms are very intuitive and easy to use.

Multi-resolution deformation algorithms [20–25] construct a hierarchical representation for the input mesh. The deformation result is obtained by deforming the base mesh and appending the local details. Differential coordinates deformation algorithms [1, 26–34] explicitly preserve the local details, and reconstruct the deformation result by solving a nonlinear minimization problem. Zhou et al. [29] present a volumetric graph Laplacian to avoid unnatural volume changes. However, it is hard to handle dense meshes because the volumetric graph is more complex than the input mesh. Huang et al. [31] propose a subspace framework for fast computation.

The above algorithms mainly preserve the local details and volume, but neglect the global features, thus leading to unrealistic results. Kraevoy et al. [35] estimate the vulnerable regions using slippage analysis and normal curvature, and then perform a non-homogeneous resizing. Xiao et al. [36] further improve this algorithm by preserving the symmetry. Xu et al. [37] detect multiple types of joints which are implicit in the mesh model, and give a joint-aware deformation framework. Gal et al. [38] extract a descriptive set of wires and analyze their mutual relations. Deforming the mesh through these wires can intuitively maintain the original design intent and object characteristics. Zheng et al. [39] introduce simple geometric primitives to capture the editing degrees of freedom, and preserve both the characteristics of each primitive and their mutual relations. Instead of L_2 norm, Gao et al. [40] introduce L_p norm. For smaller p, the deformation error tends to concentrate on a sparse set of vertices.

3 Deformation algorithm with salient features preservation

In this paper, the triangular mesh is denoted as $M = \{P, E\}$, where P is the set of mesh vertices, and E is the connectivity information of mesh vertices. The 1-ring neighborhood of mesh vertex $\boldsymbol{p}_i$ is denoted as $N(i)$. All deformed vertex positions $\{\boldsymbol{p}'_i\}$ can be arranged as a vector $\boldsymbol{p}'$.

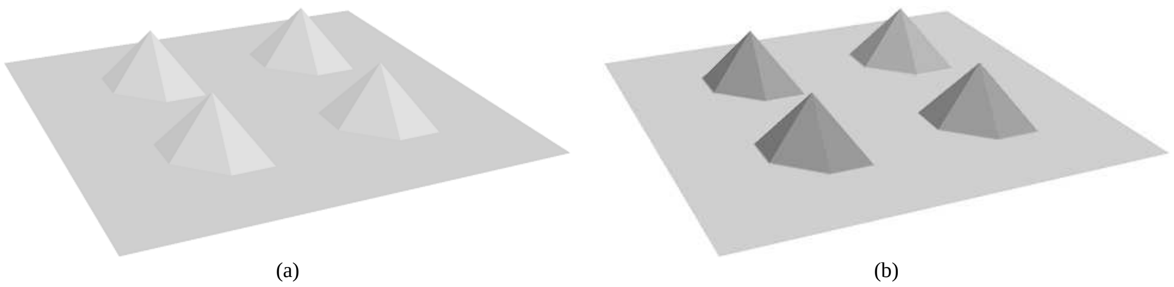

Figure 2 Demonstration of salient features extraction. (a) The mesh with four spikes; (b) the extracted salient features.

3.1 Salient features extraction

Because the triangular mesh may contain noises, the bilateral filtering [41] is first applied to remove the noises, which can improve the robustness of the feature extraction. We then automatically extract the salient features through clustering and aggregation. The extraction process consists of three steps:

Step1: The heat kernel signature (HKS) [2] is adopted to capture the local saliency of each vertex. HKS reflects the intrinsic information of the shape, and can be computed efficiently and stably.

Step2: According to the local saliency, the region-growing scheme is employed to cluster the triangular mesh in an incremental manner. Selecting the vertex with maximal saliency as the seed point, a cluster is built by successively adding the neighboring vertices until the size of the cluster or the saliency of the cluster reaches the threshold, where the saliency of the cluster is defined as the average saliency of its vertices. All the vertices of the current cluster are then excluded, and the above manner continues to build the next cluster.

Step3: The salient feature is extracted by aggregating together a collection of clusters if they are connected and have a high saliency.

In our examples, each salient feature is rendered with a random color. As shown in Figure 2, the four spikes are extracted as the salient features. Moreover, our algorithm allows interactively specifying the salient features, or grouping two features into one.

3.2 Salient features preservation

Suppose $\{F_i\}_{i=1}^n$ is the set of n extracted salient features. During the deformation, each salient feature should deform rigidly to strictly preserve its shape. Therefore, a salient feature preservation energy is introduced for F_i $(1 \leqslant i \leqslant n)$:

$$E_{F_i}(\boldsymbol{p}') = \sum_{\boldsymbol{p}_j \in F_i} \|(\boldsymbol{p}'_j - \boldsymbol{c}'_i) - \boldsymbol{R}_{F_i}(\boldsymbol{p}_j - \boldsymbol{c}_i)\|^2, \tag{1}$$

where $\boldsymbol{p}_j$ is a vertex of F_i, $\boldsymbol{p}'_j$ is its deformed position, $\boldsymbol{R}_{F_i}$ is a rotation matrix, and $\boldsymbol{c}_i$, $\boldsymbol{c}'_i$ are respectively the original and deformed centroids of F_i.

Considering all salient features, the salient features preservation energy can be formulated as a weighted sum:

$$E_{\text{sfp}}(\boldsymbol{p}') = \sum_{i=1}^{n} w_i E_{F_i}(\boldsymbol{p}'), \tag{2}$$

where w_i is the weight for F_i, and we simply take its size into account.

Now we briefly describe the computation for these rotation matrices $\{\boldsymbol{R}_{F_i}\}_{i=1}^n$, which is actually a shape matching problem. According to Müller et al. [42], we first estimate a linear transformation $\boldsymbol{A}_{F_i}$ that satisfies $\min_{\boldsymbol{A}_{F_i}} \sum_{\boldsymbol{p}_j \in F_i} \|(\boldsymbol{p}'_j - \boldsymbol{c}'_i) - \boldsymbol{A}_{F_i}(\boldsymbol{p}_j - \boldsymbol{c}_i)\|^2$, and then derive $\boldsymbol{R}_{F_i}$ from the singular value decomposition of $\boldsymbol{A}_{F_i}$. Suppose $\boldsymbol{A}_{F_i} = \boldsymbol{U}_{F_i}\boldsymbol{D}_{F_i}\boldsymbol{V}_{F_i}^{\mathrm{T}}$, thus

$$\boldsymbol{R}_{F_i} = \boldsymbol{U}_{F_i}\boldsymbol{V}_{F_i}^{\mathrm{T}},$$

where $\boldsymbol{U}_{F_i}$, $\boldsymbol{V}_{F_i}$ are orthogonal matrices, and $\boldsymbol{D}_{F_i}$ is a diagonal matrix.

3.3 Local details preservation

For the rest of the regions of M, the differential coordinates deformation algorithm is adopted to preserve the geometric details. In this paper, we use the Laplacian coordinates. The Laplacian coordinate $\boldsymbol{\delta}_i$ at vertex $\boldsymbol{p}_i$ is defined as the weighted sum of its 1-ring edges:

$$\boldsymbol{\delta}_i = L_M(\boldsymbol{p}_i) = \sum_{j\in N(i)} w_{ij}(\boldsymbol{p}_i - \boldsymbol{p}_j),$$

where $L_M(\cdot)$ is the Laplacian operator, and w_{ij} is the cotangent form weight [43] for vertex $\boldsymbol{p}_j$.

Because the Laplacian coordinate is not rotation-invariant, a rotation matrix should be computed to adjust $\boldsymbol{\delta}_i$ to correctly reflect the local detail of the deformed mesh. Therefore, the local details preservation energy can be formulated as follows:

$$E_{\mathrm{ldp}}(\boldsymbol{p}') = \sum_{\boldsymbol{p}_i\in M\backslash\{F_1,\dots,F_n\}} \|L_M(\boldsymbol{p}_i') - \boldsymbol{R}_i\boldsymbol{\delta}_i\|^2, \tag{3}$$

where $\boldsymbol{p}_i'$ and $\boldsymbol{R}_i$ are respectively the deformed position and rotation matrix of $\boldsymbol{p}_i$. Similar to Subsection 3.2, $\boldsymbol{R}_i$ is computed by shape matching within $N(i)$ [42].

3.4 Nonlinear energy minimization

To guide the deformation, we select some mesh vertices $\{\boldsymbol{p}_{h_1},\dots,\boldsymbol{p}_{h_k}\}$ as the handle whose deformed positions are directly given. The position energy can be formulated as follows:

$$E_p(\boldsymbol{p}') = \sum_{i=1}^{k} \|\boldsymbol{p}_{h_i}' - \overline{\boldsymbol{p}}_{h_i}\|^2, \tag{4}$$

where $\boldsymbol{p}_{h_i}'$, $\overline{\boldsymbol{p}}_{h_i}$ are respectively the deformed position and position constraint of $\boldsymbol{p}_{h_i}$.

Therefore, the deformation result can be achieved by solving the following energy minimization problem:

$$\min_{\boldsymbol{p}'}\{E_{\mathrm{sfp}}(\boldsymbol{p}') + \alpha E_{\mathrm{ldp}}(\boldsymbol{p}') + \beta E_p(\boldsymbol{p}')\}, \tag{5}$$

where α and β are the weights balancing the three energies.

Since the rotation computation depends on the unknown deformed mesh [42], it is a nonlinear energy minimization problem. According to previous work [30, 34], this problem can be iteratively minimized. At each iteration, a sparse linear system is solved in the least squares sense:

$$\boldsymbol{A}\boldsymbol{p}' = \boldsymbol{b}, \tag{6}$$

where matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ are derived from the quadratic energies $E_{\mathrm{sfp}}(\boldsymbol{p}')$, $E_{\mathrm{ldp}}(\boldsymbol{p}')$ and $E_p(\boldsymbol{p}')$. During the iteration, $\boldsymbol{A}$ is a constant matrix, whereas $\boldsymbol{b}$ is a variable vector depending on those rotation matrices. However, this iterative method suffers from slow convergence and numerical instability under dense meshes with poor sampling or complex shape.

4 Robust subspace solver

To address this issue, Huang et al. [31] give a subspace solver that embeds the input mesh into a coarse triangular mesh and projects the deformation energies onto the coarse mesh by the mean value coordinates [5]. The energy minimization is then performed in this subspace to enable stable computation and fast response.

Unfortunately, the mean value coordinates are globally supported and have negative values, thus leading to unrealistic deformation results. To improve this idea, we generate a coarse tetrahedral mesh as the solid subspace and adopt the modified barycentric coordinates [3] which are locally supported and unconditionally positive. Our solid subspace solver is more robust than that of Huang et al. [31].

Table 1 Algorithm performance measured in seconds. Precomputation: run time for the derivation of linear system (11) and cholesky decomposition. Deformation: run time for the rotation computation, back-substitutions and interpolation

Mesh	Number of vertices	Number of vertices in the solid subspace	Precomputation	Deformation
CGI	25796	51	0.407553	0.142902
Bunny-iH	34834	98	0.487147	0.276528
Dragon	101108	200	1.503765	0.712807
Octopus	149671	227	2.394237	1.192586
Asian Dragon	249934	184	3.662829	1.947541

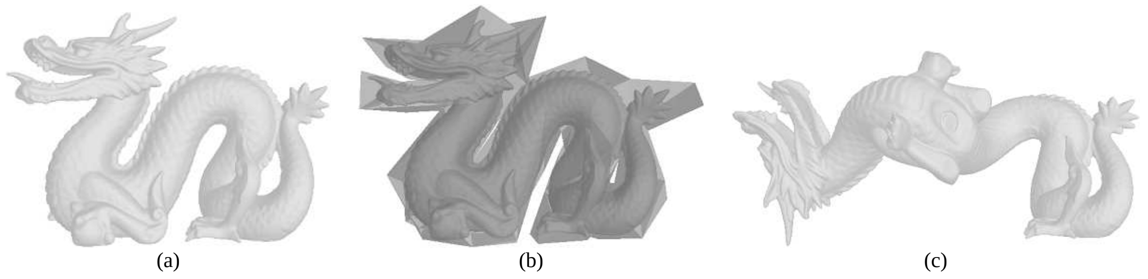

Figure 3 Deformation of Dragon. (a) The Dragon mesh; (b) the solid subspace; (c) our result.

4.1 Solid subspace generation

The input mesh is first simplified to a desired number of vertices. These simplified vertices are then offset along the outer normal direction to guarantee that the simplified mesh can completely contain the input mesh. At last, the simplified mesh is tetrahedralized to generate a tetrahedral mesh [44], i.e., the solid subspace. The resolution of the solid subspace is controlled by prescribing its local edge length. As shown in Table 1, these solid subspaces are very coarse in our examples. Figure 3 gives the solid subspace and the large deformation result of a Dragon.

4.2 Modified barycentric coordinates

The solid subspace of M is denoted as T. For a point $\boldsymbol{u}$ inside T, the modified barycentric interpolation [3] is defined as follows:

$$\boldsymbol{x}(\boldsymbol{u}) = \sum_i \phi_i(\boldsymbol{u})(\boldsymbol{x}_i + \boldsymbol{M}_i(\boldsymbol{u} - \boldsymbol{u}_i)), \tag{7}$$

where $\phi_i(\cdot)$ is the barycentric coordinate basis function, $\boldsymbol{u}_i$ is a vertex of T, and $\boldsymbol{x}_i$, $\boldsymbol{M}_i$ are respectively the deformed position and linear transformation of $\boldsymbol{u}_i$.

The linear transformations are introduced to tune the deformation gradient to be as close as possible to each other at the tetrahedron boundaries. All deformed subspace vertex positions $\{\boldsymbol{x}_i\}$ and linear transformations $\{\boldsymbol{M}_i\}$ can be respectively arranged as vector $\boldsymbol{x}$ and vector $\boldsymbol{m}$. During the deformation, they should satisfy two linear energies: discontinuity energy $E_{\text{disc}}(\boldsymbol{x}, \boldsymbol{m})$ and vibration energy $E_{\text{vibr}}(\boldsymbol{x}, \boldsymbol{m})$. Please refer [3] for more details.

Minimizing the two linear energies leads to solving a constant linear system:

$$\boldsymbol{B}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix} = \widehat{\boldsymbol{b}}, \tag{8}$$

where matrix $\boldsymbol{B}$ and vector $\widehat{\boldsymbol{b}}$ are derived from $E_{\text{disc}}(\boldsymbol{x}, \boldsymbol{m})$ and $E_{\text{vibr}}(\boldsymbol{x}, \boldsymbol{m})$.

To obtain the deformation result of M, the interpolation function (7) can be rewritten in the matrix form:

$$\boldsymbol{p}' = \boldsymbol{C}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}, \tag{9}$$

where $\boldsymbol{C}$ is a constant matrix which is related to $\phi_i(\cdot)$, M and T.

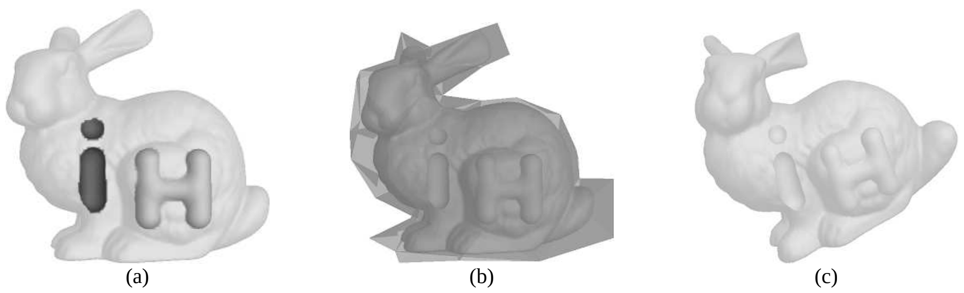

Figure 4 Deformation of Bunny-iH. (a) The Bunny-iH mesh and extracted salient features; (b) the solid subspace; (c) our result.

4.3 Robust energy minimization

With the help of formula (9), the above energies $E_{\rm sfp}(\boldsymbol{p}')$, $E_{\rm ldp}(\boldsymbol{p}')$ and $E_p(\boldsymbol{p}')$ can be projected onto the coarse solid subspace. Taking $E_{\rm disc}(\boldsymbol{x},\boldsymbol{m})$ and $E_{\rm vibr}(\boldsymbol{x},\boldsymbol{m})$ into account, the new energy minimization problem can be expressed as follows:

$$\min_{\{\boldsymbol{x},\boldsymbol{m}\}}\left\{E_{\rm sfp}\left(\boldsymbol{C}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}\right)+\alpha E_{\rm ldp}\left(\boldsymbol{C}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}\right)+\beta E_p\left(\boldsymbol{C}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}\right)+\gamma E_{\rm disc}(\boldsymbol{x},\boldsymbol{m})+\lambda E_{\rm vibr}(\boldsymbol{x},\boldsymbol{m})\right\}, \quad (10)$$

where α, β, γ and λ are the weights balancing the five energies. Their default values are respectively $\alpha = 1.0$, $\beta = 2.0$, $\gamma = 1.0$ and $\lambda = 1.0$. Furthermore, the user can manually reset these weights for different examples.

The energies $E_{\rm sfp}(\boldsymbol{p}')$, $E_{\rm ldp}(\boldsymbol{p}')$ and $E_p(\boldsymbol{p}')$ lead to the linear system (6) at each iteration. After projection, it becomes

$$\boldsymbol{AC}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}=\boldsymbol{b}.$$

Besides, the energies $E_{\rm disc}(\boldsymbol{x},\boldsymbol{m})$ and $E_{\rm vibr}(\boldsymbol{x},\boldsymbol{m})$ lead to the linear system (8). Therefore, the new linear system to be solved at each iteration becomes

$$\begin{pmatrix}\boldsymbol{AC}\\ \boldsymbol{B}\end{pmatrix}\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{m}\end{pmatrix}=\begin{pmatrix}\boldsymbol{b}\\ \hat{\boldsymbol{b}}\end{pmatrix}. \quad (11)$$

Because the solid subspace is much coarser than the input mesh, the size of linear system at each iteration is greatly reduced, thus even complex meshes can be easily dealt with. When the iteration converges in the solid subspace, the derived solution is passed to the input mesh to obtain the final deformation result through the modified barycentric interpolation (formula (9)).

5 Results and discussion

Our algorithm is implemented on a computer with Intel Core i7-3770 3.4G CPU, 3.45G RAM, and AMD Radeon HD 7450 graphics card. Figure 1 gives a comparison with the deformation algorithm [1] on a poorly-sampled mesh. Employing the uniform surface Laplacian, algorithm [1] produces obvious distortions around the features. Our algorithm achieves a visual-pleasing result due to salient features preservation.

A coarse solid subspace is generated to obtain a robust numerical solver and efficiently handle complex meshes. As shown in Figure 3, the solid subspace should compactly enclose the input mesh. In Figure 4, the two letters are extracted as salient features and deformed rigidly to avoid serious artifacts. Figure 5 extracts the suckers as salient features, and then edits the tentacles of Octopus. The shapes of these suckers are well preserved during the deformation.

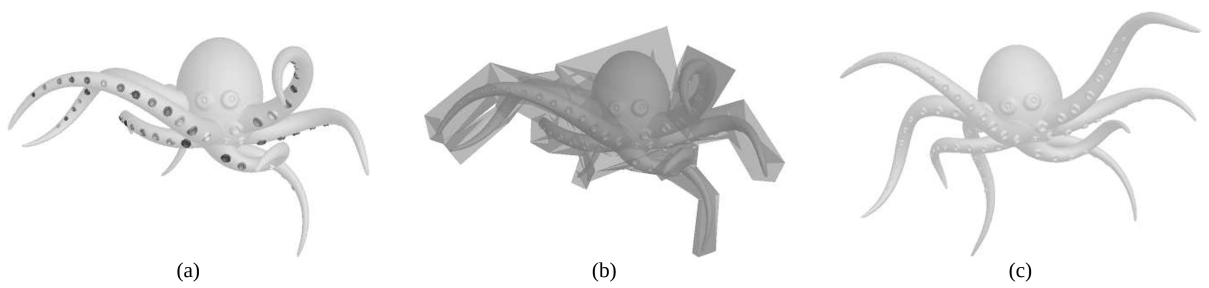

Figure 5 Deformation of Octopus. (a) The Octopus mesh and extracted salient features; (b) the solid subspace; (c) our result.

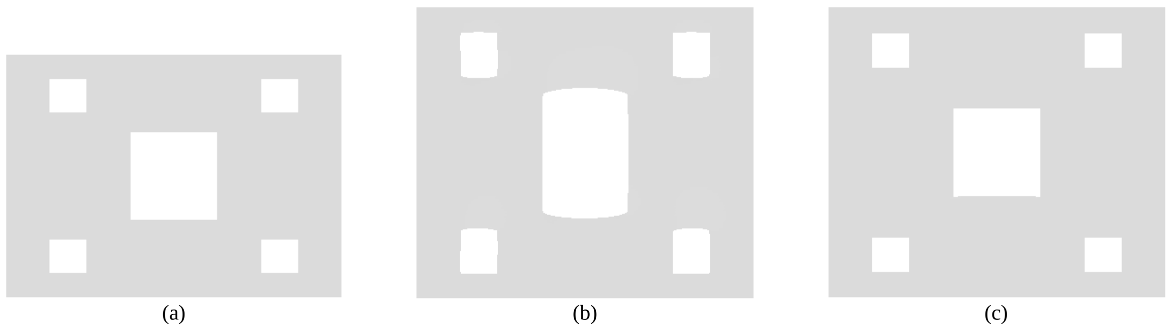

Figure 6 Comparison with the subspace deformation algorithm [31]. (a) The Rectangle-holes mesh; (b) the result of algorithm [31]; (c) our result. Note that our algorithm keeps the holes which reflect the design intent.

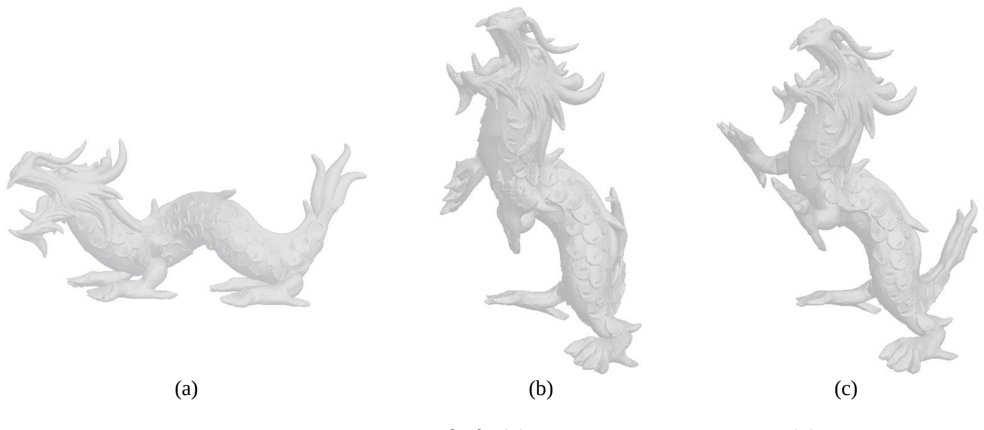

Figure 7 Comparison with the deformation algorithm [34]. (a) The Asian Dragon mesh; (b) the result of algorithm [34]; (c) our result.

Figure 6 compares our algorithm with the subspace deformation algorithm [31], which preserves the low-level properties (e.g., the local details and volume). The holes on the mesh are designed by the modeller. During the deformation, algorithm [31] stretches these holes, thus destroying the original design intent. Our algorithm extracts the holes as salient features, and effectively preserves the high-level characteristics.

Figure 7 gives a comparison with the deformation algorithm [34], which retains the rigidity of local regions. Because the mesh is densely-sampled, algorithm [34] needs to solve a sequence of large linear system and fails to converge. Apparent degenerations exhibit in the front claws and the tail of Asian Dragon. Our algorithm carries out the energy minimization in a coarse solid subspace and obtains a satisfactory result.

Table 1 lists the geometry information and the performance for some deformation examples. With the help of the coarse solid subspace, the size of linear system solved at each iteration becomes very small, thus greatly reducing the time and memory cost of our algorithm. As the linear system (11) has a constant coefficient matrix, we precompute its cholesky decomposition and perform only back-substitutions during the deformation. In these examples, three to five iterations are enough for convergence. Generally speaking, our algorithm is two times faster than the subspace algorithm [31].

6 Conclusion and future work

In this paper, a novel deformation algorithm is proposed to achieve the preservation of salient features. The salient features are automatically extracted, and constrained to undergo a rigid deformation to maintain the shape. To robustly deal with complex meshes, the energy minimization is performed in a coarse solid subspace. In the future, we will introduce a more effective feature extraction method, and consider the deformation of time-varying mesh sequences.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 61303145), Research Award Fund for Excellent Young and Middle-aged Scientists of Shandong Province (Grant No. BS2012DX043), Fundamental Research Funds for the Central Universities (Grant No. 201313005), and Open Project of the State Key Laboratory of CAD&CG, Zhejiang University (Grant No. A1308).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Lipman Y, Sorkine O, Cohen-Or D, et al. Differential coordinates for interactive mesh editing. In: Proceedings of Shape Modeling International, Genova, 2004. 181–190
2 Sun J, Ovsjanikov M, Guibas L. A concise and provably informative multi-scale signature based on heat diffusion. Comput Graph Forum, 2009, 28: 1383–1392
3 Huang J, Chen L, Liu X G, et al. Efficient mesh deformation using tetrahedron control mesh. Comput Aided Geom Des, 2009, 26:617–626
4 Singh K, Fiume E. Wires: a geometric deformation technique. In: Proceedings of the International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH), Orlando, 1998. 405–414
5 Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. ACM Trans Graph, 2005, 24: 561–566
6 Joshi P, Meyer M, DeRose T, et al. Harmonic coordinates for character articulation. ACM Trans Graph, 2007, 26: 71
7 Lipman Y, Levin D, Cohen-Or D. Green coordinates. ACM Trans Graph, 2008, 27: 78
8 Ben-Chen M, Weber O, Gotsman C. Variational harmonic maps for space deformation. ACM Trans Graph, 2009, 28: 34
9 Jacobson A, Baran I, Popović J, et al. Bounded biharmonic weights for real-time deformation. ACM Trans Graph, 2011, 30: 78
10 García F G, Paradinas T, Coll N, et al. *Cages:: a multi-level, multi-cage based system for mesh deformation. ACM Trans Graph, 2013, 32: 24
11 Li X Y, Ju T, Hu S M. Cubic mean value coordinates. ACM Trans Graph, 2013, 32: 126
12 Lewis J P, Cordner M, Fong N. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: Proceedings of the SIGGRAPH Conference, New Orleans, 2000. 165–172
13 James D L, Twigg C D. Skinning mesh animations. ACM Trans Graph, 2005, 24: 399–407
14 Yoshizawa S, Belyaev A, Seidel H-P. Skeleton-based variational mesh deformations. Comput Graph Forum, 2007, 26: 255–264
15 Yan H B, Hu S M, Martin R, et al. Skeleton-based shape deformation using simplex transformations. IEEE Trans Vis Comput Graph, 2008, 14: 693–706
16 Shi X H, Zhou K, Tong Y Y, et al. Example-based dynamic skinning in real time. ACM Trans Graph, 2008, 27: 29
17 Jacobson A, Baran I, Kavan L, et al. Fast automatic skinning transformations. ACM Trans Graph, 2012, 31: 77
18 Kavan L, Sorkine O. Elasticity-inspired deformers for character articulation. ACM Trans Graph, 2012, 31: 196
19 Vaillant R, Barthe L, Guennebaud G, et al. Implicit skinning: real-time skin deformation with contact modeling. ACM Trans Graph, 2013, 32: 125
20 Zorin D, Schröder P, Sweldens W. Interactive multiresolution mesh editing. In: Proceedings of the International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH), Los Angeles, 1997. 259–268
21 Kobbelt L, Campagna S, Vorsatz J, et al. Interactive multi-resolution modeling on arbitrary meshes. In: Proceedings of the International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH), Orlando, 1998. 105–114
22 Guskov I, Sweldens W, Schröder P. Multiresolution signal processing for meshes. In: Proceedings of the SIGGRAPH Conference, Los Angeles, 1999. 325–334

23 Botsch M, Kobbelt L. Multiresolution surface representation based on displacement volumes. Comput Graph Forum, 2003, 22: 483–492
24 Sauvage B, Hahmann S, Bonneau G-P. Volume preservation of multiresolution meshes. Comput Graph Forum, 2007, 26: 275–283
25 Manson J, Schaefer S. Hierarchical deformation of locally rigid meshes. Comput Graph Forum, 2011, 30: 2387–2396
26 Sorkine O, Cohen-Or D, Lipman Y, et al. Laplacian surface editing. In: Proceedings of the Eurographics Symposium on Geometry Processing, Nice, 2004. 179–188
27 Yu Y Z, Zhou K, Xu D, et al. Mesh editing with Poisson-based gradient field manipulation. ACM Trans Graph, 2004, 23: 644–651
28 Lipman Y, Sorkine O, Levin D, et al. Linear rotation-invariant coordinates for meshes. ACM Trans Graph, 2005, 24: 479–487
29 Zhou K, Huang J, Snyder J, et al. Large mesh deformation using the volumetric graph Laplacian. ACM Trans Graph, 2005, 24: 496–503
30 Au O K-C, Tai C-L, Liu L G, et al. Dual Laplacian editing for meshes. IEEE Trans Vis Comput Graph, 2006, 12: 386–395
31 Huang J, Shi X H, Liu X G, et al. Subspace gradient domain mesh deformation. ACM Trans Graph, 2006, 25: 1126–1134
32 Rivers A R, James D L. FastLSM: fast lattice shape matching for robust real-time deformation. ACM Trans Graph, 2007, 26: 82
33 Au O K-C, Fu H B, Tai C-L, et al. Handle-aware isolines for scalable shape editing. ACM Trans Graph, 2007, 26: 83
34 Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In: Proceedings of the Eurographics Symposium on Geometry Processing, Nice, 2007. 109–116
35 Kraevoy V, Sheffer A, Shamir A. Non-homogeneous resizing of complex models. ACM Trans Graph, 2008, 27: 111
36 Xiao C X, Jin L Q, Nie Y W, et al. Content-aware model resizing with symmetry-preservation. Vis Comput, 2015, 31: 155–167
37 Xu W W, Wang J, Yin K K, et al. Joint-aware manipulation of deformable models. ACM Trans Graph, 2009, 28: 35
38 Gal R, Sorkine O, Mitra N J, et al. iWIRES: an analyze-and-edit approach to shape manipulation. ACM Trans Graph, 2009, 28: 33
39 Zheng Y, Fu H, Cohen-Or D, et al. Component-wise controllers for structure-preserving shape manipulation. Comput Graph Forum, 2011, 30: 563–572
40 Gao L, Zhang G X, Lai Y K. Lp shape deformation. Sci China Inf Sci, 2012, 55: 983–993
41 Fleishman S, Drori I , Cohen-Or D. Bilateral mesh denoising. ACM Trans Graph, 2003, 22: 950–953
42 Müller M, Heidelberger B, Teschner M, et al. Meshless deformations based on shape matching. ACM Trans Graph, 2005, 24: 471–478
43 Desbrun M, Meyer M, Schröder P, et al. Implicit fairing of irregular meshes using diffusion and curvature flow. In: Proceedings of the SIGGRAPH Conference, Los Angeles, 1999. 317–324
44 Alliez P, Cohen-Steiner D, Yvinec M, et al. Variational tetrahedral meshing. ACM Trans Graph, 2005, 24: 617–625