Journal of Computers Vol. 29 No. 4, 2018, pp. 161-175
doi:10.3966/199115992018082904013

# Bi-harmonic Surface Based As-Rigid-As-Possible Mesh Deformation

Qiyun Sun[1,2*], Wanggen Wan[1,2], Xiang Feng[1,2], Guoliang Chen[1,2],
Muhammad Rizwan[1,2], J. Alfredo Sánchez[3]

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China
18817393221@163.com, wanwg@staff.shu.edu.cn, fengxiang0727@shu.edu.cn, chencglt@gmail.com, rizsoft@gmail.com

[2] Institute of Smart City, Shanghai University, Shanghai 200444, China

[3] Department of Computing, Electronics and Mechatronics, Universidad de las Américas Puebla (UDLAP), Mexico
j.alfredo.sanchez@gmail.com

Received 25 March 2017; Revised 2 July 2017; Accepted 9 August 2017

**Abstract**. Mesh deformation, as an interactive shape editing and deformation technology, has important value in geometric modeling and computer animation. In recent years, detail-preserving mesh deformation techniques have been receiving much attention, such as as-rigid-as-possible (ARAP) deformation. The ARAP method shows good performance in detail preservation by requiring rigidity of the local transformations of the mesh, but the alternative iterations in the optimization process needed to keep rigidity is often computationally expensive, resulting in the failure of real time mesh editing and animation when it comes to high-resolution meshes. In this paper, we propose a novel way to acquire the coarse version of the detailed high-resolution mesh by constructing bi-harmonic surface on it. We implement a multi-resolution deformation framework based on the bi-harmonic surface construction and ARAP deformation. In this way, we allow the optimization process in ARAP method to run on a coarse version of the detailed mesh in order to speed up convergence. In the same time, the details of original mesh are preserved well and can be easily restored. The experiment results show that the proposed method is feasible and effective, and most importantly, the real-time performance has been greatly improved.

**Keywords**: ARAP, Bi-harmonic surface, mesh deformation, shape deformation, surface editing

## 1 Introduction

As an interactive geometry editing technique, mesh deformation, also known as surface deformation, plays an important role in geometric modeling, computer animation, and artistic design. Taking 3D animation as an example, in order to animate a 3D mesh, modelers typically use a mesh deformation tool to create new poses and a shape interpolation tool to generate the frames between these different poses. The linear blending skin (LBS) [1-3] method has been the most popular and practical animation technique. It needs to bind a specific skeleton according to the mesh shape, which limits the deformation to articulated motion, and results in a low quality deformation in terms of local shape preservation near joints. In order to achieve intuitive free-form deformation, many alternatives to LBS have been proposed. The most intuitive deformation method allows the user to manipulate handle points, and the rest of the mesh points are automatically deformed in a natural way. However, the LBS algorithm and its alternatives used in skeleton based deformation can not satisfy the editing requirements of high-resolution mesh models. So that surface-based mesh deformation methods have become more attractive

* Corresponding Author

in recent years. These methods include differential coordinates based surface deformation methods, multi-resolution deformation methods, local transformation based surface deformation method and so on. In this paper, we focus on a kind of method called As-Rigid-As-Possible (ARAP) deformation.

The ARAP method [4-7] is a mesh deformation technique based on minimizing the so-called As-Rigid-As-Possible (ARAP) energy under particular user constraints, asking for the rigidity of the local transformations to preserve the shape of the mesh surface locally. In other words, small parts of the mesh surface should change as rigidly as possible. However, this method usually have large calculating quantity, and it is hard to produce real-time animation by implementing ARAP methods on some large and detailed meshes. Our method proposed in this paper can improve the real-time performance of ARAP method, making it possible to apply the ARAP deformation method to high-resolution mesh models to acquire real-time animations. At the same time, our method keeps the deformation quality during the animation when we improve the deformation rate.

In this paper, we propose a deformation framework to improve the performance of the ARAP deformation method and enables real-time animation production. We implement a deformation framework (Section 3.4) consisting of three main operators: the decomposition operator, realized by constructing bi-harmonic surface (Section 3.3), which separates the low and high frequencies; the deformation operator, mainly referring to ARAP deformation method (Section 3.2), used to deform the base surface (low frequencies part); and the reconstruction operator, which adds the details back onto the deformed base surface. In this way, we let the iterative optimization of ARAP method to run on a coarse version (Bi-harmonic surface) of the original mesh, accelerating the convergence of the iterative process. In the experiment, we use different mesh models to test our method and the results show that the calculation speed has been greatly improved compared to the original ARAP method. Our method, termed bi-harmonic surface based ARAP (BS-ARAP) mesh deformation, enables real-time animation even with large meshes. However, in the comparison between the ARAP method and BS-ARAP method, we have not strictly taken animation quality into consideration. Actually, there is not a standard evaluation criteria for the deformation quality during an animation.

The rest of the paper is organized as follows. In Section 2, different kinds of surface based deformation methods are introduced, some of which are referenced in our method. In Section 3, we firstly introduce the mathematical foundation of surface based methods. Then ARAP deformation method and bi-harmonic surface based deformation method are introduced in Section 3.2 and Section 3.3, respectively. Our deformation framework is mainly based on these two methods. Finally, our deformation framework is introduced in Section 3.4 and we give the details about the reason why our deformation method can improve the real-time performance of ARAP method in Section 3.5. In Section 4, we are going to show experimental results. First of all, in Section 4.1, we show that original ARAP deformation method can be applied to low-resolution models to acquire real-time animation, but can not acquire real-time animation when applied to high-resolution mesh models. Then, we show the final results of our BS-ARAP method in Section 4.2 and make comparison with other methods in Section 4.3. In Section 5, we will summarize our results and make some conclusions.

## 2 Related Work

Since our application is a pure 3D mesh-based animation, we will review some related research methods in shape deformation, especially in mesh-based shape deformation. Modern mesh based shape deformation methods satisfy user deformation constraints at handles (selected vertices or regions on the mesh) and propagate these handle deformations to the rest of shape smoothly and without removing or distorting mesh details.

### 2.1 Shell Energy Based Mesh Deformation

Shell energy based deformation theory is the mathematical foundation for surface deformation (see Section 3.1 for details), which was already proposed in 1987 by Terzopoulos D [8]. The shell energy is used to measure the stretching and bending degree of the deformation, and is minimized to get the optimal deformation. To solve the energy minimization problem, the shell energy is simplified based on the displacement function [9-10]. Handle based interactive mesh deformation method proposed by Botsch et al. [11] divided the surface into handle region, fixed region and transition region, thus the

energy minimization problem is transformed into a problem of solving a sparse linear system by applying the discrete Laplace-Beltrami operator.

### 2.2 Multi-resolution Deformation

Shell energy based deformation method can not handle rotation problems with geometric details of the mesh surface. The main idea of multi-resolution deformations is to consider the surface $S$ as a "geometric signal". By decomposing the surface $S$ into low frequency parts, a smooth base surface $B$, and high frequency parts, the geometric detail $D$, the smooth base surface $B$ is deformed first, and then the geometric detail D of the high frequency is added back onto the deformed base surface $B'$. Thus, we can acquire an intuitively deformed surface with geometric details better reserved. There are different methods to represent geometric details: displacement vector methods [12], normal vector displacement methods [13, 15], volume displacement methods [16] and deformation transmission methods [17-18].

In the displacement vector method [12], the geometric details are represented as a displacement function $h: B \to R^3$ of the base surface, and the displacement vector $h$ rotates with the rotation of the deformed base surface $B'$ so that the surface details can be preserved well. In order to reduce the unsteady deformation caused by the long displacement vector, methods in reference [13-14] resample the original mesh and take the displacement between the sampling points and the vertices of the base mesh as geometric details. Moreover, to avoid self-intersection problems, the geometric details were represented by triangular prisms formed by triangles of mesh surface $S$ and corresponding triangles on the base surface $B$ [16]. And methods in [17-18] got a similar result with [16] by transferring the deformation of the base surface $B$ to the detailed mesh $S$.

### 2.3 Differential Coordinates-based Surface Deformation

Geometric details are expressed by local differential coordinates in differential coordinates based surface deformation technique. Under the condition of satisfying the user constraints, the method tries to keep the geometric details of surface by maintain the differential coordinates as much as possible during the deformation. However, the differential coordinates defined in the global coordinate system are not rotation invariant, only to maintain the differential coordinates of the mesh will produce a distorted mesh. In order to better approximate the differential coordinates of the deformed mesh, researchers have carried out a lot of fruitful research work.

The gradient-based mesh deformation method is derived from gradient-based image process techniques. Grads-based mesh deformation [19] reduces the problem of surface deformation to the Poisson problem satisfying the Dirichlet boundary condition. Since the direction of the gradient depends on the global coordinate system and the gradient is not a rigid invariant, solving the above gradient-based deformation directly based on position constraints can not produce satisfactory results. In reference [20], linear interpolation was applied to the transformation of deformation region based on the handle's transformation. And then, the gradient was updated based on the interpolation to solve the above Poisson equation.

Laplacian-based approaches represent the geometric detail of the surface by the so-called Laplacian coordinates [21-22], which have the ability to describe the mean curvature and normal vector of the vertex. These coordinates are obtained by applying the Laplacian operator to the mesh vertices. The uniform Laplacian coordinate of a mesh vertex is expressed as the difference between the vertex coordinate and the mean coordinate of its adjacent point. Similar with the gradient-based deformation, Laplacian coordinates are not rotation invariant, so much of the research work about Laplacian deformation is focused on how to better approximate the unknown Laplacian coordinates, in particular, to deal with the rotation of the Laplacian coordinate [23-24].

### 2.4 Local Transformation Based Surface Deformation

Reuse existing mesh deformation data, can effectively improve the production efficiency of mesh deformation and reduce animation production costs. Deformation transmission reuses existing mesh deformations by transferring the deformation on the source mesh model to the target mesh model. In reference [25], "expression cloning" based on vertex displacement vectors is proposed to transfer facial

expression changes of one face model to another face model, and Sumner et al. [18] extended it to the general surface mesh deformation transfer. Furthermore, Sumner et al. [26] proposed an example-based inverse mesh kinematics method and applied it to handle-based mesh deformations based on the idea of local deformation transformation to represent deformation details. And in reference [27], a simplified deformed mesh based on bone agent is used to improve the efficiency of the algorithm, so that the deformation operation of the algorithm can be carried out interactively.

## 3 BS-ARAP Surface Deformation

### 3.1 Mathematical Foundations of Surface Deformation

Surface deformation aims to acquire a new intuitively reasonable surface under some user's editing constraints, from the mathematical point of view, it is to define a displacement function between the surfaces before and after the deformation. And the most typical and direct constraint of mesh deformation is the position constraint, that is, the user selects some specific points on the surface as the deformation handle. By specifying the new position of those handle points, the surface $S$ is to be deformed to $S'$ (Let us denote by $S \subset R^3$ a 3D surface, parameterized by a position function $p:\Omega \subset R^2 \to S \subset R^3$). In other words, the surface $S$ is to be deformed to $S'$ by adding to each point $p(u,v) \subset S$ a displacement vector $d(u,v)$, such that $S' = p'(\Omega) = \{p + d(p) \mid p \in S\}$. Handle points on the new surface have to satisfy these position constraints: $d(p_i) = d_i, \forall p_i \in H$, the $H$ means handle regions.

Intuitively, the surface can be regarded as an elastic skin, so we analyse the classical elastic energy that measures the difference between two surfaces, also known as shell energy [8]:

$$E_{shell}(S,S') = \int_{\Omega} k_s \left\| I' - I \right\|^2 + k_b \left\| II' - II \right\|^2 dudv , \quad \textbf{(1)}$$

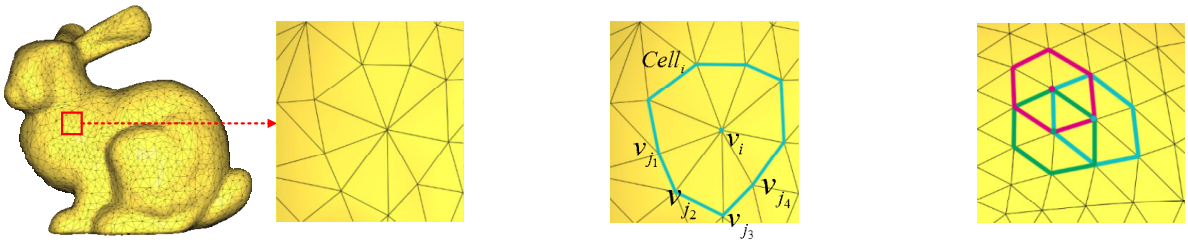
where $I$, $II$ is respectively the first and second fundamental form of the surface $S$ and similarly $I'$, $II'$ are the fundamental forms of the surface $S'$. It is known from differential geometry [28] that the former part $k_s \left\| I' - I \right\|^2$ and latter part $k_b \left\| II' - II \right\|^2$ are used to measure the stretching and bending degree of the deformation respectively. The stiffness parameters $k_s$ and $k_b$ are used to control the resistance to stretching and bending. To acquire the optimal mesh deformation, we should minimize the shell energy (1) under user's position constraints. We can see that the energy is minimized when $S'$ is a rigid transformation of the $S$. However, local rigidity of the deformed surface $S'$ cannot hold completely in any non-trivial surface deformation scenario. Therefore, the shell energy (1) can not reach zero, but rather has a minimum value and this minimum is attained when the local transformations between $S$ and $S'$ are as-rigid-as-possible which means the shape of surface is locally preserved.

### 3.2 As-Rigid-As-Possible Deformation

In the context of above energy-minimization approaches, the problem stems from comparing mesh positions in the same coordinate frame of the original undeformed shape, without considering the local rotations. However, these quadratic energies (formula (1)) are often not rotation invariant. Non-linear deformation techniques present a solution to these problems. They work by comparing the deformation of a mesh vertex to its rest position rotated to a new coordinate frame which best matches the deformation. These techniques [4, 6-7] are often labeled "as-rigid-as-possible".

We start our discussion of ARAP methods with surface discretization. In the latter experiments, all models used are represented by the triangle mesh (Fig. 1(a)).

Given a surface mesh $M$ with $n$ vertices: $\{v_i\}$, $i \in \{1,2,\ldots,n\}$, we denote by $Cell_i$ the space polygon formed by one-ring neighbors [29] of vertex $i$, made up of several adjacent triangles (Fig. 1(b)). And we analyze the rigidity of the deformation by taking cell as unit. Let us consider the following energy formulation [5] which is used to evaluate the rigidity of cell deformation:




(a) Model representation by triangle mesh

(b) One-ring neighbors of vertex $i$ are the set of vertices connected to vertex $i$, and they form a space polygon called cell

(c) Overlaps of cells

**Fig. 1.** The cells of three vertices are drawn with different colors, the corresponding vertex is colored accordingly

$$E(Cell_i, Cell_i') = \sum_{(v_m, v_n) \in e(v_i)} w_{mn} \left\| (v_m' - v_n') - R_i (v_m - v_n) \right\|^2, \tag{2}$$

where:

- $e(v_i)$ consists of the set of edges incident to $v_i$ (the spokes, i.e. red lines in Fig. 2) and the set of edges in the link (the rims, i.e. dark lines in Fig. 2) of $v_i$ in the mesh surface $M$;
- $R_i$ is a rotation matrix, $R_i \in \{R_1, \ldots R_n\}$, each vertex or cell has a corresponding $R_i$;
- $w_{ij}$ denotes a weight, we usually use the cotangent weight formula for $w_{ij}$ [30-31];

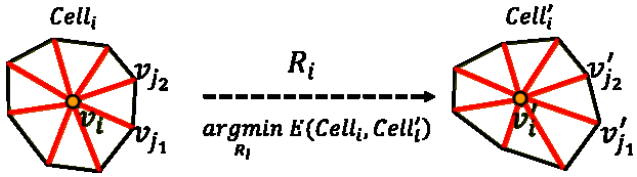



**Fig. 2.** If the position of $Cell_i$ and $Cell_i'$ are known, the rotation matrix $R_i$ is uniquely defined by minimizing the ARAP energy of two cells

If the deformation $Cell_i \to Cell_i'$ is rigid, then there exists a rotation matrix $R_i$ making $E(Cell_i, Cell_i')$ to be zero. However, the deformation is always not rigid, but we can still find an as-rigid-as-possible deformation by minimizing the energy shown as formula (2) in a weighted least squares sense. And the rigidity of a deformation of the whole mesh is to sum up over the deviations from rigidity per cell, then we obtain the following energy function:

$$E_{ARAP}(M, M') = \sum_{i=1}^{n} E(Cell_i, Cell_i') = \sum_{i=1}^{n} \sum_{(v_m, v_n) \in e(v_i)} w_{mn} \left\| (v_m' - v_n') - R_i (v_m - v_n) \right\|^2. \tag{3}$$

The intuitive idea behind this energy function is to allow each cell to have an individual rotation which minimize the energy shown as formula (2), and at the same time to prevent shearing by taking advantage of the overlapping of cells (Fig. 1(c)).

In the deformation tasks, we need to solve for positions $v'$ of $M'$ that minimize energy (3) under some user-defined constraints. As we can see from the energy formula (3), we can interpret $E_{ARAP}(M, M')$ as a function of $v'$ and $\{R_i\}$ $i \in \{1, 2, \ldots n\}$ to enable a simple iterative optimization [4] as follows:

$$\frac{\partial E_{ARAP}(M, M')}{\partial v'} = 0, \tag{4}$$

$$R_i = \arg\min_{R_i} E_{ARAP}(M, M'). \quad \textbf{(5)}$$

Formulae (4) and Formulae (5) show the global position optimization process and local rotation estimation optimization process, respectively. If the rotations $\{R_i\}$ are held fixed then the energy is quadratic in the remaining variable $v'$, in order to compute optimal vertex positions from given rotations, the gradients of $E_{ARAP}(M, M')$ with respect to the vertex positions $v'$ are computed shown as equation (4). Alternatively, if $v'$ are held fixed then each rotation in $\{R_i\}$ is the solution to a localized Procrustes problem (found via singular value decomposition (SVD) or polar decomposition [4]) by solving the equation (5). The above two steps each weakly decrease the energy, thus the iteration can be safely implemented until convergence (the energy is below a certain threshold or iteration times is reached). However, in the local step of optimization as shown in Formulae (5), every rotation in $\{R_i\}$ is attained by the singular value decomposition (SVD) method, which is a time-consuming operation. The resulting computational complexity is prohibitive for large meshes, and leads to failure of real-time mesh animation production with the whole mesh (see the experiment results in Section 4). So, we introduced the bi-harmonic surface based deformation methods to enable the quick convergence of the above local optimization step and realize the real-time editing and animation of the whole mesh with our proposed BS-ARAP deformation method.

### 3.3 Bi-harmonic Surface and Bi-harmonic Displacements

To better understand the proposed deformation framework of BS-ARAP (introduced in Section 3.4), we first introduce the definition of bi-harmonic surface.

A bi-harmonic surface is a smooth polynomial surface which conforms to the bi-harmonic equation. In order to generate a bi-harmonic surface, boundary conditions are usually required. In mathematics, the bi-harmonic equation is a fourth-order partial differential equation. So we casually define bi-harmonic surfaces as surfaces whose position functions are bi-harmonic equation with some initial parameterization:

$$\nabla^4 X = 0 \text{ or } \Delta^2 X = 0, \quad \textbf{(6)}$$

and subject to some handle constraints, in mathematics called boundary conditions:

$$X_b = X_{bc}, \quad \textbf{(7)}$$

where $X$ is the unknown 3D position of a point on the surface. So we are asking that the bi-Laplacian of each of spatial coordinate function to be zero. To solve the bi-harmonic equation (6), we can get a smooth surface that interpolates the handle constraints, but all details on original surface mesh will be smoothed away and the original surface can not be reproduced giving all handles the identity deformation (keeping them at their rest positions).

To realize the "rest pose reproduction", which means the shape should remain unchanged as long as there is no deformation applied to the handles, we consider working with displacements $d$ rather than directly with position $X$. Similarly, the $d$ is obtained by solving the bi-harmonic functions:

$$\nabla^4 d = 0 \text{ or } \Delta^2 d = 0, \quad \textbf{(8)}$$

subject to the same handle constraints:

$$d_b = X_{bc} - X_b. \quad \textbf{(9)}$$

Finally, we add the $d$ to the original surface $M$ to get a bi-harmonic deformation surface $M'$.

### 3.4 Our Deformation Framework: BS-ARAP Deformation

To solve the computational complexity problem in the local step of optimization with ARAP method, we introduced a hierarchy to create a real-time surface mesh deformation framework based on ARAP deformation and bi-harmonic surface construction, called bi-harmonic surface based ARAP (BS-ARAP) method. The flow chart of BS-ARAP method is shown as Fig. 3.

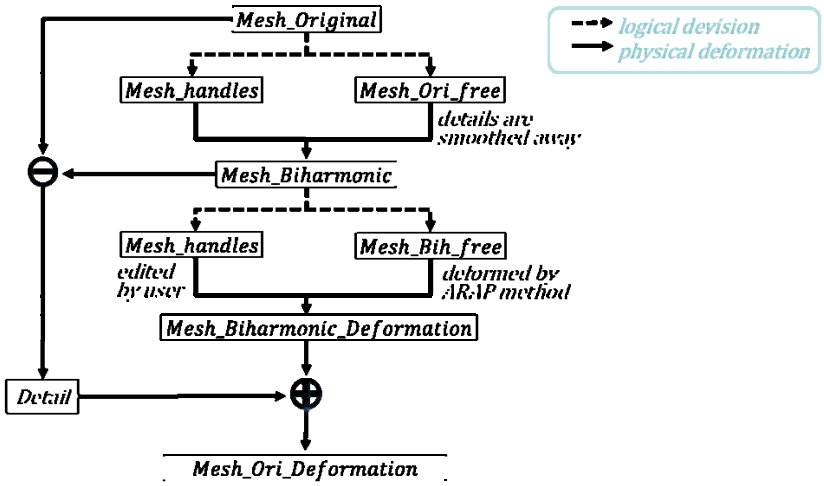



**Fig. 3.** Flow chart of proposed BS-ARAP deformation framework

In the Fig. 3, *Mesh_Original* represents the original mesh model to be deformed; Firstly, some regions of the *Mesh_Original* are selected by users to be handles (formula (7)), and the rest of the mesh (*Mesh_Ori_free*), called as free regions, is smoothed away by differential calculation (formula (6)), then we get the bi-harmonic surface of *Mesh_Original*, i.e. *Mesh_Bi-harmonic*. At the same time, the details of *Mesh_Original* are acquired by computing the difference between *Mesh_Original* and *Mesh_Bi-harmonic*. Secondly, the *Mesh_handles* are edited by users and the *Mesh_Bih_free* (the smooth version of *Mesh_Ori_free*) is deformed by ARAP method automatically, resulting in the deformed bi-harmonic surface *Mesh_Bi-harmonic_Deformation*. Finally, the details of *Mesh_Original* are added back to the *Mesh_Bi-harmonic_Deformation* to get the final BS-ARAP deformation result, i.e. *Mesh_Ori_Deformation*.

In order to explain our deformation framework more intuitively, some experimental results are presented in advance (Fig. 4).

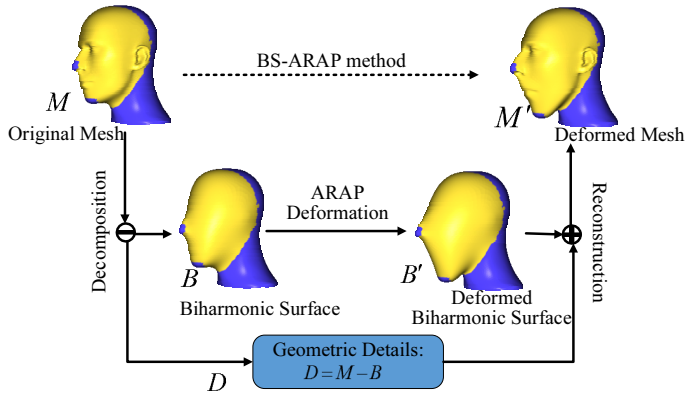


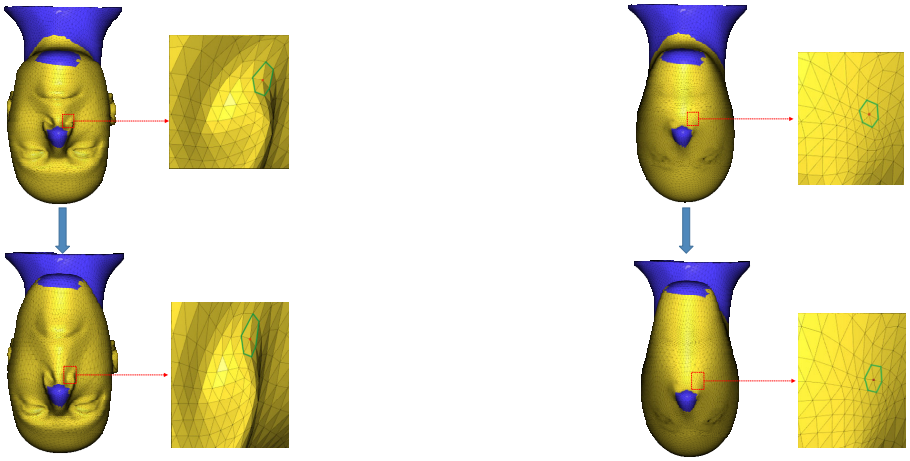

**Fig. 4.** Experimental results of BS-ARAP deformation framework (Taking a head model for example)

Here we take a head model to realize our deformation framework. The original head mesh model $M$ is represented by two colors: golden and blue. Comparing the initial input $M$ and final output $M'$, the blue regions (nose, chin, neck and back of the head) are known as handles and manipulated by users, the golden parts are free and deformed by the implementation of BS-ARAP method. Specific steps are as follows: Firstly, handle regions are selected on original mesh $M$ (formula (7)) to acquire a bi-harmonic surface $B$ by solving equation (6), $B$ represents the low frequency part of the original mesh $M$ considered as a "geometric" signal. The detailed information of the original mesh can be simply acquired by vector subtraction: $D = M - B$ (the connection of the mesh stay the same). Then we can use ARAP method to deform the bi-harmonic surface $B$ (actually just the free golden parts) by controlling the movement of user handles (here we let the vertexes of nose parts move forward and the vertexes of the lower jaw parts move down). Because of the smoothness of bi-harmonic surface $B$, the "shape

matching" problem between $B$ and $B'$ shown as Fig. 2 and formula (5) is easy to solve, leading to the acceleration of convergence speed. And the operation time for final step of BS-ARAP, adding details back onto $B'$ (simple matrix addition operation), can be ignored. In this way, the speed of the whole BS-ARAP method is greatly improved.

### 3.5 Easier Shape Matching with Bi-harmonic Surface

In this part, we intuitively show that the introduction of bi-harmonic surface can improve the optimization process of ARAP deformation method. The optimization process (formula (4) and formula (5)) of ARAP can be described as a 'shape matching' problem, aiming to minimize the 'shape distance' shown as formula (3). Here we use a head model (Fig. 5) to show the deformation results. We edit the blue handles of the model, and the free golden parts are deformed by ARAP method. The number of iteration is set to be 1 to make the difference between the cell deformation of original model and coarse model more obvious. As we can see from Fig. 5(a) and Fig. 5(b), the distortion degree of coarse model cell (marked by green lines) is smaller than original model cell, which means the $E_{ARAP}(M, M')$ (formula (3)) of coarse model is also smaller, resulting the faster convergence rate of iterations process shown as formula (4) and formula (5).



(a) For detailed model, cell distortion degree is relatively bigger and shape matching needs more iterations

(b) For coarse model, cell distortion degree is smaller and shape matching needs less iterations

**Fig. 5.** Intuitive ARAP deformation results with only one iteration

We can also consider shape matching problem as curvature matching problem. To understand that we have to expand the formula (4):

$$\frac{\partial E_{ARAP}(M,M')}{\partial v_i'} = \frac{\partial}{\partial v_i'}\Big(\sum_{(v_m,v_n)\in e(v_i)} w_{mn}\left\|(v_m'-v_n')-R_i(v_m-v_n)\right\|^2\Big)$$

$$= \frac{\partial}{\partial v_i'}\Big(\sum_{\substack{v_m=v_i\\ v_n\in N(i)}} w_{in}\left\|(v_i'-v_n')-R_i(v_i-v_n)\right\|^2 + \sum_{\substack{v_n=v_i\\ v_m\in N(i)}} w_{mi}\left\|(v_m'-v_i')-R_i(v_m-v_i)\right\|^2 + \sum_{\substack{v_n,v_m\neq v_i\\ v_n,v_m\in N(i)}} w_{mn}\left\|(v_m'-v_n')-R_i(v_m-v_n)\right\|^2\Big)$$

$$= \sum_{\substack{v_m=v_i\\ v_n\in N(i)}} 2w_{in}[(v_i'-v_n')-R_i(v_i-v_n)] + \sum_{\substack{v_n=v_i\\ v_m\in N(i)}} -2w_{mi}[(v_m'-v_i')-R_i(v_m-v_i)] \quad \textbf{(10)}$$

$$= \sum_{\substack{m=i\\ v_n\in N(m)}} 4w_{mn}\Big[(v_m'-v_n')-\frac{1}{2}(R_m+R_n)(v_m-v_n)\Big]$$

$$= 0,$$

where, $N(i)$ represents the one-ring neighbors [31] of vertex $i$, and $i = 1,2,...,n$. Finally, we arrive at the following equation:

$$\sum_{\substack{m=i \\ v_n \in N(m)}} w_{mn}(v'_m - v'_n) = \sum_{\substack{m=i \\ v_n \in N(m)}} \frac{w_{mn}}{2}(R_m + R_n)(v_m - v_n) \quad i = 1,2,\ldots,n . \tag{11}$$

From [4], we can know that the left-hand side of formula (11) is non-other than the discrete Laplace-Beltrami operator [31] applied to $v'$. The Laplace-Beltrami, like the Laplacian, is the divergence of the gradient: $\Delta v' = \nabla^2 v' = \nabla \cdot \nabla v'$. The right-hand side of formula (11) can be expressed as a function of $R_i : f(R_i)$. Then the formula (11) can be compactly written as:

$$\Delta v' = f(R_i) . \tag{12}$$

From another aspect, we know that applying the Laplace-Beltrami operator to the surface positions is one way to extract mean curvature $H_i$ [31]:

$$-\Delta v' = H_i . \tag{13}$$

So we can re-express the optimization process (formula (4) and formula (5)) as following:

$$\Delta v' = f(R_i) = -H_i , \tag{14}$$

$$R_i = \arg\min_{R_i} D(H_M, H_{M'}) . \tag{15}$$

Formula (14) shows that global position optimization can be also understood as curvature optimization, aiming to minimize the difference between the mean curvature of mesh $M$ and $M'$ shown as formula (15), where $H_M$ and $H_{M'}$ represent the mean curvature. Then the "shape matching" problem can be converted to 'curvature matching' problem.

From the Fig. 5, we can see that the coarse model based on bi-harmonic surface has a smoother curvature variation against the original model with details, which makes the curvature matching problem easier to solve, and accelerates the convergence speed of iteration shown as formula (14) and formula (15).

## 4 Experiments and Results

We have implemented the BS-ARAP deformation technique using C++ based on the IGL library (LIBIGL) on a laptop with: Linux system, 2.5GHz i7-4710HQ CPU, 8GB of memory. LIBIGL is an open source C++ library for geometry processing research and development and a simple header-only library of encapsulated functions. Our experiments are based on this library.

### 4.1 ARAP Deformation Results

Firstly, we apply ARAP deformation method to a coarse model with only 502 vertices and 1000 faces, called Decimated-Knight (Fig. 6). Three parts of this mesh model (left foot, right foot and the top of head) are selected as handle regions. As we can see from the model shown as Fig. 6, the blue parts are handles controlled by users and golden parts are free regions deformed by ARAP method automatically. We let handles move along a fixed circular path, and the free regions are deformed accordingly, leading to a smooth animation running at 17.6 FPS. In order to show the results, we output the animation frames every second ($t = 0,1,2,3,4s$) and compute general frame numbers according to the frame rate acquired from the experiments. So, this is a successful example of ARAP deformation for real-time animation.

However, when we use models with more vertices and faces to be deformed by ARAP method, the results are disappointing. The Cheburashka model (Fig. 7 upper) has 6669 vertices and 13334 faces, its feet and hands parts are selected to be handles (blue color) and moved by us, then the free parts (golden color) are deformed by ARAP method, but the simulation runs at a very low frame rate. Running from the first frame (rest pose) to the second frame spends about 1.93 second (the first move needs some initialization work leading to additional cost), then the animation runs at a steady rate 1.36 FPS. The Armadillo model (Fig. 7 lower) has 43243 vertices and 86482 faces, and its first move from the first frame to second frame spends 9.46 second, and later the simulation runs at 0.26 FPS. Although we only

move the feet handles (keep other handles still) of Armadillo model, the animation frame rate is still very low.

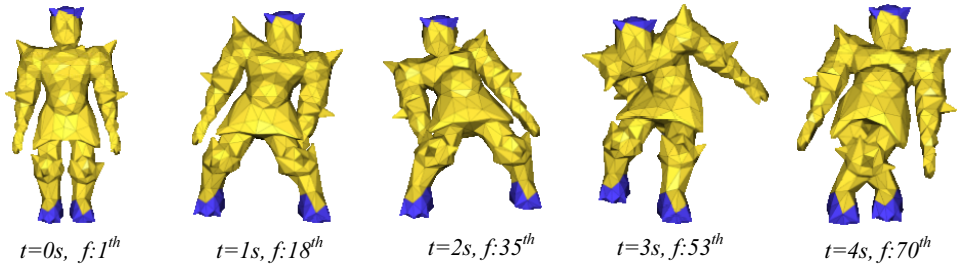



**Fig. 6.** The animation runs at 17.6 FPS; The leftmost picture ($t=0s$, $f:1^{th}$) shows the first frame (rest pose) of the model animation at 0 second (simulation not started); Animation frames are also given when simulation time reaches at $1^{th}$,$2^{th}$,$3^{th}$,$4^{th}$ second from the beginning and the frame number are computed and denoted by $f$

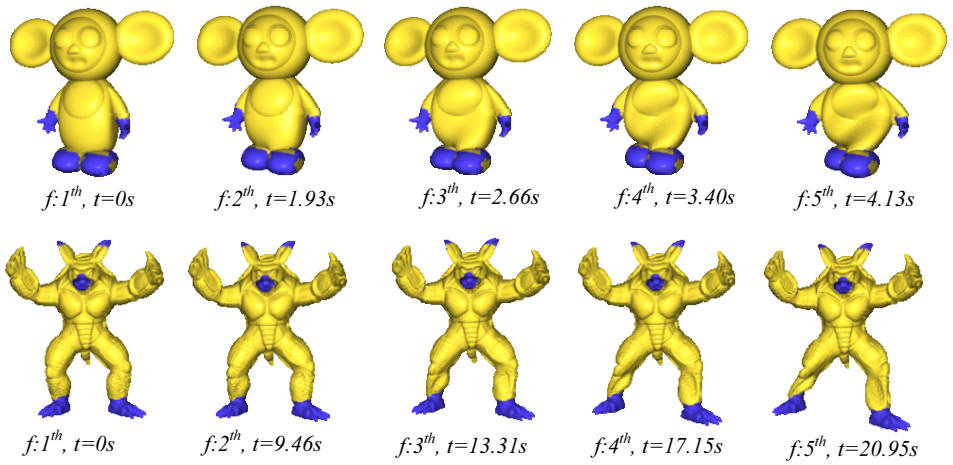
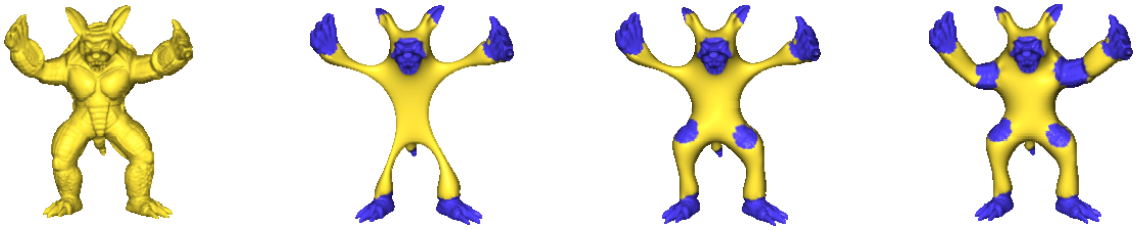



**Fig. 7.** The animation of Cheburashka model (upper) runs at 1.36 FPS (starts from the second frame) and Armadillo model (lower) runs at 0.26 FPS; The first five animation frames are shown and the corresponding simulation time is also given. (These are two failure examples of ARAP deformation method because that the two models have much more vertices than the previous model named Decimated-Knight in Fig. 6)

### 4.2 BS-ARAP Deformation Results

To speed up the deformation, we introduce the bi-harmonic surface to improve the ARAP deformation method, called BS-ARAP. We use different models to test BS-ARAP method, the results show great improvement in simulation speed.

We take the Armadillo model for example to show the steps and results of BS-ARAP deformation framework (we have already shown one example in Fig. 4). Firstly, in order to construct the bi-harmonic surface for Armadillo model, handles need to be selected. Fig. 8 shows the different resulting bi-harmonic surface with different handle count, we get a best result when twelve handle regions are selected (palms, feet, face, ears, tail, thigh and shoulder), shown as Fig. 8(d). So Fig. 8(d) is the final bi-harmonic surface (also known as base surface $B$) of the original Armadillo model denoted by $M$ shown as Fig. 8(a). And the geometric details are represented as a displacement function $h: B \to M$ of the base surface. Then we use ARAP method to deform the coarse version of Armadillo, i.e. base surface $B$, and the simulation runs at 9.3 FPS shown as upper Fig. 9. At the same time, the displacement vector $h$ rotates with the rotation of the deformed base surface $B'$ so that the surface details can be preserved well, shown as lower Fig. 9, which are obtained simply by adding $h$ back to $B'$.



(a) is the original mesh model

(b) is the constructed bi-harmonic surface by selecting 5 handle regions on (a): palms, feet, face, ears and tail

(c) is constructed BS by adding thigh handle regions based on (b)

(d) is the final best result by adding shoulder handle regions based on (c)

**Fig. 8.** Bi-harmonic surface construction of Armadillo model

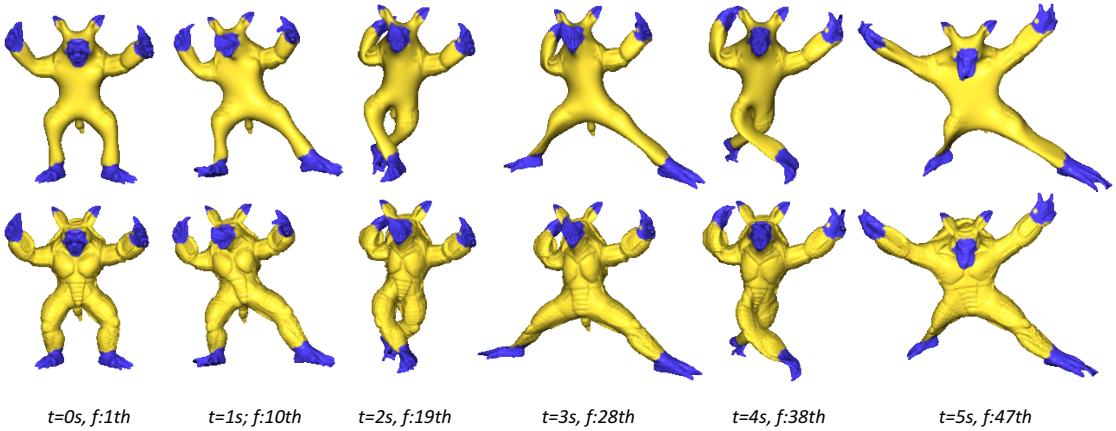



**Fig. 9.** The animation frames of BS-ARAP deformation results with Armadillo model (lower) and its bi-harmonic surface version (upper). The animation runs at 9.3 FPS, and several frames are output at t=1,2,3,4,5s with corresponding frame numbers (for example, '*t=1s, f:10th*' means the animation frame at first second, which is the 10th frame of the animation)

Compared with the simulation with the ARAP deformation method with a 0.26 FPS rate (as for the Armadillo model), the simulation of BS-ARAP method runs at 9.3 FPS, obtaining a 35.8× speedup. We also used other models to test the BS-ARAP deformation method, their results are shown as Table 1. In the programming, a triangle mesh model with $N$ vertices and $M$ faces is represented by a position matrix ( $N \times 3$ ) and connection matrix ( $M \times 3$ ). During the deformation, the model's connection matrix remains unchanged and only the position matrix continues to change, so completing a position matrix's calculation means a new animation frame is obtained. When different models are deformed, we record the average time for computing a position matrix which is denoted by SPF, meaning the computation time per frame in seconds. And we obtain the animation frame rate by computing the reciprocal of SPF. In the experiments, we set the maximum frame rate to 30 FPS. From the results shown as Table 1, we get a least 10× speedup, which proves the the feasibility and effectiveness of the improved BS-ARAP deformation method.

### 4.3 Real-time Performance of BS-ARAP Method

In this part, I would like to analyze the real-time performance improvement of our method in comparison to other deformation methods.

**Table 1.** Comparison of each frame's computation time between ARAP and BS-ARAP methods with different models

| Models | Vertices | faces | SPF | | FPS | | Speedup |
|---|---|---|---|---|---|---|---|
| | | | ARAP | BS-ARAP | ARAP | BS-ARAP | |
| Decimated-Knight.off | 502 | 1000 | 0.0568s | 0.0056s | 17.6 | ≥ 30 | 10.1× |
| Cheburashka.off | 6669 | 13334 | 0.733s | 0.0284s | 1.36 | ≥ 30 | 25.8× |
| Bunny.obj | 34813 | 69630 | 3.207s | 0.0987s | 0.31 | 10.1 | 32.5× |
| Armadillo.obj | 43243 | 86482 | 3.830s | 0.108s | 0.26 | 9.3 | 35.8× |
| Tyra.obj | 100002 | 200000 | 10.224s | 0.261s | 0.097 | 3.8 | 39.2× |

Firstly, as we can see from the experimental results in Table 1, our deformation method (BS-ARAP) shows great improvement in deformation rate compared with original ARAP method. As we have described in sections 3.4 and 3.5, it is because constructing bi-harmonic surface of original mesh surface helps to speedup the ARAP optimization process, which is proved by the results in figure 10. We use "decimated-max" model (5272 vertices and 10540 faces) in our experiment. The steps of our experiments are as follows:

**Step 1.** Read mesh model files $M$ $D$_select handles of model $M$

**Step 2.** Compute the bi-harmonic surface of $M$ according to the selected handles and denote it by $M_{bs}$.

**Step 3.** Compute Laplacian coordinates of each vertex in mesh and $M_{bs}$. Then edit handles of $M$ and $M_{bs}$.

**Step 4.** Propagate the manipulation of handles to the whole mesh by fitting the Laplacian coordinates to the original Laplacians in rest poses acquired in step3. Then we obtain deformed model $M'$ and bi-harmonic surface $M'_{bs}$.

**Step 5.** Compute the $E_{ARAP}(M, M')$ and $E_{ARAP}(M_{bs}, M'_{bs})$ according to formula (3). The ARAP energy is normalized to a 0~1 scale.

**Step 6.** Implement the optimization process (formula (4) and (5)) until the ARAP energy is below the specified threshold, and record the number of iterations ( $N$ ).

From the figure 10, we can see that when we allow the optimization process to run on a coarse version of model, i.e. bi-harmonic surface of original model (part (b) in figure 10), the iteration times decrease a lot, leading to the great improvement on computation speed.

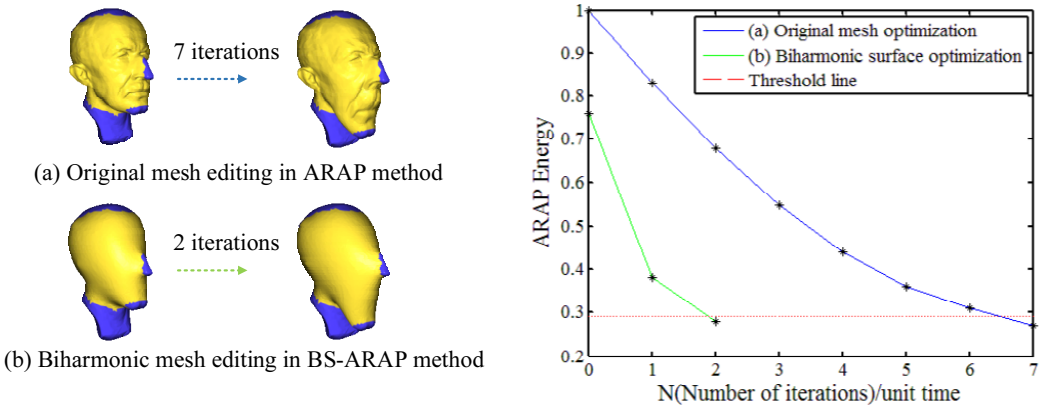



**Fig. 10.** The number of iterations in optimization process of ARAP and BS-ARAP method

Secondly, we conduct more experiments to compare BS-ARAP method with other deformation methods. We still use the Armadillo model with 43243 vertices and 86482 faces. The results are shown in Fig. 11. Our method still shows better real-time animation performance compared with fast automatic skinning transformations method (Fig. 11(b)) and fast subspace method (Fig. 11(c)) without considering the deformation quality during the animation. Actually, it is difficult to find an objective way to assess the deformation quality during an animation. In most cases, the quality of animation is evaluated by human vision. As for our deformation method, as can be seen from Fig. 11, we can still acquire

intuitively reasonable deformation results when we improve the real-time performance.

(a) Animation of Armadillo model deformed by BS-ARAP deformation method, running at 9.3 FPS

(b) Fast automatic skinning transformations method. The simulation runs at 8.7 FPS

(c) Fast subspace method

**Fig. 11.** The simulation at 6.0 FPS

## 5 Conclusions and Future Work

The important features of our approach are: (1) fast convergence speed, resulting from the introduced hierarchies based on bi-harmonic surface; (2) details preservation, as we use the APAR deformation method to deform the base surface and details are independently represented by the displacement vector.

We introduced a new ARAP-type animation technique that is based on a new BS-ARAP deformation framework. Our technique makes up for the biggest shortcoming in the popular ARAP methodology. It allows the detailed mesh model to be smoothed away by constructing bi-harmonic surfaces, which accelerates the iterative process of local optimization in ARAP method. And the details expressed by the displacement vector are added back to the deformed bi-harmonic surfaces to achieve intuitive detail preservation. We demonstrated the effectiveness of our method (BS-ARAP) in the application of mesh animation, which is superior in computation speed to prior ARAP [5] or SR-ARAP [6] method.

Work continues to find more ways to speed up the ARAP method. For example, we can vary rigidity across the surface, paying more attention to the rigidity of interested regions. Furthermore, we can reduce the degrees of freedom to better reflect the frequency of the desired deformations. As for the BS-ARAP methods we improved in this paper, collision detection can be implemented to avoid self-intersections in the animation, enhancing the animation authenticity. Moreover, as we have mentioned in the introduction part, we have not compared the deformation quality in terms of detail preservation between the ARAP method and BS-ARAP method, so it can also be the next research topic.

## Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61373084) and Innovation Program of Shanghai Municipal Education Commission (No. 14YZ011).

## References

[1] N. Magnenat-Thalmann, R. Laperriere, D. Thalmann, Joint-dependent local deformations for hand animation and object grasping, in: Proc. Graphics Interface ’88, 1988.

[2] A. Mohr, L. Tokheim, M. Gleichier, Direct manipulation of interactive character skins, in: Proc. 2003 Symposium on Interactive 3D Graphics Conference, 2003.

[3] J.P. Lewis, M. Cordner, N. Fong, Pose space deformation- a unified approach to shape interpolation and skeleton-driven deformation, in: Proc. the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000.

[4] O. Sorkine, M. Alexa, As-rigid-as-possible surface modeling, in: Proc. the Fifth Eurographics Symposium on Geometry

Processing, 2007.

[5] I. Chao, U. Pinkall, P. Sanan, P. Schröder, A simple geometric model for elastic deformations, in: Proc. ACM SIGGRAPH ’10, 2010.

[6] Z. Levi, C. Gotsman, Smooth rotation enhanced as-rigid-as-possible mesh animation, IEEE Transactions on Visualization and Computer Graphics 21(2)(2015) 264-277.

[7] L. Moser, D. Corral, D. Roble, As-rigid-as-possible deformation transfer for facial animation, in: Proc. ACM SIGGRAPH, 2016.

[8] D. Terzopoulos, J. Platt, A. Barr, K. Fleischer, Elastically deformable models, in: Proc. the 14th Annual Conference on Computer Graphics and Interactive Techniques, 1987.

[9] G. Celniker, D. Gossard, Deformable curve and surface finite-elements for free-form shape design, in: Proc. the 18th Annual Conference on Computer Graphics and Interactive Techniques, 1991.

[10] W. Welch, A. Witkin, Variational surface modeling, in: Proc. the 19th Annual Conference on Computer Graphics and Interactive Techniques, 1992.

[11] M. Botsch, L. Kobbelt, An intuitive framework for real-time freeform modeling, ACM Transactions on Graphics 23(3)(2004) 630-634.

[12] D. Zorin, P. Schroder, W. Sweldens, Interactive multi-resolution mesh editing, in: Proc. the 24th Annual Conference on Computer Graphics and Interactive Techniques, 1997.

[13] I. Guskov, K. Vidimce, W. Sweldens, P. Schroder, Normal meshes, in: Proc. the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000.

[14] A. Lee, H. Moreton, H. Hoppe, Displaced subdivision surfaces, in: Proc. the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000.

[15] L. Kobbelt, J. Vorsatz, H.-P. Seidel, Multiresolution hierarchies on unstructured triangle meshes, Computational Geometry Journal: Theory and Applications 14(1-3)(1999) 5-24.

[16] M. Botsch, L. Kobbelt, Multiresolution surface representation based on displacement volumes, Computer Graphics Forum 22(3)(2003) 483 -491.

[17] M. Botsch, R.W. Sumner, M. Pauly, M. Gross, Deformation transfer for detail-preserving surface editing, in: Proc. Vision, Modeling, and Visualization, 2006.

[18] R.W. Sumner, J. Popovic, Deformation transfer for triangle meshes, ACM Transactions on Graphics 23(3)(2004) 399-405.

[19] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, H.-Y. Shum, Mesh editing with Poisson-based gradient field manipulation, ACM Transactions on Graphics 23(3)(2004) 644-651.

[20] R. Zayer, C. Rossl, Z. Karni, H.-P. Seidel, Harmonic guidance for surface deformation, Computer Graphics Forum 24(3)(2005) 601-609.

[21] M. Alexa, Differential coordinates for local mesh morphing and deformation, The Visual Computer 19(2)(2003) 105-114.

[22] O. Sorkine, D. Cohen-Or, S. Toledo, High-pass quantization for mesh encoding, in: Proc. Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 2003.

[23] Y. Lipman, O. Sorkine, D. Cohen-Or, D. Levin, C. Rossi, H.P. Seidel, Differential coordinates for interactive mesh editing, in: Proc. Shape Modeling Applications, 2004.

[24] K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, H.-Y. Shum, Large mesh deformation using the volumetric graph laplacian, ACM Transactions on Graphics 24(3)(2005) 496-503.

[25] J. Noh, U. Neumann, Expression cloning, in: Proc. The 28th Annual Conference on Computer Graphics and Interactive Techniques, 2001.

[26] R.W. Sumner, M. Zwicker, C. Gotsman, J. Popovic, Mesh-based inverse kinematics, ACM Transactions on Graphics 24(3)(2005) 488-495.

[27] K.G. Der, R.W. Sumner, J. Popovic', Inverse kinematics for reduced deformable models, ACM Transactions on Graphics 25(3)(2006) 1174-1179.

[28] V.A. Toponogov, V. Rovenski, Differential Geometry of Curves and Surfaces: A Concise Guide, Birkhäuser, Boston, 2006.

[29] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, H.-P. Seidel, Laplacian surface editing, in: Proc. 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, 2004.

[30] M. Meyer, M. Desbrun, P. Schröder, A.H. Barr, Discrete differential-geometry operators for triangulated 2-manifolds, Visualization and Mathematics III 6(8-9)(2003) 35-57.

[31] U. Pinkall, K. Polthier, Computing discrete minimal surfaces and their conjugates, Experimental Mathematics 2(1)(1993) 15-36.