International Journal of CAD/CAM Vol. 6, No. 1, pp. 89~97 (2006)

INTERNATIONAL JOURNAL OF CAD/CAM
www.ijcc.org

Non-iterative Global Mesh Smoothing with Feature Preservation

Zhongping Ji[1], Ligang Liu[1]* and Guojin Wang[2]
[1]*Department of Mathematics, Zhejiang University, Hangzhou, 310027, China*
[2]*State Key Lab of CAD&CG, Zhejiang University, Hangzhou, 310027, China*

Abstract – This paper presents a novel approach for non-iterative surface smoothing with feature preservation on arbitrary meshes. Laplacian operator is performed in a global way over the mesh. The surface smoothing is formulated as a quadratic optimization problem, which is easily solved by a sparse linear system. The cost function to be optimized penalizes deviations from the global Laplacian operator while maintaining the overall shape of the original mesh. The features of the original mesh can be preserved by adding feature constraints and barycenter constraints in the system. Our approach is simple and fast, and does not cause surface shrinkage and distortion. Many experimental results are presented to show the applicability and flexibility of the approach.

Keywords **: triangular mesh, mesh smoothing, laplace operator, least squares, feature preservation**

1. Introduction

Large and complex mesh models are often obtained from points sampled over real-world objects. Recent advances of 3D scanning and acquisition technology call for developing robust and reliable methods for processing large and complex point and mesh data sets. But due to the inevitable physical noise added by a scanning device, points sampled from a 3D object often do not reflect their correct locations, resulting in meshes containing undesirable rough features. Developing fast and robust methods for removing noise has become one of the most important 3D data processing operations [21,29].

Mesh smoothing, or denoising, is a process dedicated to the removal of noise with minimal damage caused to geometric features of the object. The ultimate goal of mesh smoothing is to produce highly smooth meshes efficiently, for rendering, modelling, and visualization, while still preserving the basic overall shape and important features of the original model. However, removing noise while preserving the features of the shape is not a trivial matter. Furthermore, scanned meshes often have cracks and non-manifold regions, which makes the matter worse [13].

1.1. Previous work

A great deal of mesh smoothing algorithms has been proposed in the literature. Smooth meshes are characterized by low discrete curvature. The operators by which curvatures can be computed on meshes lead to simple filter algorithms that improve the smoothness of an existing mesh by moving the vertices in order to minimize the discrete curvature.

The most common techniques are based on Laplace smoothing [4,10]. Taubin [32] introduced signal processing on surfaces that is based on the definition of the Laplacian operator on meshes and developed a fast and simple iterative Laplacian smoothing scheme. Desbrun et al.[5] extended this approach to irregular meshes using a geometric flow analogy. Ohtake et al. [24] extended the Laplace smoothing by combining geometry smoothing with parameterization regularization. Peng et al. [28] applied locally adaptive Wiener filtering to meshes.

However, these techniques are all isotropic, and therefore indiscriminately smooth noise as well as salient features, leading to shrinkage or undesired distortion of the mesh shape. To compensate these drawbacks, Liu et al. [19] proposed a method that keeps the volume of each star of a vertex. Vollmer et al. [36] suggested a method that is based on the idea to push the vertices back to their previous positions. Hildebrandt et al [12]. proposed an anisotropic smoothing scheme that reduces diffusion across edges. To achieve this, they modified the curvature normal operator by reducing the weight of terms that have a magnitude greater than a threshold.

Feature-preserving mesh smoothing methods [1,3,6] were mostly inspired by anisotropic diffusion in image processing [27] more recently. These methods modified the diffusion equation to make it nonlinear or anisotropic, thus could preserve sharp features. The work of [25,33,35] proposed diffusion-type smoothing on the normal field first, and then constructed the surface to match the new normals. Although these approaches are superior to those using isotropic techniques, but they would cause significant computational times.

Recently, Jones et al. [13] proposed a statistical method to anisotropically smooth a mesh in one pass.

*Corresponding author:
Tel: +86-571-87953668
Fax: +86-571-89753668
E-mail: ligangliu@zju.edu.cn

This approach predicts the location of a vertex based on its neighbors. Robust statistics are used to deemphasize the contribution of vertices dissimilar to the one being predicted. Fleishman et al.[8] introduced a similar method based on bilateral filtering that is iterative. Bilateral filtering was originally formulated for image processing, and is a non-linear variation of Gaussian smoothing that weights sample points based on their similarity to the one being processed. However, it is not straightforward to assign appropriate parameters to get good results in the algorithms.

1.2. Our approach and contributions

In previous Laplacian smoothing algorithm, a local iterative procedure is used to update the poistions of the vertices. The new position of a vertex may not solely depend on the set of old positions of its adjacent vertices but can depend on their previously calculated new positions, too. Hence, the result of one smoothing pass through all vertices will depend on the order how the vertices are considered.

Unfortunately, local Laplacian smoothing leads to a variety of artifacts such as geometric distortion and shrinkage due to the irregular connectivity of the mesh. Moreover, the iteration is sometimes numerically unstable, which seriously slows down the convergence. Furthermore, it provides insufficient control over global behavior during smoothing for irregular connectivity meshes.

In this paper, we take a different approach for smoothing arbitrary meshes with feature preservation. We consider about the mesh smoothing as a problem of finding an approximating surface with a global minimization of a surface fairing energy. Thus we adopt the Laplacian operator in a global way instead of a local way. The smoothed mesh is constructed by solving a spare linear system, which is fast and efficient, in a non-iterative way.

Feature constraints and barycenter constraints are also considered in our approach to keep the features of the original noisy mesh.

To our knowledge, our smoothing approach is the first approach to handle the issue of smoothing using non-iterative Laplacian operator in a global way. The contributions of our global smoothing for arbitrary meshes are summarized in the following:

- **Global smoothing**: The Laplacian operator is performed over the mesh but not over each vertex locally. The smoothed mesh is obtained by performing the global Laplacian operator.
- **Non-iterative**: Our global smoothing approach is non-iterative. The smoothed mesh can be constructed by solving a sparse linear system.
- **Feature preserving**: The smoothed mesh can keep the features of the noisy mesh without shrinkage and distortion by adding feature constraints and barycenter constraints in the linear system.
- **Fast and efficient**: Our approach is fast and efficient as it only needs to solve a sparse linear system which can be effectively solved by Cholesky factorization of the sparse matrix.

1.3. Overview

The paper is organized as follows. Section 2 gives the mathematical formulization for global smoothing and introduces the traditional Laplacian smoothing method. Our approach of global surface smoothing with feature preservation is described in Section 3. Section 4 discusses the implementations of our algorithm. Additional experimental results are illustrated in Section 5. We conclude the paper in Section 6 with the summary and future work.

2. Preliminaries

2.1. Surface smoothing---a global optimization problem

We start with formulating the problem of global surface smoothing more precisely: *Given a mesh surface S with geometric noise, our goal is to produce a smooth and good quality mesh surface S' which is as close as possible to S, and preserves the features of S.*

For a parametric surface, the problem of the global surface smoothing can be formulized as a mathematical problem:

$$\min_{S'} E(S') \tag{1}$$

where the minimized energy has to contain two terms: one that measures the fairness of the surface and the second that takes the resemblance with the original surface (data fidelity) into consideration:

$$E(S') = \underbrace{\alpha \int_\Omega \Psi(S')\,du\,dv}_{\text{smoothness constraint}} + \underbrace{\beta \int_\Omega (S'-S)^2\,du\,dv}_{\text{Data fidelity}}$$

where α and β are the weights for the two terms. The above minimization model for surface smoothing is generalized from the total variational model of Rudin-Osher-Fatemi well-known in the literature of image denoising [30].

The smoothing energy term used for smoothing can be the membrane energy [37]:

$$\int_\Omega \Psi(S')\,du\,dv = \int_\Omega (F_u^2 + F_v^2)\,du\,dv,$$

or the thin-plate energy [7,23]:

$$\int_\Omega \Psi(S')\,du\,dv = \int_\Omega (F_{uu}^2 + 2F_{uv}^2 + 2F_{vv}^2)\,du\,dv.$$

The effects of these approximation errors on the resulting minimum energy surfaces are very difficult to estimate. In the discrete setting, the integrals can be approximated by a weighted sum over all vertices (quadrature formula) and the partial derivatives are approximated by divided differences [15,17,32].

2.2. Traditional local Laplacian smoothing

In this section the original version of the Laplacian smoothing will be reviewed.

Let $S=(V, E, F)$, where $V=\{v_i|i=1,2,\ldots,n\}$ is the set of vertices, E is the set of edges, $F=\{T_i|i=1,2,\ldots,m\}$ and is the set of triangles in which each triangle T_i can be represented by a triple of vertex indices as $T_i=<i_1, i_2, i_3>$, $1 \le i_1, i_2, i_3 \le n$.

The Laplacian operator can be linearly approximated at each vertex by the umbrella operator as used in [15,32]:

$$L(v_i)=\sum_{j\in i^*} w_{ij}(v_j-v_i), \tag{2}$$

where i^* is the vertex index set of neighborhood vertices to the vertex v_i, and w_{ij} is the weight of edge (i, j)corresponding to vertex v_i with $\sum_{j\in i^*} w_{ij}=1$. Several weighting schemes have been proposed, such as edge length scheme and cotangent scheme [5,32]. In our approach, we use uniform weighting for mesh smoothing since smoothing aims to improve the quality of the triangles in some sense.

The Laplacian algorithm is quite simple: the basic idea is that the position of vertex v_i is replaced with the average of the positions of adjacent vertices, see Fig. 1. Practically the vertices of a mesh are incrementally moved in the direction of the Laplacian.

The simplicity of the algorithm is based on very basic, uniform approximations of the Laplacian. For irregular connectivity meshes this leads to a variety of artifacts such as geometric distortion, slow convergence for large meshes, numerical instability, and insufficient control over global behavior during smoothing. The latter includes shrinkage problems and more precise shaping of the frequency response of the algorithms.

3. Non-iterative Global Smoothing

Instead of moving the vertices locally and iteratively in the traditional Laplacian smoothing approach, we would like to solve all the vertices of the smoothed surface in a global manner.

3.1. Global Laplacian operator

For a vertex v_i its smooth condition is:

$$\sum_{j\in i^*} w_{ij}(v_j-v_i)=0, \tag{3}$$

where w_{ij} is the weight as in Eq.2 [9]. Eq.3 means that vertex v_i lies in the weighted average of its 1-ring neighbors. It is known that the vertex v_i lies in the center of gravity of its 1-ring neighbors if we choose $w_{ij}=1/d_i$, where $d_i=\#i^*$ is the valence of v_i, in Eq.3.

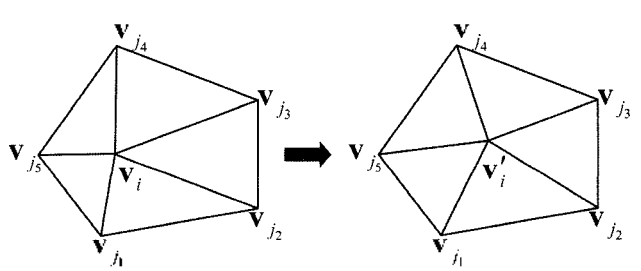


Fig. 1. The traditional laplacian smoothing algorithm.

Suppose all the vertices v_i, $i=1,2,...,n$ are unknown. It can be seen that the equations in Eq.3 of all the vertices form a sparse linear system. The linear system can be written in matrix form:

$$\mathbf{LX}=0, \tag{4}$$

where $\mathbf{L}$ is an $n\times n$ matrix with elements derived from w_{ij} :

$$L_{ij}=\begin{cases} 1, & i=j, \\ -w_{ij}, & (i,j)\in E, \\ 0, & otherwise, \end{cases}$$

$\mathbf{X}$ is the $\mathbf{n}\times 1$ column vector of the corresponding vertices.

The matrix $\mathbf{L}$ is called the *Laplacian* of the mesh S [16,32]. Note that $\mathbf{L}$ has rank $n-1$ for a connected mesh surface, which means, given $\mathbf{L}$, the vertex positions $\mathbf{X}$ can not be uniquely determined by solving the linear systems without any vertex position given.

If the geometry of some of the vertices are provided, we can reconstruct the geometry of the rest of the mesh vertices by solving the sparse linear system in Eq.4 in a least square sense like in [31]. As each vertex lies as close as possible to the weighted center of its 1-ring neighbors, the vertices on the reconstructed mesh surface are distributed over the surface in a fair way. Furthermore, if we carefully select the provided vertices as feature points of the surface (as we can see in the following sections), the reconstructed mesh can effectively approximate the given mesh. Thus the linear system (Eq.4) defines a surface that is visually smooth and fair approximation of the given mesh in a global way.

3.2. Feature constraints

In order to keep the features of the original mesh surface S, one or more feature points on S are first detected as $\{v_k=(x_k, y_k, z_k)|k\in C\}$, where $C=\{i_1,i_2,\mathrm{L},i_s\}$ is the set of indices of the feature vertices. We also allow the user selects a set of vertices as the feature points.

The system reconstructs the positions of all the vertices v' of S′ to minimize the following error functional:

$$\min_{\mathbf{X}'}|\boldsymbol{LX}'|^2+\mu^2\sum_{k\in C}|\mathbf{V}'_k-\mathbf{V}_k|^2, \tag{5}$$

where μ is the weight of the feature vertex constraints. The first term in Eq.5 is the error of Laplace operator and the second term is the error of the feature vertex constraints.

The above functional is quadratic in every vertex and hence its partial derivatives are linear expressions. The

unique minimum is found if all partial derivatives with respect to the vertices vanish, which results in a sparse linear system as the following:

$$\mathbf{AX}'=\begin{pmatrix}\mathbf{L}\\ \mathbf{F}\end{pmatrix}\mathbf{X}'=\begin{pmatrix}0\\ \mathbf{b}^F\end{pmatrix}=\mathbf{b}, \tag{6}$$

where $\mathbf{F}$ is an $s \times n$ matrix in which each row contains only one non-zero element used to constrain the position of the feature vertices with the element:

$$f_{kj}=\begin{cases}\mu, & j=i_k \in C,\\ 0, & \text{otherwise},\end{cases} \quad (1\le k\le s, 1\le j\le n;)$$

and $\mathbf{b}^F$ is an $s \times 1$ column vector of the product of feature vertices and μ:

$$b_k^F = \mu g_{ik},\ 1 \le k \le s, g = x, y, \text{ or } z.$$

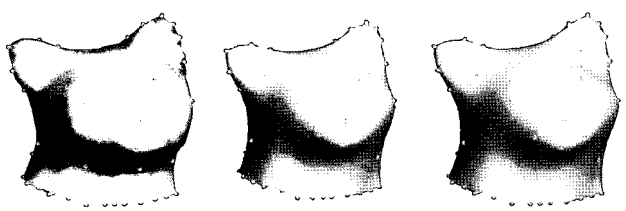
Note that the linear system is defined for each component of the coordinates x, y, and z. The positions of the vertices can be found by solving the sparse linear system in Eq. 6 in a least square sense as:

$$\mathbf{X}' = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}.$$

Fig. 2(a) shows a cat model with selected feature points shown in red color. In Fig. 2(b) and (c), the smoothed cat models are obtained by solving the sparse linear system in Eq. 6 with $\mu = 1$ and $\mu = 5$ respectively. Note that the smoothed mesh in Fig. 2(b) shrinks much than in Fig. 2(c). It can be seen that the reconstructed smooth mesh approximates the feature points closer as we increase the weight in μ Eq. 6. In our experimentations, we found that the reconstructed mesh almost interpolates the feature vertices when we set $\mu = 5$.

3.3. Barycenter constraints

Although the smoothed mesh surface S' could preserve the features of S by approximating the feature vertices in the minimization problem in Eq.5, the least squares solution to the minimization problem would still lead to undesired distortion and shrinkage in region of non-feature vertices, see Fig. 2(b) and (c).



(a) (b) (c)

Fig. 2. Global Laplacian xmoothed mesh with feature vertex constraints. The feature vertices are highlighted in red color. (a) The original cat mesh; (b) smoothed mesh with $\mu = 1$; (c) smoothed mesh with $\mu = 5$.

The problem of distortion and shrinkage occurs because the Laplacian operator causes too much relaxation on the vertices. Therefore, we can introduce extra constraints to Eq.5 to reduce the vertex relaxation. Like in [18], we add the triangle barycenter constraints in the minimization problem in Eq.5 to control the vertex relaxation. That is, we fix all the triangle barycenters in position during smoothing, which can effectively prevent its distortion and shrinkage.

The barycenter constraint for a triangle $T = \langle i, j, k\rangle$ can be described as:

$$(v'_i + v'_j + v'_k)/3 = (v_i + v_j + v_k)/3$$

Thus we try to find the solution of the following minimization problem

$$\min_{\mathbf{X}'}|L\mathbf{X}'|^2+\mu^2\sum_{k\in C}|v'_k-v_k|^2$$

$$\sum_{\langle i,j,k\rangle\in F}\lambda^2|(v'_i+v'_j+v'_k)-(v_i+v_j+v_k)|^2, \tag{5}$$

where λ is the weight of the barycenter constraint.

The set of coordinates of v' is found by minimizing the above error functional. Solving this quadratic minimization problem results in a sparse linear system:

$$\bar{\mathbf{A}}\mathbf{X}'=\begin{pmatrix}\mathbf{L}\\ \mathbf{F}\\ \mathbf{Z}\end{pmatrix}\mathbf{X}'=\begin{pmatrix}0\\ \mathbf{b}^F\\ \mathbf{b}^z\end{pmatrix}=\bar{\mathbf{b}}, \tag{7}$$

where $\mathbf{z}$ is an $m \times n$ matrix in which the k-th row contains only three non-zero elements used to constrain the position of barycenter of the corresponding triangle $T_k = \langle i_1, i_2, i_3\rangle$with the element:

$$z_{ki}=\begin{cases}\lambda, i=i_1,i_2,i_3, & T_k=\langle i_1,i_2,i_3,\rangle,\\ 0, \text{otherwise}, & 1\le k\ m,\ 1\le i\le n,\end{cases}$$

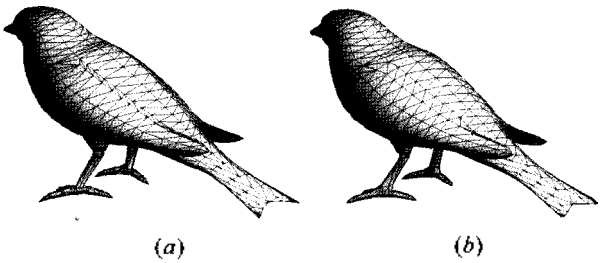
and $\mathbf{b}^z$ is an m × 1column vector with the element:

$$b_k^z=\lambda(g_{i1}+g_{i2}+g_{i3}),\ T_k=\langle i_1,i_2,i_3\rangle, 1\le k\le m,\ g=x,y,\text{or } z.$$

Note that the linear system is also defined for each component of the coordinates x, y, and z. The positions of the vertices can be found by solving the linear system in Eq.7 as:

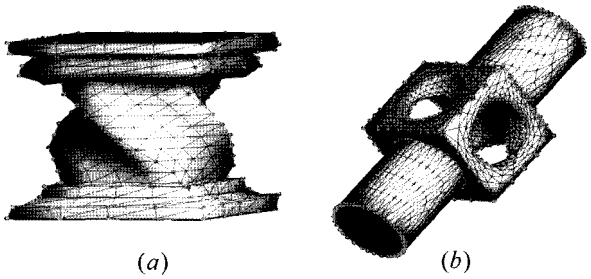
$$\mathbf{X}'=(\bar{\mathbf{A}}^T\bar{\mathbf{A}})^{-1}\bar{\mathbf{A}}^T\bar{\mathbf{b}}.$$

Fig. 3 shows a smoothing example illustrating the effect of barycenter constraints. In this example, no feature point is selected, i.e., $s = 0$and we set $\lambda = 0.3$. It can be seen from Fig.3(b) that our approach does not shrink the mesh using the barycenter constraints and the sharp feature vertices, such as vertices at the end of the



(a) (b)

Fig. 3. Global Laplacian smoothed mesh with barycenter constraints. (a) the original bird mesh; (b) the smoothed mesh with barycenter constraints with parameter $\lambda = 0.3$.



(a) (b)

Fig. 4. Detection results of feature points. The detected feature points on the meshes are shown in red color.

bird tail, are smoothed out as there are no feature constraints there. Moreover, the vertices on the smoothed mesh are distributed more uniformly over the surface.

It can be observed that the feature constraints and barycenter constraints are basically the same types of constraints applied on vertices, i.e., they both penalize the displacement of vertices from their original positions, since the barycenter constraints are actually a weighted average of the feature constraints. In fact, these two types of constraints can be regarded together as vertex displacement constraints.

3.4. Boundary smoothing

For non-closed meshes, part of the neighborhood is not defined for boundary points. In our system, the boundary curves are smoothed separately and our algorithm handles boundaries by treating them as sharp feature edges. Adopting the same idea mentioned in the previous sections, the feature constraints and center constraints may also be introduced to keep the features of the boundary curves and to prevent the boundary curves from shrinkage and distortion.

3.5. Detection of feature vertices

Feature extraction has been studied research area in many scientific fields [2,14,26]. As the definition of feature is highly subjective and is very difficult to express in algorithmic form, feature detection on 3D data is generally not an easy job, especially for 3D noisy mesh data [11,20].

In our smoothing application, we only concentrate on point-type features and we put particular emphasis on efficiency of the detection method. We therefore adopt a fast and heuristic feature detection approach based on normal variation of adjacent triangles as the following. We first perform a Laplacian operator on the mesh surface, i.e., the position of every vertex is replaced by the average of the centers of the triangles that are adjacent to this vertex. Then a vertex of the original mesh is detected as a feature vertex if the variation of the normal of its corresponding vertex in the smoothed mesh with the average normal of its 1-ring neighbor triangles is larger than a specified tolerance. This simple approach is fast and effective for most meshes in our experiments. Fig. 4 shows the results of feature detection from two mesh models using this simple scheme, where the detected feature points on the meshes are shown in red color. The user is also allowed to specify the feature vertices on the mesh surface in our system.

3.6. Parameters

As we will see in the following section, using uniform weighting scheme in Eq.3 work well for smoothing mesh models with dense or nearly uniform sampling in our approach. But for low resolution mesh, there might be large tangential movement for some vertices on the mesh, see Fig. 3(b). Therefore, we use non-uniform weighting scheme, such as cotangent scheme [5,32], for low resolution mesh models, to obtain better smoothing results.

Our approach uses the same parameters μ and λ for all the feature constraints and barycenter constraints in most of our examples. We have investigated to see how the parameters affect the reconstructed mesh in some details. In many examples, if we only separate the vertices into two groups, i.e., feature vertices and non-feature vertices, and use same weights for all feature vertices in the approach, there might be artifacts in the smoothed meshes. Like in the reference [22], we use Gaussian-weighted center-surround evaluation of curvatures on meshes and measure the features in different size and importance. Thus different weights can be assigned to different vertices by their feature importance, see an example shown in Fig. 5.

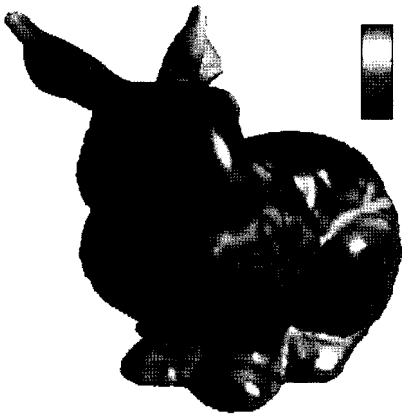


Fig. 5. Feature importance computed by Gaussian-weighted center-surround evaluation of curvatures for the vertices on the bunny model using the method in [22].

3.7. Discussions

The constraints described in the previous sections are soft constraints as the smoothed mesh only approximates the constraint points in a least squares sense but not interpolates.

We can as well put hard constraints on the mesh vertex positions in our system. That is, a set of vertices can be fixed so that the smoothing is performed only on the rest of the mesh. More complicated constraints are also possible by adding more appropriate equations into the linear system.

Furthermore, our algorithm can be easily applied locally. Our system also allows to smooth a part of a mesh less than another one, in order to keep desirable features while getting a smoother version, which happens to be useful in practice. .

4. Implementations

The matrix $\mathbf{A}^T\mathbf{A}$ or $\overline{\mathbf{A}}^T\overline{\mathbf{A}}$ are also sparse since in every row of the matrices only the entries corresponding to vertices in the 2-ring neighborhood are non-vanishing. The most time-consuming part of our algorithm is solving the sparse linear system. We use the direct solver in [34] in our implementation. The Cholesky factorization of the matrix $A^TA = R^TR$ is first found, where R is an upper triangular matrix. Then x, y, and z are respectively found by solving two triangular linear systems $R^TRX = A^Tb$, that is R^TX^* and $RX = X^*$. Most of the time is spent on computing the Cholesky factorization, while the time of the solving is negligible. As stated in [34], the factorization is fast enough for the applications.

5. Experiments and results

All the examples presented in this paper were made on a 1.8 GHz Athlon computer with 512 MB memory. Feature vertices are not shown in the mesh surface in order to compare the rendering effects of the smoothed mesh surface with the original mesh surface.

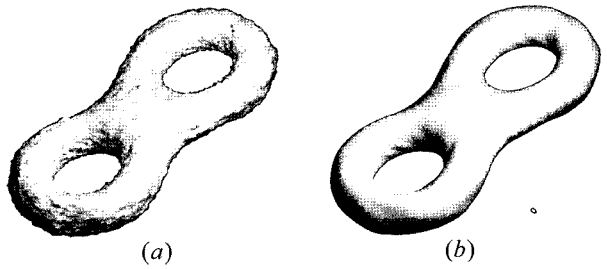
The weight parameters μ and λ measure the importance of feature constraints and barycenter constraints respectively. It is worthwhile to point out that the two parameters are important for the smooth effects. The smoothed mesh could almost interpolate the feature points when μ is large enough. The mesh could not be smoothed much if the parameter μ is set to be too large.

In Fig. 6(a), Gaussian white noise is added to an "8"-shape model. The constructed smoothed mesh is shown in 6(b) with parameters $\mu = 5$ and $\lambda = 0.1$.

Fig. 7 and 8 are two smoothing examples using our global smoothing approach for real scanned mesh data.

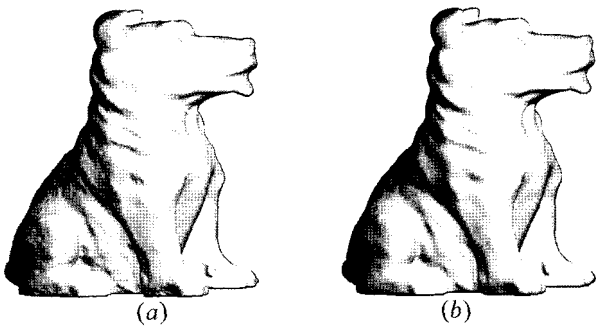
Fig. 9 shows an example for smoothing a noisy model with sharp features. Note that the sharp edge features are well preserved in this example.

It is worthwhile to point out that our smoothing approach is also applicable for noisy meshes with topological errors and non-manifold meshes. For the purpose of smoothing using our global Laplacian approach, it is not necessary to require that a surface mesh is manifold. The only requirement is that the neighborhoods of each vertex are about the same as



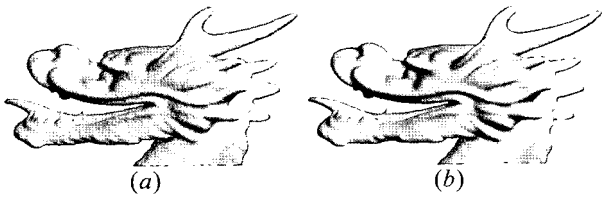
(a) (b)

Fig. 6. Global laplacian smoothing for an "8"-shape model. The models are flat shaded in order to enhance the noise effects: (a) the model of an "8" model with heavily added random moise; (b) the smoothed mesh with parameters $\mu = 5$ and $\lambda = 0.1$.



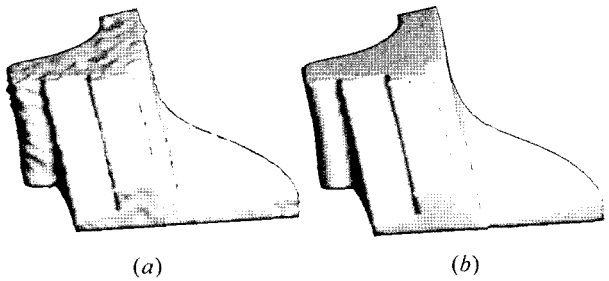
(a) (b)

Fig. 7. Denoising a scanned mesh of a dog model using our approach: (a) the noisy mesh data; (b) the smoothed mesh with parameters $\mu = 5$ and $\lambda = 0.1$.



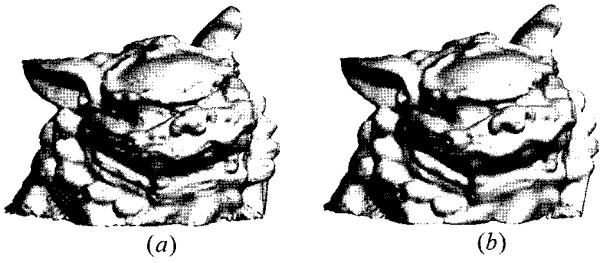
(a) (b)

Fig. 8. Denoising a scanned mesh of a dragon model using our approach: Note that features such as sharp corners are preserved: (a) the noisy mesh data; (b) the smoothed mesh with parameters $\mu = 5$ and $\lambda = 0.06$.



(a) (b)

Fig. 9. Denoising a noisy mesh with sharp features using our approach. (a) the noisy fandisk model; (b) the smoothed model with parameters $\lambda = 0.1$.



(a) (b)

Fig. 10. Denoising a scanned mesh of a lion-dog model using our global Laplacian smoothing approach. The mesh is non-manifold. Note that features such as sharp corners are preserved: (a) the noisy mesh data; (b) the smoothed mesh with parameters $\mu = 5$ and $\lambda = 0.01$.

they are on the underlying real surface. Our algorithm uses the neighbor information to perform the global Laplacian smoothing on the mesh surface. The Lion-Dog model shown in Fig. 10(a) is a real scanned noisy mesh model with variety of non-manifold structures. Fig.10(b) shows the smoothed mesh with parameters $\mu = 5$ and $\lambda = 0.1$.

A comparison to the other smoothing approaches is shown in Fig. 11. The model of Venus head shown in Fig. 11(a) is a real scanned mesh data with much noise. The smoothed mesh using traditional Laplacian smoothing method is shown in Fig. 11(b). We can see the shrinkage in the smoothed mesh. Fig. 11(c) is the result using the approach of [13]. The result using our global Laplacian smoothing approach is shown in Fig. 11(d) with parameters $\mu = 5$ and $\lambda = 00.5$. We can see that our result is as good as the result of [13], with better feature preservation in some regions, such as the eye and lip regions, in the Venus head model. We present this comparison to demonstrate the effectiveness and superiority of our approach for smoothing.

Table 1. Running time for different examples shown in the paper.

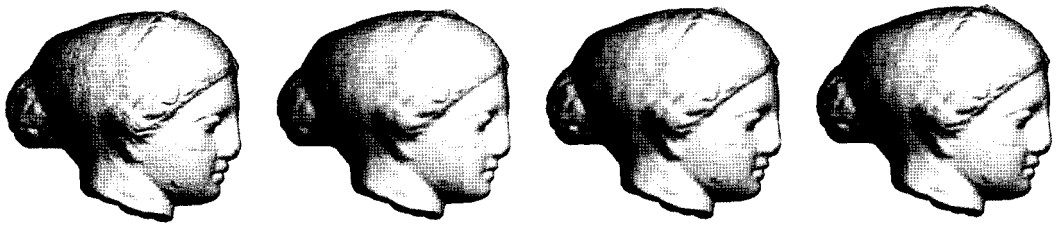
Model	Vertex number	Running time(s)
Eight (Fig. 6)	12,286	1.95
Dog (Fig. 7)	195,586	71.25
Dragon (Fig. 8)	100,056	22.75
Fandisk (Fig. 9)	6,475	0.96
Lion-Dog (Fig. 10)	24,930	2.79
Venus (Fig. 11)	134,359	42.0
Bunny (Fig. 12)	34,839	5.58

We can easily generate new surfaces with controllable smoothness from a noisy mesh surface by applying different parameters in our algorithm. Fig. 12 shows the results by applying our algorithm to the bunny model with different parameters λ.

Table 1 lists the running time of the global Laplacian smoothing examples shown in this paper. As we can see, our approach achieves a good combination of speed, smoothness quality, and shape and feature preservation.

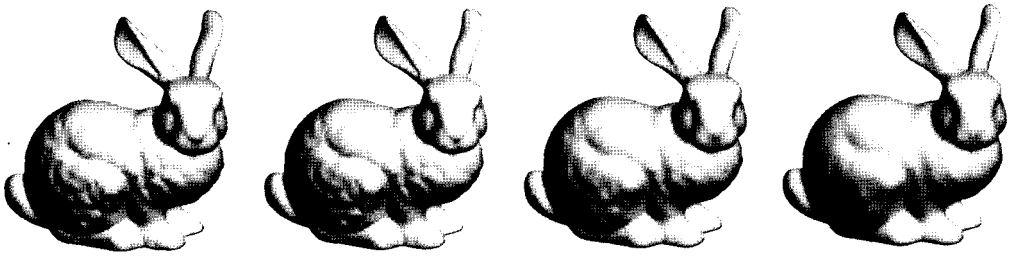
6. Conclusions

In this paper a novel technique for the elimination of



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

Fig. 11. Comparisons with other smoothing approches. Observe the difference in details in the area of the lip and eye; (a) the noisy Venus head mesh data; (b) the result denoised by the traditional laplacian smoothing approach; (c) the result using bilateral filtering approach in [13]; (d) the result of our global Laplacian smoothing approach with parameters $\mu = 5$ and $\lambda = 0.05$



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

Fig. 12. Smoothed surfaces with controllable smoothness using different choices of parameters in our system; (a) the noisy mesh; (b)-(d) the smoothed mesh surfaces using the parameters $\lambda = \lambda$ 0.3, 0.05, 0.01, respectively.

noise in arbitrary mesh surfaces has been developed and applied successfully. The Laplacian operator is performed over the mesh surface in a global way. The noisy mesh is smoothed by solving a sparse linear system. Feature constraints and barycenter constraints can be added into the linear system to keep the features of the noisy mesh and to avoid the shrinkage and distortion during the smoothing. The proposed approach presents certain advantages: it is non-iterative, fast and effective; it is also applicable for non-manifold meshes. We demonstrate the superiority of our approach over traditional approaches by many experimental results.

It is much worthwhile to extend our approach to other surface fairing energy instead of Laplacian operator. We believe that this extension is feasible but not straightforward.

Acknowledgements

The real scanned 3D data are courtesy of Stanford 3D Scanning Repository and Max-Planck Insitut für Informatik. This work is supported by the National Natural Science Foundation of China (No. 60503067, 60333010), Zhejiang Provincial Natural Science Foundation of China (No. Y105159) and the National Grand Fundamental Research 973 Program of China (No. 2002CB312101).

References

[1] Bajaj, C. and Xu, G. (2003), Anisotropic diffusion on surfaces and functions on surfaces, *ACM Transactions on Graphics*, **22**(1), 4-32.

[2] Canny, J. (1986), Computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(6).

[3] Clarenz, U., Diewald, U. and Rumpf, M. (2000), Anisotropic geometric diffusion in surface processing *Proceedings of IEEE Visualization*, 397-405.

[4] DeRose, T., Duchamp, T., Hoppe, H., McDonald, J. and Stuetzle, W. (1993), Mesh optimization, *Proceedings of SIGGRAPH*, 19-26.

[5] Desbrun, M., Meyer, M., Schröder, P. and Barr, A. (1999), Implicit fairing of irregular meshes using diffusion and curvature flow, *Proceedings of SIGGRAPH*, 317-324.

[6] Desbrun, M., Meyer, M., Schr?der, P. and Barr, A. (2000), Anisotropic feature-preserving denoising of height fields and bivariate data, *Proceedings of Graphics Interface*, 145-152.

[7] Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M. and Stuetzle, W. (1995), Multiresolution Analysis of Arbitrary Meshes, *Proceedings of SIGGRAPH*, 173-182.

[8] Fleishman, S., Drori, I. and Cohen-Or, D. (2003), Bilateral mesh denoising, *Proceedings of SIGGRAPH*, 950-953.

[9] Field, D.A. (1988), Laplacian smoothing and Delaunay triangulations, *Communications in Applied Numerical Methods*, **4**, 709-712.

[10] Gross, M. and Hubeli, A. (2000), Fairing of nonmanifolds for visualization, *Proceedings of IEEE Visualization*, 407-414.

[11] Hubeli, A., and Gross, M. (2001), Multiresolution feature extraction from unstructured meshes, *Proceedings of Visualization*.

[12] Hildebrandt, K. and Polthier, K. (2004), Anisotropic filtering of non-linear surface features. *Proceedings of Eurographics*, 2004.

[13] Jones, T., Durand, F. and Desbrun, M. (2003), Non-iterative, feature preserving mesh smoothing, *Proceedings of SIGGRAPH*, 943-949.

[14] Kobbelt, L., Botsch, M., Schwanecke, U. and Seidel, H. (2001), Feature sensitive surface extraction from volume data, Proceedings of SIGGRAPH, 2001.

[15] Kobbelt, L., Campagna, S., Vorsatz, J. and Seidel, H.P. (1998), Interactive multiresolution modeling on arbitrary meshes, *Proceedings of SIGGRAPH*, 105-114.

[16] Karni, Z. and Gotsman, C. (2000), Spectral compression of mesh geometry, *Proceedings of SIGGRAPH*, 279-286.

[17] Kobbelt, L. (1997), Discrete fairing, *Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces*, 101-131.

[18] Liu, X., Bao, H., Heng, P., Wong, T. and Peng, Q. (2001), Constrained fairing for meshes, *Computer Graphics Forum*, **20**(2), 115-123.

[19] Liu, X., Bao, H., Shum, H. and Peng, Q. (2002), A novel volume constrained smoothing method for meshes, *Graphical Models*, **64**, 169-182.

[20] Lee, Y. and Lee, S. (2002), Geometric snakes for triangle meshes, *Proceedings of Eurographics*.

[21] Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J. and Fulk, J. (2000), The Digital Michelangelo Project: 3D Scanning of Large Statues, *Proceedings of SIGGRAPH*, 131-144.

[22] Lee, C., Varshney, A. and Jacobs, D. (2005), Mesh Saliency, *Proceedings of SIGGRAPH*.

[23] Morton, H.P. and Sequin, C.H. (1992), Functional optimization for fair surface design, *Proceedings of SIGGRAPH*, 167-176.

[24] Ohtake, Y., Belyaev, A. and Bogaevski, I. (2000), Polyhedral surface smoothing with simultaneous mesh regularization, *Proceedings of Geometric Modeling and Processing*, 229-237.

[25] Ohtake, Y., Belyaev, A. and Seidel, H.P. (2002), Mesh smoothing by adaptive and anisotropic Gaussian filter, Vision, *Modeling and Visualization*, 203-210.

[26] Pauly, M., Keiser, R. and Gross, M. (2003), Multi-scale feature extraction on point-sampled surfaces, *Proceedings of Eurographics*, 2003.

[27] Perona, P. and Malik, J. (1990), Scale-space and edge detection using anisotropic diffusion, *IEEE PAMI*, **12**(7), 629-639.

[28] Peng, J., Strela, V. and Zorin, D. (2001), A simple algorithm for surface denoising, *Proceedings of IEEE Visualization*, pp. 107-112.

[29] Rusinkiewicz, S., Hall-Holt, O. and Levoy, M. (2002), Real-time 3D model acquisition, *ACM Transactions on Graphics*, **21**(3), 438-446.

[30] Rudin, L., Osher, S. and Fatemi, E. (1992), Nonlinear total variation based noise removal algorithms, *Physica D*, **60**, 259-268.

[31] Sorkine, O. and Cohen-Or, D. (2004), Least-squares meshes, *Proceedings of Shape Modeling International*, 191-199.

[32] Taubin, G. (1995), A signal processing approach to fair surface design. *Proceedings of SIGGRAPH*, 351-358.

[33] Taubin, G. (2001), Linear anisotropic mesh filtering, *IBM*

Research Technical Report, RC2213.

[34] Toledo, S. (Sept,2003), TAUCS: a library of sparse linear solvers, version 2.2. Tel-Aviv University, http://www.tau.ac.il/ stoledo/taucs/.

[35] Tasdizen, T., Whitaker, R., Burchard, P. and Osher, S. (2002), Geometric surface smoothing via anisotropic diffusion of normals, *Proceedings of IEEE Visualization*, 125-132.

[36] Vollmer, J., Mencl, R. and Müller, H. (1999), Improved Laplacian smoothing of noisy surface meshes, *Proceedings of Eurographics*, 131-138.

[37] Welch, W. and Witkin, A. (1992), Varational surface modeling, *Proceedings of SIGGRAPH*, 157-166.

Zhongping Ji was born in 1980 and received his B.A's degree in mathematics from the University of NorthWestern Poly-Technical University, Shanxi, China, in 2003. Since 2003, he has been a M.A's degree candidate in mathematics from Zhejiang University of China. His current research interests include digital geometry processing and computer graphics.

Guojin Wang is a professor and supervisor of doctoral student in the Department of Mathematics and the institute of Computer Images and Graphics, Zhejiang University, Hangzhou, China. He received his BSc and MSc in Mathematics from Zhejiang University. His teaching and research activities are concerned with applied mathematics, computer aided geometric design, computer graphics and geometric modeling. Between 1991 and 1993, he was a visiting researcher at Brigham Young University, USA. He also was a visiting researcher at Hong Kong University of Science and Technology, in 2002, 2003 and 2004. He has published more than 100 papers on CAGD and CG, and is first author of the book, CAGD, published by China Higher Education Press, Beijing and Springer-Verlag, Berlin, Heidelberg

Ligang Liu received the BS degree in mathematics in 1996 from Zhejiang University, China. In 2001, he received the PhD degree in mathematics, also from Zhejiang University. From 2001 to 2004, he was an associate researcher at the Internet Graphics Group, Microsoft Research Asia. Since 2004, he has been an associate professor in Department of Mathematics at Zhejiang University. His current research interests include geometric modeling and processing, interactive computer graphics, and image processing.

Zhongping Ji

Ligang Liu

Guojin Wang