A Novel Method for Surface Mesh Smoothing: Applications in Biomedical Modeling

Jun Wang and Zeyun Yu*

Department of Computer Science, University of Wisconsin-Milwaukee
{wang43,yuz}@uwm.edu

Abstract. In this paper, we present a surface-fitting based smoothing algorithm for discrete, general-purpose mesh models. The surface patch around a mesh vertex is defined in a local coordinate system and fitted with a quadratic polynomial function. An initial mesh smoothing is achieved by projecting each vertex onto the fitted surface. At each vertex of the initial mesh, the curvature is estimated and used to label the vertex as one of four types. The curvature-based vertex labeling, together with the curvature variation within a local region of a vertex, is utilized to adaptively smooth the mesh with fine features well preserved. Finally, three post-processing methods are adopted for mesh quality improvement. A number of realworld mesh models are tested to demonstrate the effectiveness and robustness of our approach.

Keywords: Surface mesh smoothing, Quadric surface fitting, Curvature labeling, Mesh quality improvement.

1 Introduction

The surface reconstruction and visualization from 3D imaging data have found wide applications in biomedical fields, such as computer-aided diagnosis, intervention planning, and realistic pathological analysis and prediction. The models used for such purposes are typically represented by surface meshes in triangular or quadrilateral forms. Going from imaging data to surface meshes involves a number of computational approaches. Two of the critical steps are image segmentation and surface mesh generation and processing. In digital images, image segmentation normally produces a vast number of discrete voxels (small cubes) that approximate the boundary of an object of interest. These voxels can certainly be converted into surface manifolds represented by quadrilateral meshes by extracting the faces of the cubes that face towards either outside or inside of the boundaries. But the resulting meshes suffer from an extremely bumpiness on the surface, as shown in the examples in Fig. 1. One of the goals in the present paper is hence to smooth a given mesh to reduce the bumpiness on the surface.

* Corresponding author.

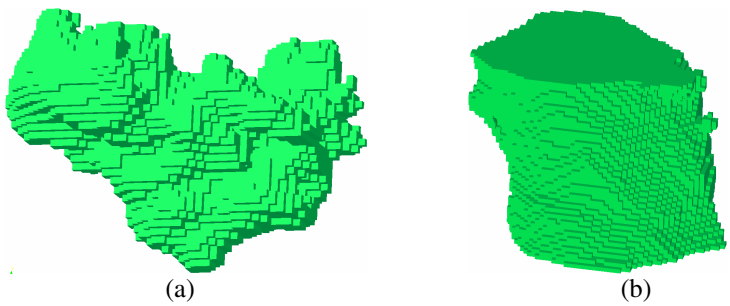


Fig. 1. Examples of surface meshes generated from image segmentation. The meshes appear very bumpy without further mesh processing.

Surface mesh smoothing has been studied for decades. Laplace iterative smoothing is one of the most common and simplest techniques for mesh smoothing [1-5]. During each iteration, each vertex of a mesh is adjusted to the barycenter of its neighboring region. It is very fast but often results in volume shrinkage. Taubin [2] used a signal processing method, the Laplacian operator, to fair surface design and reduce the shrinkage. Desbrun et al. [3] extended Taubin's work to smooth irregular meshes by using geometric flows. Their approach provides a better way to prevent the volume shrinkage. Peng [6] presented a denoising approach for geometric data represented as a semiregular mesh on the basis of locally adaptive Wiener filtering.

Other popular smoothing approaches include the energy minimization techniques. Kobblet [7, 8] proposed a general algorithm to fair a triangular mesh with arbitrary topology in R^3 by estimating the curvature for the discrete mesh model. Welch and Witkin [9] described an approach to designing fair, freeform shape by using triangulated surfaces. These methods are time-consuming due to the complicated energy functions that need to be solved.

Recently, feature-preserving smoothing methods [10-21] have drawn more and more attentions. Jones et al. [19] developed a feature-preserving smoothing algorithm by adopting statistics and local first-order predictors of triangulated surface meshes. Bajaj et al. [20] proposed a PDE-based anisotropic diffusion approach for processing noisy geometric surfaces and functions defined on surfaces. Li et al. [21] adopted the weighted bi-quadratic Bezier surface fitting and uniform principal curvature techniques to smooth surface meshes.

In the present paper, we describe a novel approach, combining the surface fitting, curvature estimation, and vertex labeling, to smooth meshes. Beside the surface smoothness that we shall demonstrate below, Antoher main goal in this paper is to preserve or improve the mesh quality so that the processed meshes have no sharp angles. The mesh quality is extremely critical in some applications such as the numerical simulation using finite or boundary element methods. However, not all meshes that appear smooth have high quality in terms of angles. For example, the marching cube algorithm can be used to generate a very smooth surface triangulation of a volumetric function but the resulting mesh usually suffers from too many sharp or skinny angles, which often cause poor accuracy in

numerical simulation. To this end, improving mesh quality, sometimes also referred to as *mesh smoothing*, is taken into serious consideration in our algorithms presented here.

While triangular surface meshes are perhaps the most common form in representing a 3D model and a majority of previous work has been focused on this type of meshes, we will consider in this paper quadrilateral meshes, especially those extracted from 3D imaging data and represented by cube-like bumpy meshes. The resulting meshes will be smoothed quadrilateral meshes with improved quality, which can be readily converted into triangular meshes if needed. In addition, the data structures we used can be easily modified to read triangular meshes as inputs. In other words, the approach we propose here will provide a general-purpose tool for mesh processing in a variety of applications.

2 Surface Mesh Smoothing Algorithm

Our algorithm includes four steps. At first, the geometric and topological relationship is constructed from the input surface mesh model. For each vertex, we construct a local coordinate system and then fit the neighboring vertices in the local coordinate system with an analytical quadric surface function. The vertex being considered is projected onto this quadric surface. The initial mesh smoothing is achieved by updating each vertex with its projection. Then, the curvature of each vertex is estimated based on the first and second fundamental forms of its quadric surface obtained and then the vertex is labeled with the corresponding type. Additionally, for each vertex, the size of its neighborhood is dynamically determined according to the curvature and vertex type so that the mesh is adaptively smoothed by similar approaches as in the first step. Finally, the mesh is further improved with a few post-processing algorithms.

2.1 Initial Mesh Smoothing

The local surface patch around a point can be approximated with a quadric surface [22]: $S(u, v)=(u, v, h(u, v))$, a parametric representation in a local coordinate system as shown in Fig 2, where p is the origin; h-axis directs along the normal vector n of p on S; and u-, v- axes are orthogonal vectors in the tangent plane of p on S. Obviously, the local coordinate system (p-uvh) can be transformed from the global coordinate system (o-xyz). According to the surface theory [23], the local shape of surface around p can be represented with Darboux system $D(p) = (p, T_1, T_2, n, k_1, k_2)$, where (T_1, T_2), (k_1, k_2) are principal directions and curvatures, respectively.

We extend this principle to discrete mesh models and construct the local coordinated system for each vertex on the meshes. Since the construction of a local system mainly relies on the h-axis, i.e., the normal vector of the vertex, we first estimate the normal vector for each vertex with the area-weighted averaging method.

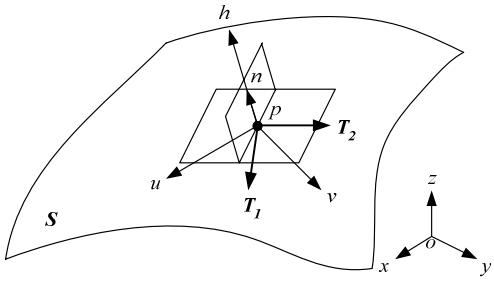


Fig. 2. An illustration of the local coordinate system on a surface

Let $NV_i(k)$ be the set of neighboring vertices of vertex v_i within the ***k*-ring** (that is, the minimum number of edges from v_i to the neighboring vertices is equal to or less than ***k***), $E_i = \{<v_i, v_j> | v_j \in NV_i\ (1)\}$ the set of edges incident to v_i, and F_i the set of faces incident to v_i. Then the normal vector n_i of v_i is calculated as:

$$n_i = \frac{1}{N}\sum_{j=1}^{N}\left(area_{f_j} \cdot n_{f_j}\right), \quad f_j \in F_i. \tag{1}$$

where $area_f$, n_f are respectively the area and normal vector of face f in F_i. After the normal vector n_i is calculated, we consider v_i, n_i as the origin and z-axis of the local coordinate system respectively. Then the x- and y- axes of the local coordinate system are arbitrarily chosen in the plane locating at v_i that is orthogonal to the normal n_i.

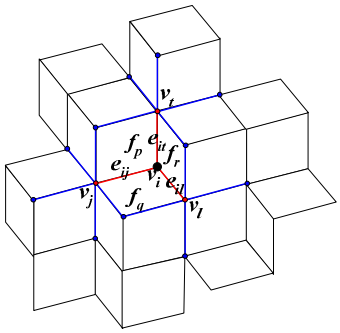


Fig. 3. The 2-ring neighboring vertices at a vertex. The minimal number of edges from these vertices to the center vertex is equal to or less than 2.

After constructing the local coordinate system, we find all the neighboring vertices for each vertex with a ring-by-ring scheme, referred to as the *k-ring neighborhood*. At the initial mesh smoothing phase, k is fixed as 2 in our algorithm. Fig. 3 shows the 2-ring neighboring vertices of v_i, in which the red and blue

vertices are the first and second ring neighboring vertices respectively, and all these vertices constitute the 2-ring neighbors of v_i.

For each vertex, the global coordinates of all the neighboring vertices are transformed to the local coordinates by homogeneous transformation and then the local coordinates are used to fit an analytical quadric surface:

$$h(u, v) = au^2 + buv + cv^2 + eu + fv + g \tag{2}$$

In our algorithm, the least square fitting method is adopted. Let $V_i = \{<x_j, y_j, z_j> | j = 1, 2, \ldots m\}$ be the local coordinates of 2-ring neighboring vertices of vertex v_i. The objective function of the least square quadric surface fitting can be expressed as:

$$F = \sum_{j=1}^{m} \left| \begin{pmatrix} x_j \\ y_j \\ ax_j^{\,2} + bx_j y_j + cy_j^{\,2} + ex_j + fy_j + g \end{pmatrix} - \begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix} \right|^2 \cdot w_j \tag{3}$$

where w_j is the weighting function defined below. It is observed that the closer the neighboring vertex to v_i, the more it affects the surface shape. Therefore, w_j can be represented as

$$w_j = e^{-\|v_j - v_i\| / \max(\|v_1 - v_i\|, \ldots \|v_m - v_i\|)} \tag{4}$$

Minimizing F gives rise to $\frac{\partial F}{\partial a} = 0$, $\frac{\partial F}{\partial b} = 0$, $\frac{\partial F}{\partial c} = 0$, $\frac{\partial F}{\partial e} = 0$, $\frac{\partial F}{\partial f} = 0$, and $\frac{\partial F}{\partial g} = 0$, that is,

$$\left(\sum_{j=1}^{m} w_j \begin{bmatrix} x_j^2 \\ x_j y_j \\ j_j^2 \\ x_j \\ y_j \\ 1 \end{bmatrix} \begin{bmatrix} x_j^2 \\ x_j y_j \\ j_j^2 \\ x_j \\ y_j \\ 1 \end{bmatrix}^T \right) \begin{bmatrix} a \\ b \\ c \\ e \\ f \\ g \end{bmatrix} = \sum_{j=1}^{m} w_j z_j \begin{bmatrix} x_j^2 \\ x_j y_j \\ j_j^2 \\ x_j \\ y_j \\ 1 \end{bmatrix} \tag{5}$$

The linear equations can be solved with Gaussian elimination method [24], which yields the coefficients of the quadric surface. The vertex v_i is then projected onto the fitted quadric surface along its normal vector. Since its normal vector is exactly the z-axis in the local coordinate system, the coordinate of the projected point of v_i should be (0, 0, g) in the local coordinate system. Its global coordinate is calculated by the coordinate transformation from local system to global system. Finally, the vertex v_i is updated with the global coordinate of its projection. The above steps are performed for every vertex, yielding what we call the *initial mesh smoothing*. Fig. 4 shows mesh smoothing of two simple models.

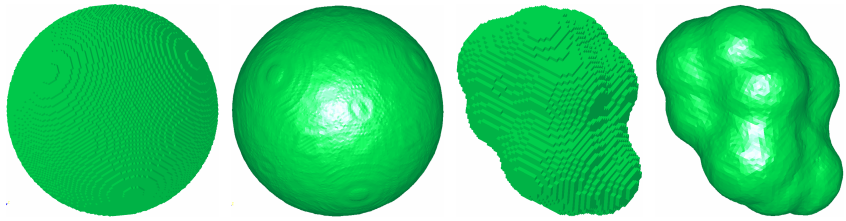


Fig. 4. Two examples of initial mesh smoothing. The first and third from left are the original models and the other two are meshes after smoothing.

2.2 Curvature Estimation and Labeling

Meshes generated in the initial smoothing step are much better than the original meshes but may not be good enough for subsequent applications if the original meshes are too bumpy. In our algorithm, an adaptive mesh smoothing algorithm is proposed to further smooth the meshes by taking advantage of the same surface fitting technique as seen in Section 2.1 but the size of the neighborhood considered adaptively changes according to the local geometric properties as explained below.

Many features in the real-world structures/models are usually defined on the basis of curvature variance of the surface meshes. When an input mesh, such as the cube-based meshes in Fig. 1, is too coarse, we cannot accurately calculate the curvature information. However, this becomes much less problematic after the initial mesh smoothing. To further smooth a mesh and preserve the features in the model, the same surface fitting and vertex projection schemes as in initial Section 2.1 are adopted, whereas the ring number of neighboring vertices for surface fitting is dynamically determined based on the curvature variance. In this subsection we introduce how the curvatures are analytically calculated and used to label the vertices.

In the initial mesh smoothing, each vertex is associated with a quadric surface using the least squre fitting technique. Since each vertex has been adjusted to its projection in the normal direction, it is straightforward to calculate the curvature of the vertex on quadric surface by using the first and second fundamental forms of a surface. The analytical form of the quadric surface of a vertex v_i can be transformed to the parametric form as:

$$S = (u, v, h(u, v)) \tag{6}$$

The coefficients of the first and second fundamental forms of the quadric surface at v_i are calculated as:

$$\begin{cases} E = 1 + e^2 \\ F = ef \\ G = 1 + f^2 \end{cases}, \text{ and } \begin{cases} L = 2a \Big/ \sqrt{e^2 + f^2 + 1} \\ M = b \Big/ \sqrt{e^2 + f^2 + 1} \\ N = 2c \Big/ \sqrt{e^2 + f^2 + 1} \end{cases} \tag{7}$$

The Gaussian and mean curvatures at v_i are then given respectively as follows:

$$K_{v_i} = \frac{LN - M^2}{2(EG - F^2)} = \frac{4ac - b^2}{(e^2 + f^2 + 1)^2} \tag{8}$$

$$H_{v_i} = \frac{EN - 2FM + GL}{2(EG - F^2)} = \frac{c + ce^2 + a + af^2 - bef}{(e^2 + f^2 + 1)^{3/2}} \tag{9}$$

The Gaussian and mean curvatures of a mesh face $face_i$ with vertices v_1, v_2, …, v_m are approximated as:

$$K_{face_i} = \frac{1}{m}\sum_{j=1}^{m} K_{v_j} \quad \text{and} \quad H_{face_i} = \frac{1}{m}\sum_{j=1}^{m} H_{v_j} \tag{10}$$

After the Gaussian (K) and mean (H) curvatures are obtained, all the mesh vertices and faces are labeled with the corresponding type. In particular, the signs of K and H define the surface type and the values of K and H define the surface sharpness. Besl and Jain [25] proposed eight fundamental surface types using the signs of K and H as shown in Table 1.

Table 1. Eight fundamental surface types

	$K > 0$	$K = 0$	$K < 0$
$H < 0$	Peak	Ridge	Saddle ridge
$H = 0$	N/A	Flat	Minimal surface
$H > 0$	Pit	Valley	Saddle valley

In our algorithm, four simplified types are considered: convex, flatten, minimal and concave. In Table 1, the convex type includes peak, ridge and saddle ridge surfaces; the flatten and minimal types only includes flat and minimal surfaces respectively; and the concave type contains pit, valley and saddle valley surfaces. These surface types are extended to mesh vertices. After the curvatures are calculated, the faces and vertices are labeled with the criterion in Table 2.

Table 2. Four surface types

Surface/Vertex type	Signs of H & K
Convex type	H < 0
Flatten type	H = 0
Minimal type	H = 0 && K < 0
Concave type	H > 0

2.3 Adaptive Mesh Smoothing

After curvature-based vertrex labeling, the whole mesh is segmented into four types of features, each of which consists of topologically adjacent vertices and

faces with the same surface type. In order to further smooth the mesh model, the quadric surface fitting on each vertex is performed in a similar way as in Section 2.1, and the vertex is adjusted by projecting onto the fitted surface. Meanwhile, to preserve the features of the mesh model during the smoothing process, the neighboring vertices used in the surface fitting here should have the same type as that of the center vertex. As a result, these four types of features will be preserved after smoothing.

However, there are cases in which more than one feature may be seen in a segment having the same surface type. Consequently, if the vertex type was considered as the only criterion for definging the neighborhood of the vertex in the quadric surface fitting, many sub-features would be filtered. To address this issue, the standard deviation of curvatures within the neighborhood chosen is used as an additional constraint to determine the neighborhood size to preserve the sub-features in a mesh. The standard deviation of curvatures in a neighborhood of a vertex should be bounded by a pre-defined threshold.

Below we give the detailed algorithm of adaptive mesh smoothing. Let $V = \{v_1, v_2, \ldots v_n\}$ be the set of mesh vertices after initial mesh smoothing; $T = \{t_1, t_2, \ldots t_n\}$ the types of vertices; k the fixed ring number in the initial mesh smoothing and *tmpVerArray* the temporary vertex array. k is set as 2 by default. For each vertex v_i,

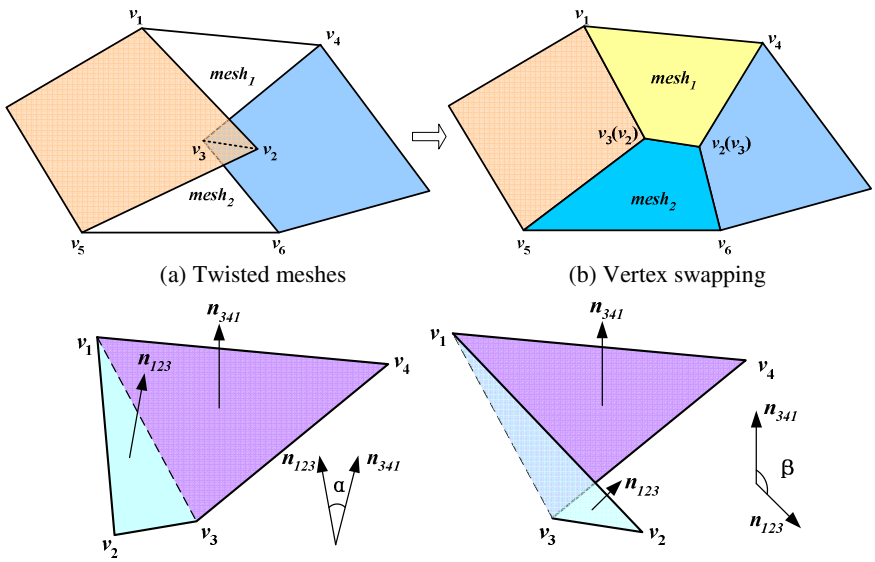
Step 1. Search its k-ring neighboring vertices and calculate the standard deviation σ_k of curvatures of all such vertices;
Step 2. Set *tmpVerArray* = NULL. For each vertex v_i in the set of the outermost (k-th) ring neighboring vertices,
i) if the types t_i and t_j of v_i and v_j are same, add the v_j into *tmpVerArray*;
ii) otherwise, go to step 4.
Step 3. Calculate the standard deviation σ_{k+1} of curvatures of k-ring neighboring vertices together with vertices in *tmpVerArray*.
i) if σ_{k+1} is less than σ_k, add each vertex of *tmpVerArray* into the neighboring vertices and update $k \leftarrow k+1$, $\sigma_k \leftarrow \sigma_{k+1}$; then go to step 2;
ii) otherwise, go to step 4.
Step 4. Fit the k-ring neighboring vertices of v_i with a quadric surface and project v_i onto the surface to get the vertex v_i';
Step 5. Adjust v_i to v_i', i.e. $v_i \leftarrow v_i'$.

During the smoothing processes, we adjust mesh vertices by updating their coordinates, while the geometric and topological relationship of the mesh model remains unchanged, which reduces the space and time complexity of this smoothing algorithm.

2.4 Mesh Quality Improvement

When the input mesh has complicated features and the resolution of the mesh is not high enough to capture the features, the smoothing approaches described above may produce some unwanted errors or low-quality meshes, such as twisted polygons, very short edges, or very sharp angles. Below we briefly describe the strategies to handle these three cases.

- ***Twisted meshes.*** Normally, the four vertices in a quadrilateral should be arranged in order (e.g., counterclockwise), while in a twisted quadrilateral, the vertices are arranged in a "Z" shape. Fig. 5(a) shows the twisted meshes, $mesh_1$, $mesh_2$ and Fig. 5(b) gives the corresponding meshes after adjustment. In order to detect whether a quadrilateral is twisted or not, we use the normal vectors of the two triangles associated with the quadrilateral. In a regular quadrilateral the angle between the normal vectors n_{123}, n_{341} of the triangles $v_1v_2v_3$, $v_3v_4v_1$ is less than $90°$, as shown in Fig. 5(c). By contrast, the angle would be more than $90°$ in a twisted quadrilateral (see Fig. 5(d)). A twisted quadrilateral usually occurs in conjunction with its opposite quadrilateral sharing the same edge. Therefore, the common edge (v_3v_2 in Fig. 5(a)) between a pair of twisted quadrilaterals can be easily identified and its end points are swapped to cure this problem.



(a) Twisted meshes (b) Vertex swapping

(c) Vertex arrangement in regular mesh (d) Vertex arrangement in twisted mesh

Fig. 5. Illustration of handling twisted meshes

- ***Meshes with short edges.*** This occurs when the length of one edge of a quadrilateral is shorter than a pre-defined threshold (for example, see v_1v_2 in Fig. 6). The steps to correct this type of meshes are described as follows. For the end point v_1 of a short edge v_1v_2,,
  - Find the two associated edges v_1v_3, v_1v_4 in the meshes, $mesh_1$, $mesh_2$, containing v_1v_2,
  - Compute the bisector c_1 of $\angle v_3v_1v_4$ and a plane pln passing through c_1 and the vector $\overrightarrow{v_1v_3} \times \overrightarrow{v_1v_4}$;

– Find the intersection point p_1 between pln and the edges of all the polygons incident to v_1;

For the other end point v_2, its intersection point p_2 can be computed in the same way. Then, the trisected points v'_1, v'_2 are determined on the polyline $\overset{\frown}{p_1 v_1 v_2 p_2}$ such that the lengths of $p_1 v'_1$, $v'_1 v_1 v_2 v'_2$ and $v'_2 p_2$ are equal. Finally, v_1, v_2 are updated with v'_1, v'_2 such that the short edge is prolonged (Fig. 6).

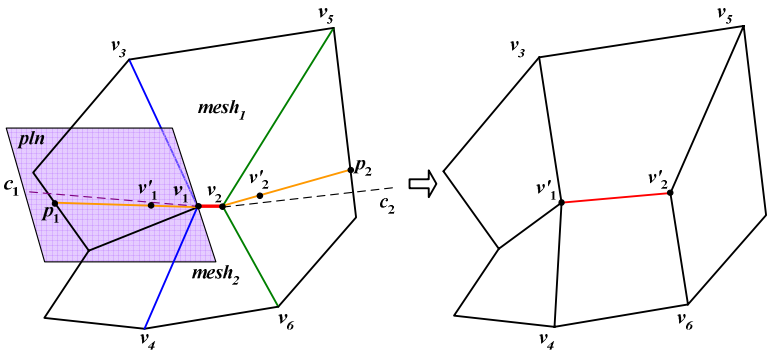


Fig. 6. Illustration of handling short edges

- ***Meshes with shape angles.*** When an internal angle of a mesh is smaller than a pre-defined threshold, the mesh needs to be improved. To this end, we adopt the angle-based approach as described in [26]. For the vertex v_1 with sharp angle, we consider the surrounding incident vertices and calculate the bisector of each of the angles formed by adjacent vertices on that ring. Then v_1 is projected onto each bisector (see the blue points in Fig. 7). Finally, the centroid of all these projection points is calculated and used to update v_1 so that the sharp angle can be improved (Fig. 7).

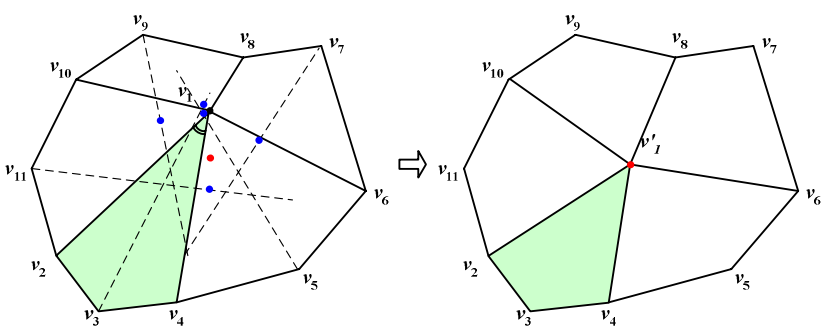


Fig. 7. Illustration of handling sharp angles

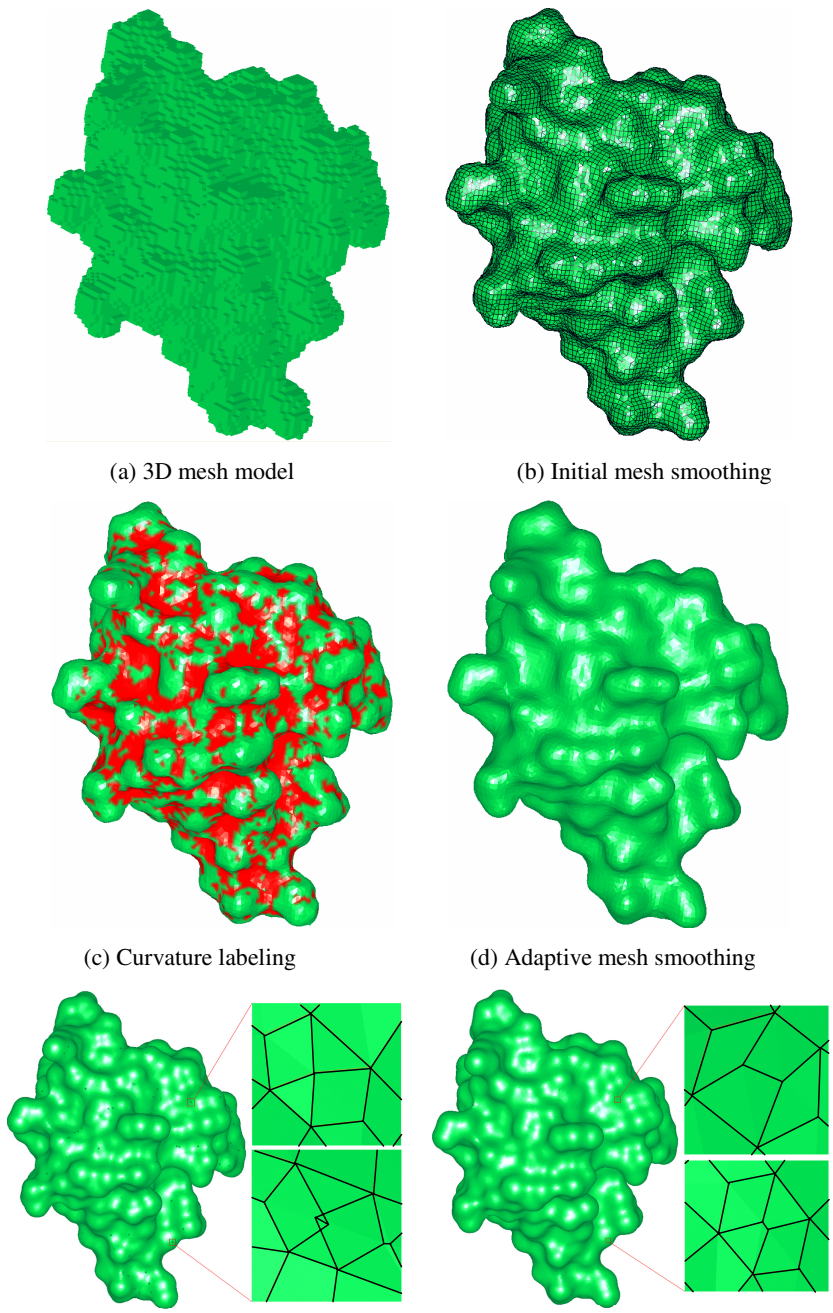
3 Implementation and Results

All algorithms described have been implemented in Visual C++ and OpenGL, running on a Pentium IV PC configuration with 3.0G Hz. Many 3D surface models have been tested and a couple of them are demonstrated here.

Fig. 8 shows the smoothing result of the 2CMP molecule, randomly chosen from the Protein Data Bank (http://www.rcsb.org/). The original mesh in Fig. 8(a) was generated using the approach described in [27]. The initial mesh smoothing gives the result as shown in Fig. 8(b). The curvature labeling is performed and shown in Fig. 8(c), where the patches in green and red are convex and concave surfaces respectively. A smoother mesh is achieved by the adaptive mesh smoothing, as seen in Fig. 8(d). However, there are still some defective polygons in the smoothed mesh, as shown with small dark dots in Fig. 8(e). The meshes after the quality improvement are shown in Fig. 8(f). To estimate the smoothness of a surface mesh, we calculate the curvature variation at each vertex as the maximal difference between the curvature of the vertex and the curvatures of the surrounding vertices. The average curvature variations over the entire mesh of 2CMP are 0.5032, 0.3529, and 0.0921 in the original mesh, the mesh after initial smoothing and the adaptively smoothed mesh respectively, suggesting that the proposed adaptive mesh smoothing method is effective in smoothing a surface mesh. Fig. 9 gives the result of another molecule called 2HAO, taken again from the Protein Data Bank. There are 72098 and 228132 vertices in the 2CMP and 2HAO models, and the computational time is 31.51s and 92.24s respectively.

As mentioned in the introduction, the marching cube method is able to produce smooth surface meshes, but many skinny triangular meshes are generated as well. For instance, the angle histograms of several molecular surface meshes generated by the marching cube method are shown in [28], where one can see a lot of very small (near 0^0) and large angles (near 180^0). To demonstrate the quality of the meshes , the angle histograms of the meshes generated with our method are plotted in Fig. 10, where the angles used are the internals angles of all the triangles obtained by dividing each quadrilateral into two triangles. It is interesting to observe that there are two obvious peaks in the histogram: one around 60° that corresponds to equilateral triangles and the other around 90° indicating that a large number of regular (equilateral) quadrilateral have been kept after the mesh smoothing.

Fig. 11 demonstrates the mesh smoothing results from 3D imaging data. Fig. 11(a) shows a cross section of the 3D cryo-electron microscopy reconstruction of the rice dwarf virus. The initial surface mesh, shown in Fig. 1(a), was generated using the 3D image segmentation algorithms [29]. Fig. 11(c) shows the surface mesh after applying the smoothing techniques as described in Section 2. Fig. 11(b) shows a cross section of the 3D electron tomographic reconstruction of the ventricle muscle cell. The initial surface mesh, illustrated in Fig. 1(b), was extracted using automatic image segmentation methods [30]. Fig. 11(d) shows the mesh after our smoothing algorithms. From both examples, we can see that our mesh smoothing approach can be used in conjunction with image segmentation for a variety of 3D biomedical imaging data.

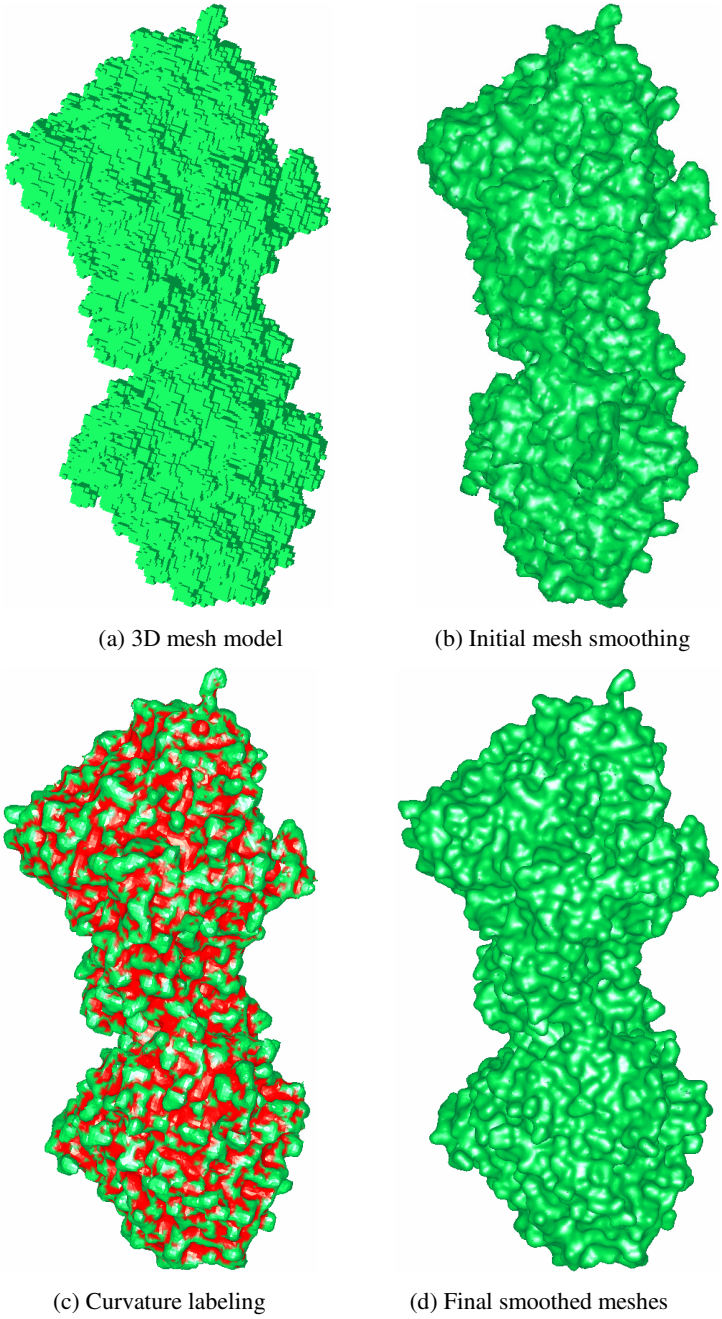


(a) 3D mesh model (b) Initial mesh smoothing

(c) Curvature labeling (d) Adaptive mesh smoothing

(e) Defective meshes after mesh smoothing (f) Final smoothed meshes

Fig. 8. Example of mesh smoothing for the molecular model 2CMP



(a) 3D mesh model (b) Initial mesh smoothing

(c) Curvature labeling (d) Final smoothed meshes

Fig. 9. Mesh smoothing of the molecular model 2HAO

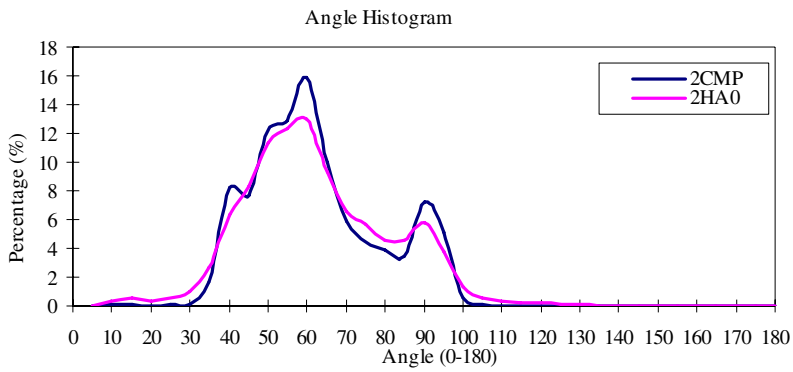
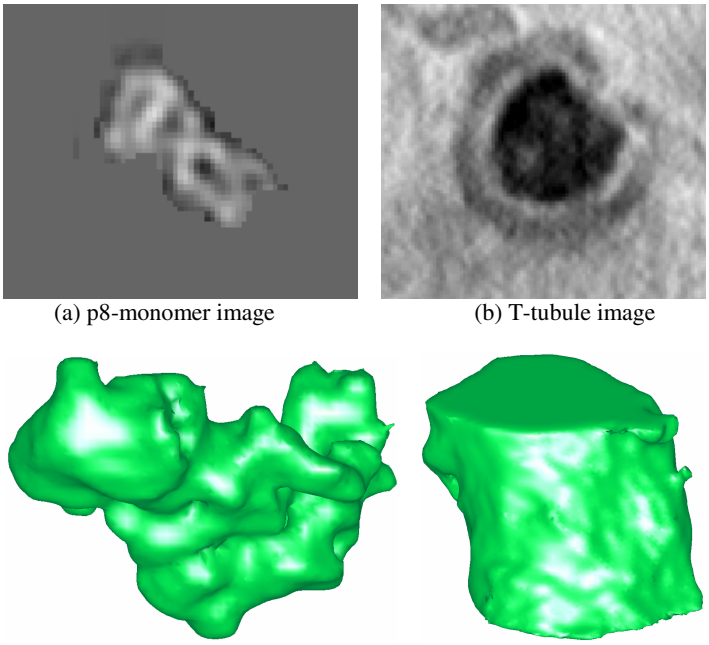


Fig. 10. Mesh angle histogram of models: 2CMP and 2HAO



(a) p8-monomer image (b) T-tubule image

(c) Final smoothed meshes (d) Final smoothed meshes

Fig. 11. Mesh smoothing of surface models extracted from 3D imaging data. (a) & (c): a cryo-electron microscopy reconstruction of the rice dwarf virus (courtesy of Dr. Wah Chiu, Baylor College of Medicine). (b) & (d): an electron tomographic reconstruction of ventricular cells (courtesy of Dr. Masahiko Hoshijima, UC-San Diego).

4 Conclusions

We have presented in this paper a novel mesh smoothing method based on quadric surface fitting for general mesh models. The fitting algorithm is combined with

vertex projection, curvature estimation, and mesh labeling of four types. Based on the geometric characteristics of surfaces, the adaptive mesh smoothing is conducted to further smooth meshes while important features are well preserved. To improve the mesh quality, three adjustment methods are adopted to handle defective meshes.

Our mesh smoothing algorithm can handle a variety of meshes, including molecular models, imaging data and industrial surface meshes. As demonstrated in the results, our method can generate meshes with high quality, i.e., no sharp angle or short edge in the output meshes. This is particularly useful and sometimes necessary in such applications as surface reconstruction, visualization, and numerical simulation. While the examples demonstrated here are all quadrilateral meshes, our method can be easily modified to handle triangular or other types of surface meshes.

Due to the quadric model being used to approximate the local shape of a freeform surface, there may be some cases where our method does not work perfectly. Such cases include very sharp and thin features and meshes with too low resolutions to capture the features of interest. One of our future efforts is to detect these ill-posed regions and apply mesh subdivision techniques to increase the mesh resolution so that our method can work more effectively in these circumstances. There are still a few small angles persisting in the final meshes as can be seen in Fig. 10. We shall work on better methods to handle these cases during the vertex projection step. While sharp edges or corners are not commonly present in the biomedical examples we have tested, these features will be taken care of in our future work by making a good balance between sharp features and possible surface noises.

References

1. Field, D.A.: Laplacian smoothing and Delaunay triangulations. Communications in Applied Numerical Methods 4, 709–712 (1988)
2. Taubin, G.: A Signal Processing Approach to Fair Surface Design. In: Proceedings of SIGGRAPH 1995, pp. 351–358 (1995)
3. Desbrun, M., Meyer, M., Schröder, P., Barr, A.H.: Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow. In: Proceedings of SIGGRAPH 1999, pp. 317–324 (1999)
4. Vollmer, J., Mencl, R., Müller, H.: Improved Laplacian smoothing of noisy surface meshes. In: Proceedings of Eurographics, pp. 131–138 (1999)
5. Ohtake, Y., Belyaev, A., Bogaeski, I.: Polyhedral Surface Smoothing with Simultaneous Mesh Regularization. In: Geometric Modeling and Processing, pp. 229–237 (2000)
6. Peng, J., Strela, V., Zorin, D.: A Simple Algorithm for Surface Denoising. In: Proceedings of IEEE Visualization 2001, pp. 107–112 (2001)
7. Kobbelt, L.: Discrete fairing. In: Proceedings of the 7th IMA Conference on the Mathematics of Surfaces, pp. 101–131. Springer, Cirencester (1996)
8. Kobbelt, L., Botsch, M., Schwanecke, U., Seidel, H.: Feature sensitive surface extraction from volume data. In: Proceedings of SIGGRAPH 2001 (2001)
9. Welch, W., Witkin, A.: Free-form shape design using triangulated surfaces. In: Proceedings of SIGGRAPH 1994, pp. 247–256. ACM Press, Orlando (1994)

10. Desbrun, M., Meyer, M., Schröder, P., Barr, A.H.: Anisotropic Feature-Preserving Denoising of Height Fields and Bivariate Data. In: Graphics Interface, pp. 145–152 (2000)
11. Taubin, G.: Linear anisotropic mesh filtering. IBM Research Technical Report. RC2213 (2001)
12. Liu, X., Bao, H., Heng, P., Wong, T., Peng, Q.: Constrained fairing for meshes. Computer Graphics Forum 20(2), 115–123 (2001)
13. Liu, X., Bao, H., Shum, H., Peng, Q.: A novel volume constrained smoothing method for meshes. Graphical Models 64, 169–182 (2002)
14. Tasdizen, T., Whitaker, R., Burchard, P., Osher, S.: Geometric surface smoothing via anisotropic diffusion of normals. In: Proceedings of IEEE Visualization, pp. 125–132 (2002)
15. Ohtake, Y., Belyaev, A., Seidel, H.-P.: Mesh Smoothing by Adaptive and Anisotropic Gaussian Filter. In: Vision, Modeling and Visualization, pp. 203–210 (2002)
16. Fleishman, S., Drori, I., Cohen-Or, D.: Bilateral Mesh Denoising. ACM Trans. Gr. (2003)
17. Zhang, H., Fiume, E.L.: Mesh Smoothing with Shape or Feature Preservation. In: Advances in Modeling, Animation, and Rendering, pp. 167–182 (2002)
18. Clarenz, U., Diewald, U., Rumpf, M.: Anisotropic geometric diffusion in surface processing. In: IEEE Visualization 2000, pp. 397–405 (2000)
19. Bajaj, C., Xu, G.: Anisotropic Diffusion on Surfaces and Functions on Surfaces. ACM Trans. Gr. 22(1), 4–32 (2003)
20. Jones, T., Durand, F., Desbrun, M.: Non-iterative, feature-preserving mesh smoothing. In: Proceedings of SIGGRAPH 2003, pp. 943–949. ACM Press, San Diego (2003)
21. Li, Z., Ma, L., Jin, X., Zheng, Z.: A new feature-preserving mesh-smoothing algorithm. Visual Comput. 25, 139–148 (2009)
22. Frey, P.J.: About surface remeshing. In: Proc. in 9th IMR, New-Orleans, pp. 123–136 (2000)
23. Milroy, M.J., Bradley, C., Vickers, G.W.: Segmentation of a wrap-around model using an active contour. Computer Aided Designed 29(4), 299–320 (1997)
24. Atkinson, K.A.: An Introduction to Numerical Analysis, 2nd edn. John Wiley & Sons, New York (1989)
25. Besl, P.J., Jain, R.: Segmentation through Variable-Order Surface Fitting. In: IEEE PAMI 1988, vol. 10(2), pp. 167–192 (1988)
26. Zhou, T., Shimada, K.: An angle-based approach to two-dimensional mesh smoothing. In: Proc. in 9th IMR, New-Orleans, pp. 373–384 (2000)
27. Yu, Z.: A list-based method for fast generation of molecular surfaces. In: The 31st International Conference of IEEE Engineering in Medicine and Biology Society (accepted, 2009)
28. Yu, Z., Holst, M., Cheng, Y., McCammon, J.A.: Feature-Preserving Adaptive Mesh Generation for Molecular Shape Modeling and Simulation. Journal of Molecular Graphics and Modeling 26(8), 1370–1380 (2008)
29. Yu, Z., Bajaj, C.L.: Computational approaches for automatic structural analysis of large bio-molecular complexes. IEEE/ACM Transactions on Computational Biology and Bioinformatics 5(4), 568–582 (2008)
30. Yu, Z., Holst, M., Hayashi, T., Bajaj, C.L., Ellisman, M.H., McCammon, J.A., Hoshijima, M.: Three-dimensional geometric modeling of membrane-bound organelles in ventricular myocytes: Bridging the gap between microscopic imaging and mathematical simulation. Journal of Structural Biology 164(3), 304–313 (2008)