Noise Removal Algorithm for Polygonal Meshes*

Liu Xinguo Hujun Bao Qunsheng Peng

State Key Lab of CAD&CG, Zhejiang University, 310027, P. R China

Abstract In this paper, we present an effective unshrinkage noise removal algorithm to generate smooth geometric models. By introducing the constraints of keeping all triangle barycenters invariant at each smoothing step, the algorithm converts the mesh fairing issue into the minimization of the constrained energy minimization functional, which is then solved efficiently by our iterative method. The experimental results demonstrate that our algorithm can not only quickly remove the noise with good shape preservation, but also be implemented efficiently and robustly.

Keywords: Polygonal mesh, energy functional, Laplacian operator, shrinkage and distortion

Polygonal meshes have been widely applied in computer aided design and computer graphics community due to their simplicity and powerful ability to model complex objects. The development of 3D scanning technology enables us to conveniently capture detailed and dense sample points of object surface. These samples are then converted to triangular mesh for later use [4,7,11]. However, the high-fidelity reconstructed models are usually noisy, and need pre-smoothing before used in other applications.

In surface fairing, the most popular way is to minimize a energy functional[7,8] defined over the surface. For a parametric surface $S: X = X(u,v)$, the following curvature energy is generally considered:

$$E(S) = \int_S \left(k_1^2 + k_2^2\right) \mathrm{d}S$$

where k_1 and k_2 are the principle curvature functions of surface *S*. Since to evaluate the principle curvatures is very difficult, in practice we usually use the membrane energy and thin-plate energy

$$E_{memb}(S) = \frac{1}{2}\int_S \left(X_u^2 + X_v^2\right) du\, dv$$

* Supported by National Excellent Youth Foundation (No.69925204)and National Science Foundation(No.69673027) of China

$$E_{thin}(S)=\frac{1}{2}\int_S \left(X_{uu}^2+2X_{uv}^2+X_{vv}^2\right)du\,dv$$

where $X_u, X_v, X_{uu}, X_{vv}, X_{uv}$ are the first and second derivatives of the surface. Nevertheless, to minimize the simplified energy functional is still very complex and time-consuming, and the results heavily depend on the parameterization of the surface.

In order to smooth a mesh with large data set, Taubin[5] generalized the classical Fourier analysis to discrete meshes, in which the discrete mesh fairing issue can be regarded as a low filtering of surface signal by linearly approximating the Laplacian operator. With a special parameterization, Kobbelt[3] derived a similar Laplacian operator for subdivision modeling purpose. Instead of solving the large sparse system straightforward, both algorithms use a local iterative technique to fair a mesh with large data set. Due to the rough approximation of the Laplacian operator, however, sometimes the resulting meshes may be accompanied with serious shrinkage and distortion, which is unacceptable for modeling purpose.

In order to reduce the shrinkage and distortion, Taubin[8] use a linear combination of the first and second Laplacian operator to fair polygonal meshes. The resulting meshes, however, heavily depend on the choice of the two mesh-related constants. Recently, Mathieu et al.[1] and Vollmer et al.[2] presented two experimental methods to achieve this goal. In this paper, we introduce a novel noise removal algorithm without shrinkage to handle the mentioned problem from a different viewpoint. The key of this algorithm is to keep all triangle barycenters invariant at each smoothing step by introducing some constraints to the energy minimization functional. The relaxation constraints not only prevent the resulting mesh from shrinkage, but also ensure the vertices to have enough degree of freedom.

1 Laplacian smoothing algorithm

Without loss of generality, we just discuss the triangular meshes in the following. Of course, our algorithm may be generalized to arbitrary polygonal meshes. A triangular mesh *M* is represented as a triple $<V,T,X>$, where $V=\{1,2,3,\ldots,N\}$ is its vertex index set, $T\subseteq 2^V$ is its triangle set composed of all its triangles. Each triangle $t\in T$ is represented as an ordered vertex index triple $t=<i,j,k>$. The geometric realization of the mesh $X:V\to R^3$ is a mapping from the vertex

indices to their location in 3D space. Let $\mathrm{Adj}(i) \subseteq V$ be the set of the 1-ring neighboring vertices of vertex i, and $\left|\mathrm{Adj}(i)\right|$ be its valence (i.e. the number of its 1-ring neighbors). For short hand, we use x_i to replace $X(i)$ to describe the position of vertex i, so the geometry of M can be defined as a vector X consisting of all its vertices, namely, $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix}^T$.

By simple derivation, the variational derivatives of the mentioned energy functionals correspond to the Laplacian and the second Laplacian respectively:

$$L(X) = X_{uu} + X_{vv} \qquad L^2(X) = L \circ L(X) = X_{uuuu} + 2X_{uuvv} + X_{vvvv}$$

For discrete polygonal meshes, the Laplacian operator can be represented as

$$L(x_i) = \sum_{j \in \mathrm{Adj}(i)} \omega_{ij} \left(x_j - x_i \right) \tag{1}$$

Obviously, the discrete Laplacian operator is a linear combination of the neighboring vertices of a vertex. Writing the equation in matrix form, we have $L(X) = KX$, where $K = \left(\omega_{ij}\right)_{N \times N}$. In Equation (1), ω_{ij} are the non-negative weights satisfying $\sum_{j \in \mathrm{Adj}(i)} \omega_{ij} = 1$, and $\omega_{ij} = 0$ when $j \notin \mathrm{Adj}(i)$. Therefore, K is a sparse matrix.

The mesh fairing issue can be regarded as a diffusion process [1]. By making integration over time, the small disturbance or noise will quickly disperse in its neighborhood so that the mesh becomes smooth. However, this approach may result in serious shrinkage and distortion.

2 Barycenter constrained fairing algorithm

In this section, we derive our fairing algorithm, called barycenter constrained fairing algorithm, by introducing barycenter constraints to the energy functional, which prevents the mesh from shrinkage and distortion during smoothing. As discussed above, the energy of mesh M can be discretely approximated by

$$E(M) = \left\| L(X) \right\|^2 = \sum_{i \in V} \left| L(x_i) \right|^2 = X^T \left(K^T K \right) X \tag{2}$$

To avoid shrinkage and distortion, we need to introduce some constraints to the discrete energy functional. In this paper, the barycenter constraints are used.

Let X^0 be the initial vertex positions of mesh M, the barycenter constraint of triangle

$t =< i, j, k > \in T$ can then described as

$$\left(x_i + x_j + x_k\right) - \left(x_i^0 + x_j^0 + x_k^0\right) = 0$$

Thus, the constraints on the whole mesh can be written in the following matrix form:

$$H(X - X^0) = 0$$

Here, each triangle $t =< i, j, k > \in T$ corresponds to a row in the $N \times N$ matrix H with elements $h_{ts} = \begin{cases} 1 & s = i, j \text{ or } k \\ 0 & \text{otherwise} \end{cases}$. The fairing for mesh *M* can then be described as

$$\begin{cases} \min & X^T\left(K^T K\right)X \\ \text{subject to} & H(X - X^0) = 0 \end{cases} \tag{3}$$

This is a typical constrained optimization problem, which can be transferred to unconstrained one using the well-known penalty approach:

$$\min \quad X^T\left(K^T K\right)X + \mu \cdot \left(X - X^0\right)^T\left(H^T H\right)\left(X - X^0\right)$$

where μ is a positive constant to achieve a compromise between the minimization and the constraints. Solving a large sparse system is usually a time and memory consuming method for a mesh with large data set. In the following we will describe an iterative solution to Equation (3).

2.1 Iterative solution

Our local iterative solution is to minimize the local energy in the neighborhood of triangle *t* under the barycenter invariant constraint. Consider a triangle *t* and its neighboring triangles illustrated in Figure 1, let p_0, p_1, p_2 be its three vertices, we define the discrete energy in the neighborhood of triangle *t* similar to Equation (2):

$$E(t) = \sum_{i=0}^{2} L(x_i) = \|AP - Q\|^2 = \left(P^T A^T - Q^T\right) \cdot (AP - Q)$$

where $A = \begin{bmatrix} 1 & -\omega_{01} & -\omega_{02} \\ -\omega_{10} & 1 & -\omega_{12} \\ -\omega_{20} & -\omega_{21} & 1 \end{bmatrix}$, $P = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \end{bmatrix}$, $Q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \end{bmatrix}$, $q_i = \sum_{j=0}^{m_i} \omega_{ij} q_{i,j}$ $(i = 0, 1, 2)$ (Figure 1).

See Figure 1

The barycenter constraint of a triangle can be rewritten as

$$BP = C$$

with $B = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$, $C = BP^0$, $P^0 = \begin{bmatrix} p_0^0 & p_1^0 & p_2^0 \end{bmatrix}^T$ is the original position of the triangle. Consequently, the local relaxation in the neighborhood of triangle *t* can be defined as the minimizer of $E(t)$ under the barycenter constraint:

$$\begin{cases} \min & \left(P^T A^T - Q^T\right) \cdot \left(AP - Q\right) \\ \text{subject to} & BP = C \end{cases} \tag{4}$$

Since the functional is quadratic, it can be solved using the Lagrangian Multiplier method,

$$P = A^{-1}Q + \frac{1}{\left\| BA^{-1} \right\|^2} \left(C - BA^{-1}Q\right)\ A^{-1}\left(A^{-1}\right)^T B^T \tag{5}$$

There are two popular ways to choose weights ω_{ij}. One depends only on the local connectivity of meshes while the other considers both geometry and connectivity of meshes. Here, we use the first method to define the weights:

$$\omega_{ij} = \begin{cases} \dfrac{1}{n_i} & j \in \text{Adj}(i) \\ 0 & \text{otherwise} \end{cases}$$

Therefore, A^{-1}, BA^{-1} and $BA^{-1}\left(A^{-1}\right)^T B^T$ in Equation (5) only depend on the valence of vertex *i*. In most cases, the valence of a vertex is around 5, 6 or 7, instead of computing these matrices on the fly, we build look-up tables of the matrices for each valence number in a preprocessing step. Hence, we can efficiently calculate the new locations of three vertices of triangle *t* by looking up the tables.

The mentioned method simultaneously relaxes three vertices of a triangle to minimize the local energy every time. Note that there are several triangles surrounding a vertex, say *p*, which means one may obtain several new locations p_{new}^1, p_{new}^2, …, p_{new}^m for vertex *p* (*m* is its valence) by relaxing it for each of its surrounding triangles. In our current implementation, we determine the location of vertex *x* by averaging these evaluated locations:

$$x_{new} = \frac{1}{m}\sum_{i=1}^{m} p_{new}^i$$

We have test our algorithm on many models, and one of the results is shown in Figures 2. We first add some white noise to the original horse model, then use our proposed smooth method to remove the noise. Only one iterative step is needed in this example. Comparing the resulting model with the original one, we find that they are almost the same one! The example demonstrates the ability and stability of our algorithm to effectively remove noise with feature preservation.

2.2 Boundary Fairing

For an open mesh, we also need to smooth its boundary curves. Here, we extract the boundary curves from the mesh to smooth separately. Similarly, the Laplacian operator for a discrete curve, say $p_0 p_1 \cdots p_m$, is defined as follows:

$$L(p_i) = \omega_{i,-1}(p_{i-1} - p_i) + \omega_{i,1}(p_{i+1} - p_i)$$

where $\omega_{i,-1}$, $\omega_{i,1} > 0$, and $\omega_{i,-1} + \omega_{i,1} = 1$. For simplicity, they may be given as $\omega_{i,-1} = \omega_{i,1} = \frac{1}{2}$. Adhering to the same idea mentioned before, the barycenter/midpoint constraints may also be introduced to prevent the curves from shrinkage and distortion. It is easy to deduce the local iterative equation:

$$\begin{bmatrix} p_{i-1} \\ p_i \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{2}{3} \end{bmatrix} \begin{bmatrix} p_{i-2} \\ p_{i+1} \end{bmatrix} + \begin{bmatrix} d \\ d \end{bmatrix} \quad \text{with } d = \left[\left(p_{i-1}^0 + p_i^0 \right) - \left(p_{i-2} + p_{i+1} \right) \right] / 2$$

See Figure 2

References

[1] Desbrun M, Meyer M, Schroder P, et al. Implicit fairing of irregular meshes using diffusion and curvature flow. In Proceedings of SIGGRAPH' 99 Conference, USA, Addison-Wesley, 317–324.

[2] Vollmer J, Mencl R, Müller H. Improved Laplacian smoothing of noisy surface meshes. In Proceedings of EUROGRAPHICS ’99 Conference, UK, Blackwell Publishers, 211~218.

[3] Kobbelt L, Campagna S, Vorsatz J, et al. Interactive multiresolution modeling on arbitrary meshes. In Proceedings of SIGGRAPH' 98 Conference, USA, Addison-Wesley, 105-114.

[4] Curless B, Levoy M. A volumetric method for building complex models from range images. In Proceedings of SIGGRAPH' 96 Conference, USA, Addison-Wesley, 303–312.

[5] Taubin G. A signal processing approach to fair surface design. In Proceedings of SIGGRAPH' 95 Conference, USA, Addison-Wesley, 351–358.

[6] Hoppe H, DeRose T, Duchamp T, et al. Surface reconstruction from unorganized points. In Proceedings of SIGGRAPH' 92 Conference, USA, Addison-Wesley,71–78.

[7] Morton H P, Sequin C H. Functional optimization for fair surface design. In Proceedings of SIGGRAPH' 92 Conference, USA, Addison-Wesley, 167-176.

[8] Welch W, Witkin A. Varational surface modeling. In Proceedings of SIGGRAPH' 92 Conference, USA, Addison-Wesley, 157-166.

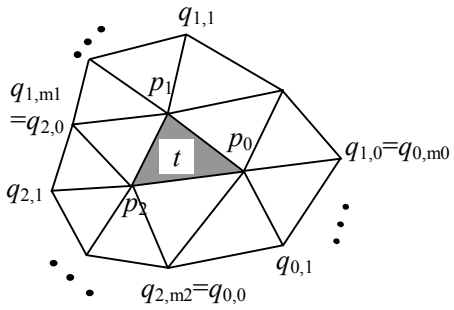

Figure 1 Triangle t and its neighborhood

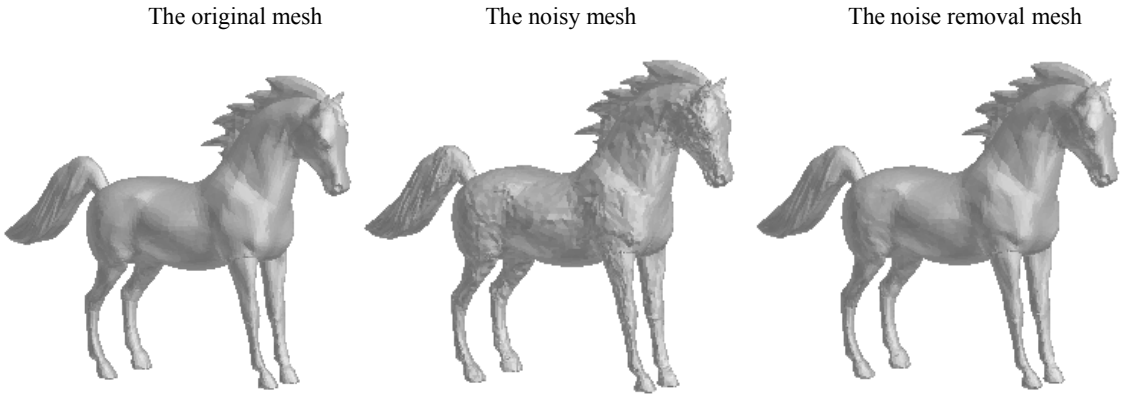

Figure 2 Noise removal for the horse model