Available at
www.ElsevierMathematics.com
POWERED BY SCIENCE DIRECT®

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

ELSEVIER Journal of Computational and Applied Mathematics 163 (2004) 295–309

www.elsevier.com/locate/cam

Surface fairing and featuring by mean curvature motions☆

Guoliang Xu

LSEC, ICMSEC, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Beijing 100080, China

Received 10 December 2002; received in revised form 15 June 2003

Abstract

The aim of this paper is smoothing triangular surface meshes using a diffusion process while preserving or enhancing several important features. These features include sharp feature, detail structure, homogeneity of the mesh, interpolation of some vertices and approximation of the initial mesh. We realize these goals using a unified partial differential equation model, so that the numerical solving process of the equation is significantly simplified.

© 2003 Elsevier B.V. All rights reserved.

Keywords: Heat equation; Fairing; Feature preserving; Loop's subdivision

1. Introduction

The aim of this paper is smoothing and denoizing triangular surface meshes by solving a partial differential equation (PDE), which is a generalization of the heat equation customized to surfaces. The heat equation has been successfully used in the image processing for about two decades. The literature on this PDE based approach to image processing is large [7,15,16,22]. It is well known that the solution of heat equation $\partial_t\rho - \Delta\rho = 0$, based on the Laplacian Δ, at time τ for a given initial image ρ_0 is the same as taking a convolution of the Gauss filter $G_\sigma(x) = (1/2\pi\sigma^2)\exp(-|x|^2/(2\sigma^2))$ with standard deviation $\sigma = \sqrt{2\tau}$ and image ρ_0. Taking the convolution of G_σ and image ρ_0 is performing a weighted averaging process to ρ_0. When the standard deviation σ become larger, the averaging is taken over a larger area. This explains the filtering effect of the heat equation to noisy images. The generalization of the heat equation for a surface formulation has recently been proposed [2,3] and shown to be very effective even for higher-order methods [1] and for two-manifold in high-dimensional space. The counterpart of the Laplacian Δ is the Laplace–Beltrami operator [5]

☆ Supported in part by NSFC 10241004 and National Innovation Fund 1770900, Chinese Academy of Sciences.

E-mail address: xuguo@lsec.cc.ac.cn (G. Xu).

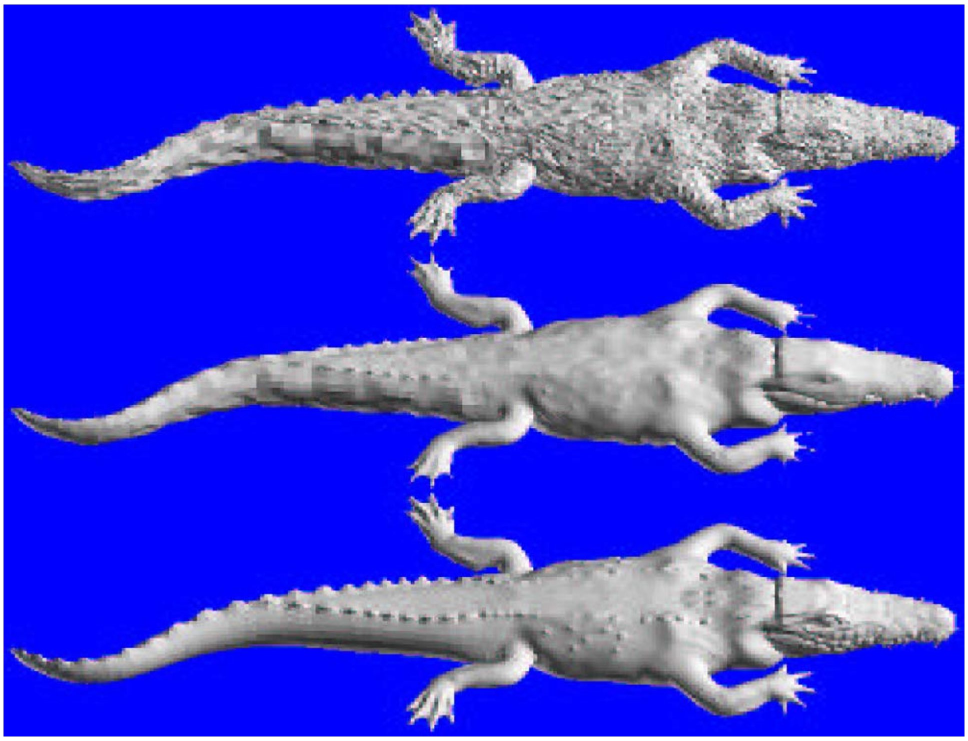

Fig. 1. The top figure is the initial geometry mesh. The second figure is the faired mesh after 4 fairing iterations with identity adaptor and timestep $\tau = 0.0001$. The last one is the faired mesh after 4 fairing iterations with nonidentity adaptor $a(x)$ and $\tau = 0.0016$.

Δ_M for a surface M. Hence, the heat equation is generalized to

$$\partial_t x - \Delta_M x = 0 \tag{1.1}$$

for surface point x, which is also known as mean curvature flow. Using this generalized heat equation to smooth surface has shown to be effective. However, the smoothing effect is strong such that some undesirable features may occur.

Over/under-fairing. Unlike the 2D images, where the grids are often structured, the discretized triangular surfaces are often un-structured. Certain regions of the surface meshes are often very dense, with a wide spectrum of noise distribution. Applying a single Gauss-like filter to such surface meshes would have the following side-effects: (1) the lower frequency noise is not filtered (*under-fairing*) if the evolution period of time is suitable for removing high frequency noise, (2) detailed features are removed unfortunately, as higher-frequency noise (*over-fairing*) if the evolution period of time is suitable for removing low-frequency noisy components. Fig. 1 illustrates this under-fairing and over-fairing effects. The top figure is the input mesh, the next one is the evolution result by Eq. (1.1). It can be seen that the large features (see the tails of the crocodiles) are not fair enough but the detailed features are already over-faired (see the snout of the crocodiles). Hence, a phenomena that often appears for the triangular surface mesh denoizing is that whenever the desirable smoothing results are achieved for larger features, the smaller features are lost. Prior work has attempted to solve the over-fairing problem by using an anisotropic diffusion tensor in the diffusion equation [1,2]. However, this is far from satisfactory. One of the aims of this paper is to overcome the under-fairing and over-fairing dilemma in solving the diffusion equation by involving an adaptor in Eq. (1.1).

Singularities. It is known that mean curvature flow (1.1) moves the vertices of the mesh in the normal direction of the surface. Such a motion could cause very tiny even collapsed triangles. These tiny and collapsed triangles make the stiffness matrix of the finite element discretization ill-conditioned or even singular. Hence, to avoid generating tiny and collapsed triangles is crucial for producing high quality mesh. Our second goal of this paper is to homogenize the mesh during smoothing process by introducing a homogenizer in the equation.

No steady-solution. Under mean curvature motion, the evolved surface shrink to the origin according to the following Eqs. [2,19]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{Area}(M(t)) = -\int_{M(t)} H^2\,\mathrm{d}x, \quad \frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{Volume}(M(t)) = -\int_{M(t)} H\,\mathrm{d}x, \tag{1.2}$$

where Area($M(t)$) and Volume($M(t)$) represent the area of $M(t)$ and volume enclosed by $M(t)$, respectively, H is the mean curvature. Hence, the evolution equation has no steady-solution before the surface degenerating to zero. Such a feature of the motion makes the question "when to stop the diffusion process or what is the stopping criteria" hard to answer. We shall modify the diffusion model, so that a steady-solution exists.

1.1. Previous work

For the PDE-based surface fairing or smoothing, several methods have been proposed [1–4] recently. Desbrun et al. [3,4] also use Laplacian, which is discretized as the umbrella operator in the spatial direction. In the time direction discretization, they propose to use the semi-implicit Euler method to obtain a stable numerical scheme. Clarenz et al. [2] generalize the Laplacian to the Laplace–Beltrami operator Δ_M, and use linear finite elements to discretize the equation. In the paper [1], the problem is reformulated for two-dimensional Riemannian manifold embedded in $\mathbb{R}^k$ aiming at smoothing geometric surfaces and functions on surfaces simultaneously. The C^1 higher-order finite element space used is defined by the Loop's subdivision (box spline). One of the shortcomings of all these proposed methods that we address here is their nonadaptivity. Hence they quite often suffer from under-fairing or over-fairing problems and singularity problem.

Another approach for smoothing and denoizing of polyhedral surface is minimizing some energy functionals. In this approach, one constructs an optimization problem that minimizes certain objective functions associated with geometric surface characteristics [6,8,9,14,18,23], such as thin plate energy, membrane energy [10], total curvature [11,24], or sum of distances [13]. Using local interpolation or fitting, or replacing differential operators with divided difference operators, the optimization problems are discretized to arrive at finite dimensional linear or nonlinear systems. Approximate solutions are then obtained by solving the constructed systems. Such an approach is usually computational expensive and lacks local shape control.

1.2. Our approach

For a feature-adaptive or error-adaptive mesh, the ideal evolution strategy would be to correlate the evolution speed relative to the mesh density. In short, we desire the lower-frequency errors use a faster evolution rate and the higher-frequency errors succumbs to a slower evolution rate. To achieve this goal, the discretization in the time direction could be mesh adaptive, using a timestep $\tau a(x)$ where $a(x)$ depends on the position x of the surface. The part of the surface that is coarse uses larger $\tau a(x)$. We have observed that the adaptive discretization in the time direction is equivalent to introducing an adaptor in the PDE while using a uniform time step discretization. Hence in this paper, we do not involve variable time step, but introduce an adaptor. The last figure of Fig. 1 shows the improvement of adaptive evolution over the nonadaptive evolution. We also introduce a homogenizer in the equation for avoiding singularities and a right-handed side for interpolating and approximating the initial mesh.

The remaining of the paper is organized as follows: Section 2 introduces the used terminologies and some basic facts in the differential geometry. Section 3 summarizes the diffusion PDE model used, followed by the discretization Section 4. In the spatial direction, the discretization is realized using the C^1 smooth finite element space defined by the limit function of Loop's subdivision (box spline), while the discretization in the time direction is realized by semi-implicit Euler scheme. The implementation details are given in Section 5. Section 6 concludes the paper.

2. Preliminaries

In this section, we introduce some terminologies and basic facts in the differential geometry (see also paper by [1] for the higher-dimensional case).

Tangent Space of Differential Manifold. Let $M \subset \mathbb{R}^3$ be a two-dimensional manifold, and $\{U_\alpha, x_\alpha\}$ be the differentiable structure. The mapping x_α with $x \in x_\alpha(U_\alpha)$ is called a parameterization of M at x. Denoting the coordinate U_α as (ξ_1, ξ_2), then the tangent space T_xM at $x \in M$ is spanned by $\{\partial/\partial\xi_1, \partial/\partial\xi_2\}$. For a given point $x \in x_\alpha(U_\alpha) \subset M$, the tangent vector components $\partial/\partial\xi_1$ and $\partial/\partial\xi_2$ depend upon α, unlike T_xM. The set $TM = \{(x, v);\ x \in M,\ v \in T_xM\}$ is called a tangent bundle.

Riemannian Manifold. To define integration on M, a *Riemannian metric* (inner product) is required. A differentiable manifold with a given Riemannian metric is called a *Riemannian Manifold*. A Riemannian metric $\langle , \rangle_x$ of M is a symmetric, bilinear and positive-definite form on the tangent space T_xM. Since M is a sub-manifold of Euclidean space $\mathbb{R}^3$, we use the *induced metric*

$$\langle u, v\rangle_x = u^{\mathrm{T}} v, \quad u, v \in T_xM.$$

Integration. Let f be a function on M, and let $\{\phi_\alpha\}_\alpha$ be a finite partition of unity on M with support $\phi_\alpha \subset U_\alpha$. Then define

$$\int_M f\,\mathrm{d}x := \sum_\alpha \int_{U_\alpha} \phi_\alpha f(x_\alpha)\sqrt{\det(g_{ij})}\,\mathrm{d}\xi_1\,\mathrm{d}\xi_2, \tag{2.1}$$

where $g_{ij} = \langle \partial/\partial\xi_i, \partial/\partial\xi_j \rangle_x$. Then we can define the inner product of two functions on M and two vector fields on TM as

$$(f,g)_M = \int_M fg\,\mathrm{d}x, \quad f,g \in C^0(M), \quad (\phi,\psi)_{TM} = \int_M \langle \phi,\psi \rangle\,\mathrm{d}x, \quad \phi,\psi \in TM. \tag{2.2}$$

Gradient. Suppose $f \in C^1(M)$. The gradient $\nabla_M f \in T_xM$ of f is defined by the following conditions:

$$t_i^{\mathrm{T}} \nabla_M f = \frac{\partial(f \circ x)}{\partial \xi_i}, \quad i = 1,2, \tag{2.3}$$

where $t_i = \partial x/\partial\xi_i$ are the tangent vectors. Note that $\nabla_M f$ is invariant under the surface local reparameterization. From (2.3), we have

$$\nabla_M f = [t_1,\ t_2] G^{-1} \left[\frac{\partial(f \circ x)}{\partial \xi_1},\ \frac{\partial(f \circ x)}{\partial \xi_2} \right]^{\mathrm{T}}, \tag{2.4}$$

where

$$G^{-1} = \frac{1}{\det G} \begin{bmatrix} g_{22} & -g_{12} \\ -g_{21} & g_{11} \end{bmatrix}, \quad G = \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix},$$

and G is known as the first fundamental form.

Divergence. The divergence $\mathrm{div}_M\, \psi$ for a vector field $\psi \in TM$ is defined as the dual operator of the gradient [17]:

$$(\mathrm{div}_M\, v, \phi)_M = -(v, \nabla_M \phi)_{TM}, \quad \forall \phi \in C_0^\infty(M), \tag{2.5}$$

where $C_0^\infty(M)$ is a subspace of $C^\infty(M)$, whose elements have compact support.

3. Geometric diffusion equations

Recently, people have solved the following nonlinear system of parabolic differential equations [1,2] for surface fairing:

$$\partial_t x(t) - \Delta_{M(t)} x(t)) = 0, \tag{3.1}$$

where $\Delta_{M(t)} = \mathrm{div}_{M(t)} \circ \nabla_{M(t)}$ is the Laplace–Beltrami operator on $M(t)$, $M(t)$ is the solution surface at time t and $x(t)$ is a point on the surface. $\nabla_{M(t)}$ is the gradient operator on the surface, and $\mathrm{div}_{M(t)}$ is the divergent operator acting on a vector field on the surface.

3.1. Anisotropic diffusion

To enhance sharp features, a *diffusion tensor D*, acting on the gradient, has been introduced [1,2].

$$\partial_t x(t) - \mathrm{div}_{M(t)}(D(x) \nabla_{M(t)} x(t)) = 0. \tag{3.2}$$

The diffusion tensor $D(x)$ is a symmetric and positive definite operator from T_xM to T_xM. Here T_xM is the tangent space of M at x. The detailed discussion for choosing the diffusion tensor can be found in [1,2]. We omit the discussion on choosing $D(x)$ in this paper and use an identity diffusion tensor. The aim to mention this diffusion tensor is to show that this tensor could be incorporated in our unified PDE model.

3.2. Adaptive diffusion

We already know that Eq. (3.1) describes the mean curvature motion. Its regularization effect could be seen from Eq. (1.2). From these equations, we see that the evolution speed depends on the mean curvature of the surface but not on the density of the mesh. Hence if the mesh is spatially adaptive, the dense parts that have detailed structures and hence have larger curvatures are very possibly over-faired. Therefore, we introduce an *adaptor* $a(x)$ in this model for achieving adaptive fairing effect. Then (3.1) becomes

$$\partial_t x(t) - a(x)\,\mathrm{div}_{M(t)}(\nabla_{M(t)} x(t)) = 0, \tag{3.3}$$

where $a(x)$ is a smooth function which is adaptive to the mesh density (see Section 5 for the definition of $a(x)$).

3.3. Homogenization of the mesh

To homogenize the mesh while smoothing, we introduce a homogenizer $h(x)$ in the equation

$$\partial_t x(t) - a(x)\,\mathrm{div}_{M(t)}(h(x)\nabla_{M(t)} x(t)) = 0. \tag{3.4}$$

It is easy to derive that

$$\mathrm{div}_M(h\nabla_M f) = (\nabla_M f)^{\mathrm{T}}\nabla_M h + h\Delta_M f, \tag{3.5}$$

where f, h are smooth functions on M. From (2.4), (3.5) and the fact that $\Delta_M x = 2H(x)n(x)$, we could rewrite (3.4) as

$$\partial_t x(t) = a(x)[\nabla h(x) + 2h(x)H(x)n(x)]. \tag{3.6}$$

Eq. (3.6) implies that the motion of the surface $M(t)$ can be decomposed into two parts, one is the tangential displacement caused by $\nabla h(x)$, and the other is the normal displacement (mean curvature motion) caused by $2h(x)H(x)n(x)$.

We shall define $h(x)$ such that it is adaptive to the density of the mesh in the sense that it takes smaller values at denser regions of the mesh. Consider a case where a small triangle is surrounded by large triangles. In such a case, function $h(x)$ is small on the triangle and larger elsewhere. This implies that the gradient of $h(x)$ on the small triangle points to the outside direction, and the tangential displacement makes the small triangle become enlarged. If the density of the mesh is even, then $h(x)$ is nearly a constant. Then the tangential displacement is minor. Hence, $h(x)$ has homogenizing effect. Such an effect is nice and important, as it avoids producing collapsed or tiny triangles in the faired meshes.

3.4. Approximation

A term $r \in \mathbb{R}^3$ on the right-handed side of the equation, which represents an external force, is imposed. Hence the model becomes

$$\partial_t x(t) - a(x)\,\mathrm{div}_{M(t)}(h(x)\nabla_{M(t)} x(t)) = r(x(t)). \tag{3.7}$$

The function r is chosen in the following form:

$$r(x(t)) = \omega_A(x(0) - x(t)), \quad \omega_A \geqslant 0. \tag{3.8}$$

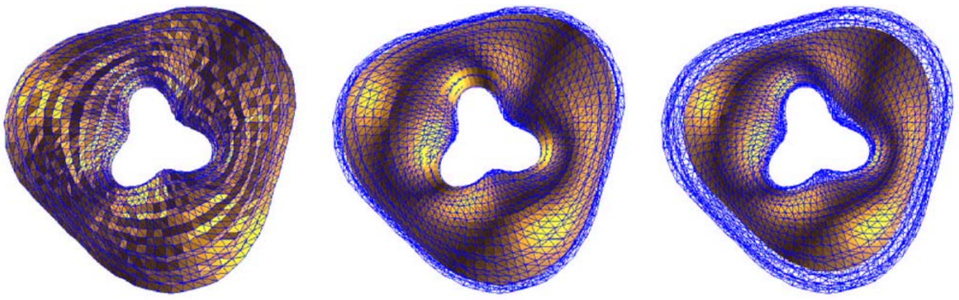

Fig. 2. The left is the input. The remaining two figures are the steady-solutions for $\omega_A = 20.0$ and $\omega_A = 6.0$, respectively. The net outside each of the surface meshes consists of all the edges of the initial mesh.

This term is used to approximate the initial mesh in the smoothing process, so that the smoothed surfaces do not evolve too much from the initial surface $M(0)$. ω_A is a user specified parameter, where the subscript A stands for approximation. If $a(x) = h(x) = 1$, then (3.7) becomes $\partial_t x(t) = 2H(x)n(x) + \omega_A(x(0) - x(t))$. Hence the equation described is a motion that is decomposed into the mean curvature motion, caused by $2H(x)n(x)$ and in the normal direction, and a motion towards the original surface, caused by $\omega_A(x(0)-x(t))$ and in the direction of $x(0)-x(t)$. The magnitude of ω_A determines which part of two motions dominates composite motion. Fig. 2 shows the effect of ω_A. The left figure shows the input mesh. The figures in the middle and right show the stable states of the evolution for $\omega_A = 20.0$ and $\omega_A = 6.0$, respectively. Here we choose $\tau = 0.05$.

3.5. Interpolation

Adding a term in external force $r(x(t))$, we could even interpolate approximately some vertices of the initial mesh. Hence the final model we use is

$$\begin{aligned} &\partial_t x(t) - a(x)\,\mathrm{div}_{M(t)}(h(x)\nabla_{M(t)} x(t)) = r(x(t)), \\ &M(0) = M, \end{aligned} \tag{3.9}$$

where

$$r(x(t)) = \omega(x)(x(0) - x(t)), \quad \omega(x) = \omega_I(x) + \omega_A,$$

$\omega_I(x) \geqslant 0$ is a smooth function which is large at the interpolation vertices, and zeros at other vertices, the subscript I of $\omega_I(x)$ stands for interpolation.

Since $\partial_t x(t) \to 0$ as $t \to \infty$, the steady-solution $M_s = M(\infty)$ satisfies the following equation:

$$-a(x)\,\mathrm{div}_{M_s}(h(x)\nabla_{M_s} x) = r(x), \tag{3.10}$$

where x is a surface point on M_s.

3.6. Variational form

Using (2.5), the final diffusion problem (3.9) could be reformulated into the following variational form:

Find a smooth $x(t)$ such that

$$(a(x)^{-1}\partial_t x(t),\theta)_{M(t)} + (h(x)\nabla_{M(t)}x(t),\nabla_{M(t)}\theta)_{TM(t)} = (a(x)^{-1}r(x(t),\theta))_{M(t)},$$
$$M(0) = M, \tag{3.11}$$

for any $\theta \in C_0^\infty(M(t))$. Similarly, the variational form of (3.10) is as follows:

Find a smooth x *such that*

$$(h(x)\nabla_{M_s}x,\nabla_{M_s}\theta)_{TM_s} = (a(x)^{-1}r(x),\theta)_{M_s}, \tag{3.12}$$

for any $\theta \in C_0^\infty(M_s)$. These variational forms are the starting point for the discretization.

4. Discretization

4.1. Temporal discretization

We discretize Eq. (3.11) in the time direction first and then in the spatial direction. Given an initial value $x(0)$, we wish to have a solution $x(t)$ of (3.11) at $t=\tau$. Using a semi-implicit Euler scheme, we have the following time direction discretization:

Find a smooth $x(\tau)$ such that

$$\left(\frac{x(\tau)-x(0)}{\tau a(x(0))},\theta\right)_{M(0)} + (h(x(0))\nabla_{M(0)}x(\tau),\nabla_{M(0)}\theta)_{TM(0)} = (a(x(0))^{-1}r(x(\tau)),\theta)_{M(0)} \tag{4.1}$$

for any $\theta \in C_0^\infty(M(0))$. If we want to go further along the time direction, we could treat the solution at $t=\tau$ as the initial value and repeat the same process. Hence, we consider only one time step in our analysis. Note that if we use a variable timestep $\tau a(x(0))$ to discretize the equation $\partial_t x(t) - \mathrm{div}_{M(t)}(h(x)\nabla_{M(t)}x(t)) = 0$ in the time direction, we will arrive at the same equation as (4.1). This is the reason we do not use variable timestep in the time discretization but introduce an adaptor in the PDE.

4.2. Spatial discretization

The function in our finite element space is locally parameterized as the image of the unit triangle

$$\mathscr{T} = \{(\xi_1,\xi_2)\in\mathbb{R}^2 : \xi_1 \geqslant 0, \xi_2 \geqslant 0, \xi_1+\xi_2 \leqslant 1\}.$$

That is, $(1-\xi_1-\xi_2,\xi_1,\xi_2)$ are the barycentric coordinates of the triangle. Using this parameterization, our discretized representation of M is $M=\bigcup_{\alpha=1}^{k}\mathscr{T}_\alpha, \mathring{\mathscr{T}}_\alpha\cap\mathring{\mathscr{T}}_\beta=\emptyset$ *for* $\alpha\neq\beta$, where $\mathring{\mathscr{T}}_\alpha$ is the interior of $\mathscr{T}_\alpha$. Each triangular patch is parameterized locally as $x^\alpha : \mathscr{T}\to\mathscr{T}_\alpha;\ (\xi_1,\xi_2)\mapsto x^\alpha(\xi_1,\xi_2)$. Under this parameterization, tangents and gradients can be computed directly. The integration on surface M is given by

$$\int_M f\,\mathrm{d}x := \sum_\alpha \int_{\mathscr{T}} f(x_\alpha(\xi_1,\xi_2))\sqrt{\det(g_{ij})}\,\mathrm{d}\xi_1\,\mathrm{d}\xi_2.$$

The integration on triangle $\mathscr{T}$ is computed adaptively by numerical methods.

Let M_d be the given initial triangular mesh, x_i, $i=1,\dots,m$ be its vertices. We shall use C^1 smooth quartic Box spline basis functions to span our finite element space. The piecewise quartic basis function at vertex x_i, denoted by ϕ_i, is defined by the limit of Loop's subdivision for the zero control values everywhere except at x_i where it is one (see paper [1] for a detailed description). For simplicity, we call it the *Loop's basis*.

4.2.1. Loop's subdivision

In Loop's subdivision scheme, the initial control mesh and the subsequent refined meshes consist of triangles only. In the refinement, each triangle is subdivided linearly into 4 sub-triangles. Then the vertex position of the refined mesh is computed as the weighted average of the vertex position of the unrefined mesh. Consider a vertex x_0^k at level k with neighbor vertices x_i^k for $i=1,\dots,n$, where n is the valence of vertex x_0^k. The coordinates of the newly generated vertices x_i^{k+1} on the edges of the previous mesh are computed as

$$x_i^{k+1} = \frac{3x_0^k + 3x_i^k + x_{i-1}^k + x_{i+1}^k}{8}, \quad i=1,\dots,n,$$

where index i is to be understood modulo n. The old vertices get new positions according to

$$x_0^{k+1} = (1-na)x_0^k + a(x_1^k + x_2^k + \cdots + x_n^k),$$

where

$$a = \frac{1}{n}\left[\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right].$$

Note that all newly generated vertices have a valence of 6, while the vertices inherited from the original mesh at level zero may have a valence other than 6. We will refer to the former case as *ordinary* and the latter case as *extraordinary*.

Let e_j, $j=1,\dots,m_i$ be the two-ring neighborhood elements of x_i. Then if e_j is regular (meaning its three vertices have valence 6), explicit Box-spline expressions exist [20,21] for ϕ_i on e_j. Using these explicit Box-spline expressions, we derive the BB-form expressions for the basis functions ϕ_i. These expressions could be used to evaluate ϕ_i in forming linear system (4.3). If e_i is irregular, local subdivision is needed around e_i until the parameter values of interest are interior to a regular patch. An efficient evaluation method, that we have implemented, is the one proposed in [20].

Compared with the linear finite element space, using the higher-order C^1 smooth finite element space spanned by Loop's basis does have advantages. The basis functions of this space have compact support (within 2-rings of the vertices). This support is bigger than the support (within 1-ring of the vertices) of hat basis functions that are used for the linear discrete surface model. Such a difference in the size of support of basis functions makes our evolution more efficient than those previously reported, due to the increased bandwidth of the affected frequencies. The reduction speed of high frequency noise in our approach is not that drastic, but still fast, while the reduction speed of lower-frequency noise is not slow. Hence, the bandwidth of affected frequencies is wider. A comparative result showing the superiority of the Loop's basis function is given in paper [1].

4.2.2. Finite element discretization of (4.1)

Let $V_{M(0)}$ be the finite-dimensional space spanned by the Loop's basis functions $\{\phi_i\}_{i=1}^m$. Then $V_{M(0)} \subset C^1(M(0))$. For the given time step τ, let $x(k\tau) = \sum_{i=1}^m x_i(k\tau)\phi_i$ be the numerical solution after kth iteration, and let $\theta = \phi_j$. Then for obtaining the numerical solution at time $t = k\tau + \tau$, Eq. (4.1) could be approximated by

$$\begin{aligned}\sum_{i=1}^m \frac{x_i(k\tau+\tau)-x_i(k\tau)}{\tau}\left(\frac{\phi_i}{a(x(k\tau))},\phi_j\right)_{M(k\tau)}\\ +\sum_{i=1}^m x_i(k\tau+\tau)(h(x(k\tau))\nabla_{M(k\tau)}\phi_i,\nabla_{M(k\tau)}\phi_j)_{TM(k\tau)}\\ =\sum_{i=1}^m (x_i(0)-x_i(k\tau+\tau))\left(\frac{\omega(x(k\tau))\phi_i}{a(x(k\tau))},\phi_j\right)_{M(k\tau)}\end{aligned} \tag{4.2}$$

for $j=1,\dots,m$, where $x_i(0) := x_i$ is the ith vertex of the input mesh M_d. Eq. (4.2) is a linear system for unknown $x_i(k\tau+\tau)$. Now (4.2) can be written in the following matrix form:

$$(\mathscr{M}_0+\tau\mathscr{M}_1+\tau\mathscr{L})X(k\tau+\tau)=\mathscr{M}_0X(k\tau)+\tau\mathscr{M}_1X(0), \tag{4.3}$$

where $X(\ell\tau)=[x_1(\ell\tau),\dots,x_m(\ell\tau)]^{\mathrm{T}}$, for $\ell=0,1,\dots,$ and

$$\begin{aligned}&\mathscr{M}_0=[(a^{-1}\phi_i,\phi_j)_{M(k\tau)}]_{i,j=1}^m,\quad \mathscr{M}_1=[(a^{-1}\omega\phi_i,\phi_j)_{M(k\tau)}]_{i,j=1}^m,\\ &\mathscr{L}=[(h\nabla_{M(k\tau)}\phi_i,\nabla_{M(k\tau)}\phi_j)_{TM(k\tau)}]_{i,j=1}^m.\end{aligned} \tag{4.4}$$

Note that $\mathscr{M}_0$, $\mathscr{M}_1$ and $\mathscr{L}$ are symmetric. Since $\phi_1,\phi_2,\dots,\phi_m$ are linearly independent and have compact support, $\mathscr{M}_0$ and $\mathscr{M}_1$ are sparse and positive definite. Similarly, $\mathscr{L}$ is symmetric and nonnegative definite. Hence, $\mathscr{M}_0+\tau\mathscr{M}_1+\tau\mathscr{L}$ is symmetric and positive definite.

The coefficient matrix of system (4.3) is highly sparse. An iterative method for solving such a system is desirable. We solve it by the conjugate gradient method with a diagonal preconditioning.

4.2.3. Finite element discretization of (3.12)

Eq. (3.12) is a nonlinear system. We solve it by an iterative process, i.e. we solve progressively a sequence of linear equations.

Find a smooth $x^{(k+1)}$ such that

$$\varepsilon_{k+1}(h(x^{(k)})\nabla_{M^{(k)}}x^{(k+1)},\ \nabla_{M^{(k)}}\theta)_{TM^{(k)}}=(a(x^{(k)})^{-1}r(x^{(k+1)}),\ \theta)_{M^{(k)}}, \tag{4.5}$$

for any $\theta\in C_0^\infty(M^{(k)})$ and $k=0,1,\dots,n-1$, where $0<\varepsilon_1<\varepsilon_2<\cdots<\varepsilon_n=1$ (say $\varepsilon_{k+1}=(k+1)/n$), $x^{(k+1)}$ is a point on the solution surface $M^{(k+1)}$. $M^{(0)}$ is chosen to be the initial surface M and $M^{(n)}$ is the required approximate solution of (3.12).

Denote the numerical solution of (4.5) by $x^{(k+1)} = \sum_{i=1}^{m} x_i^{(k+1)}\phi_i$ and put $\theta = \phi_j$. Then Eq. (4.5) could be approximated by

$$\sum_{i=1}^{m} x_i^{(k+1)} \varepsilon_{k+1}(h(x^{(k)})\nabla_{M^{(k)}}\phi_i, \nabla_{M^{(k)}}\phi_j)_{TM^{(k)}}$$

$$= \sum_{i=1}^{m} (x_i^{(0)} - x_i^{(k+1)})(a(x^{(k)})^{-1}\omega(x^{(k)})\phi_i, \phi_j)_{M^{(k)}} \tag{4.6}$$

for $j = 1, \ldots, m$. Eq. (4.6) is a linear system for unknown $x_i^{(k+1)}$. It can be written in the following matrix form:

$$(\mathscr{M}_1 + \varepsilon_{k+1}\mathscr{L})X^{(k+1)} = \mathscr{M}_1 X^{(0)}, \tag{4.7}$$

where $X^{(k+1)} = [x_1^{(k+1)}, \ldots, x_m^{(k+1)}]^{\mathrm{T}}$, and

$$\mathscr{M}_1 = [(a^{-1}\omega\phi_i, \phi_j)_{M^{(k)}}]_{i,j=1}^{m}, \qquad \mathscr{L} = [(h\nabla_{M^{(k)}}\phi_i, \nabla_{M^{(k)}}\phi_j)_{TM^{(k)}}]_{i,j=1}^{m}. \tag{4.8}$$

Again, $\mathscr{M}_1 + \varepsilon_{k+1}\mathscr{L}$ is symmetric, positive definite and highly sparse.

5. Implementation details

This section addresses several implementation issues of problem (3.11), including defining the adaptor $a(x)$, homogenizer $h(x)$ and the interpolator $\omega_I(x)$.

5.1. Defining the adaptor $a(x)$

Now we illustrate how $a(x)$ is defined. At each vertex x_i of the mesh M_d, we first compute a value $a_i > 0$, which measures the density of the mesh around x_i. We define a_i as the sum of the areas of the triangles surrounding x_i. To make the a_i's relative to the density of the mesh but not the geometric size, we always resize the mesh into the box $[-3, 3]^3$. This value a_i is used as control value for defining adaptor:

$$a(x) = \sum_{i=1}^{m} a_i\phi_i. \tag{5.1}$$

Hence, $a(x)$ is a function in the finite element space $V_{M(0)}$. Note that since $a(x)$ is not a constant any more, it is involved in the integration in computing the stiffness matrix $\mathscr{M}_0$ and $\mathscr{M}_1$. Since $a(x) \in V_{M(0)}$, it is C^2, except at the extraordinary vertices, where it is C^1. However, $a(x)$ may also be noisy, since it is computed from the noisy data. To obtain a smoother $a(x)$, we smooth repeatedly the control value a_i at the vertex x_i by the following rule:

$$a_i^{(k+1)} = (1 - n_i l_i)a_i^{(k)} + l_i \sum_{j=1}^{n_i} a_j^{(k)}, \tag{5.2}$$

where $a_i^{(0)} = a_i$ for $i = 1, \ldots, m$, $a_j^{(k)}$ in the sum are the control values at the one-ring neighbor vertices of x_i, n_i is the valence of x_i, l_i and $a(n_i)$ are given as follows:

$$l_i = \frac{1}{n_i + 3/8a(n_i)}, \quad a(n_i) = \frac{1}{n_i}\left[\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n_i}\right)^2\right].$$

The smoothing rule (5.2) is in fact for computing the limit value of Loop's subdivision (see paper [12, pp. 41–42]) applying to the control values $a_i^{(k)}$ at the vertices. In our examples, we apply this rule three times. Experiments show that even more times of smoothing of a_i are not harmful, but the influence to the evolution results are minor. The smoothing effect of (5.2) could be seen by rewriting it in the following form:

$$\frac{a_i^{(k+1)} - a_i^{(k)}}{n_i l_i} = \frac{1}{n_i}\sum_{j=1}^{n_i}(a_j^{(k)} - a_i^{(k)}).$$

The left-handed side could be regarded as the result of applying the *forward Euler method* to the function $a_i(t)$, the right-handed side is the umbrella operator [3]. Hence, (5.2) is a discretization of the equation $\partial D/\partial t = \Delta D$. Since $n_i l_i < 1$, the stability criterion for (5.2) is satisfied.

Fig. 1 is used to illustrate the difference between the identity adaptor evolution and the nonidentity adaptor evolution. The top figure shows the input mesh, the next one is the result of the identity adaptor evolution. Comparing this to the bottom figure, which is the result of the nonidentity adaptor evolution, many detailed features on the back and the snout of the crocodile are preserved by the adaptive approach. Furthermore, the large features of the identity adaptor evolution (compare the tails of the crocodiles) are less fairer than that of the nonidentity adaptor, even though the detailed features are already over-faired.

5.2. Defining the homogenizer $h(x)$

The homogenizer $h(x)$ is defined in a similar way as $a(x)$. We first define a number $h_i > 0$ for vertex x_i. Here h_i is defined as the sum of all the areas of triangles around x_i. Then we scale and translate h_i so that $\max_i h_i \in [0, 1]$ by $(h_i - h_{\min})/(h_{\max} - h_{\min})$, where $h_{\min} = \min_i\{h_i\}$ and $h_{\max} = \max_i\{h_i\}$. Then define $h(x) = \sum_i h_i \phi_i(x)$. Fig. 3 illustrates such an effect for a simple input mesh on the left. Two fairing results, after 114 fairing iteration ($\tau = 0.01$), are presented on the right with a homogenizer $h(x)$ and without the homogenizer, respectively.

5.3. Defining the interpolator $\omega_I(x)$

We define $\omega_I(x)$ as $\omega_I(x) = \sum_i \omega_i \phi_i(x)$, where ω_i is defined by

$$\omega_i = \begin{cases} W, & x_i \text{ is an interpolatory vertex,} \\ 0, & \text{otherwise,} \end{cases}$$

where $W > 0$ is large number. The larger of W, the closer of the surface to the interpolatory vertices. The indices of the interpolating vertices can be specified by users. The default choice in our implementation is by the magnitude of curvature at the vertices. If one of the principal curvatures of a vertex is larger than a specified value, then we interpolate that points. Fig. 4 illustrates the

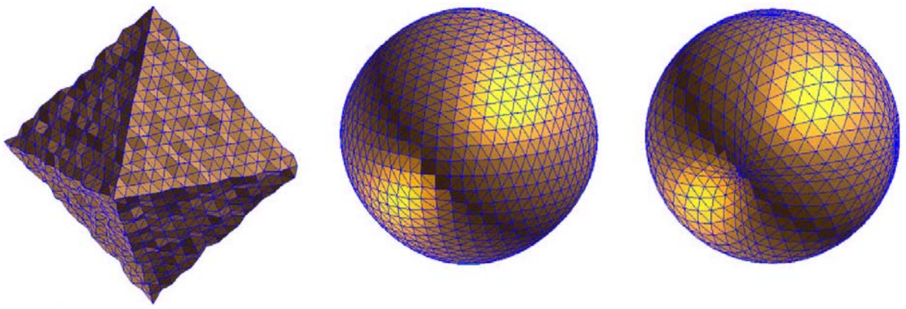
Fig. 3. The left figure is the initial mesh. The right two figures are the faired meshes after 114 fairing iterations with timestep $\tau = 0.01$ and with a homogenizer $h(x)$ and without the homogenizer, respectively. The ratios of minimal and maximal areas of the triangles are 0.831 and 0.000865, respectively.

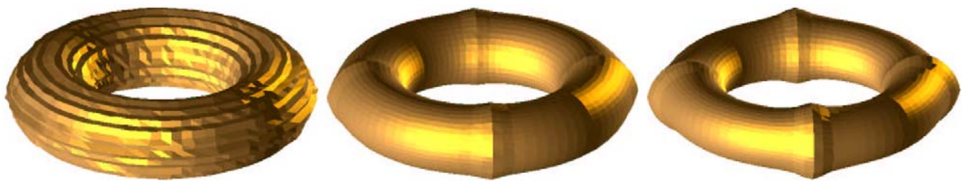
Fig. 4. The left figure shows the initial noisy mesh with some vertices are specified as interpolatory. The interpolatory vertices are those which lie approximately on either the xz-plane or the yz-plane. The middle and right figures are the faired meshes after 7 and 14 fairing iterations, respectively, with timestep $\tau = 0.001$ and with a nonzero $\omega_I(x)$.

interpolating effect for an input mesh on the left with some specified interpolatory vertices. Fairing results, after 7 and 14 iterations ($\tau = 0.001$), are presented in the middle and right, respectively, with $W = 110.0$. Fig. 5 shows the difference of the effects caused by nonzero $\omega_I(x)$ and zero $\omega_I(x)$, for an input mesh on the left with a default choice of the interpolatory vertices. Fairing results, after 3 iterations ($\tau = 0.0025$), are presented in the middle and right with $W = 300.0$ and $W = 0.0$, respectively.

6. Conclusions

We have proposed a simple approach in solving the diffusion PDE by the finite element discretization in the spatial direction and the semi-implicit discretization in the time direction, aiming at solving the under-fairing/over-smooth problem, the singularity problem and the nonsteady solution problem. The implementation shows that the proposed scheme works very well. Currently we are conducting theoretical research on how the parameter ω_A in (3.8) affect the error bound between the faired surface and the initial surface.

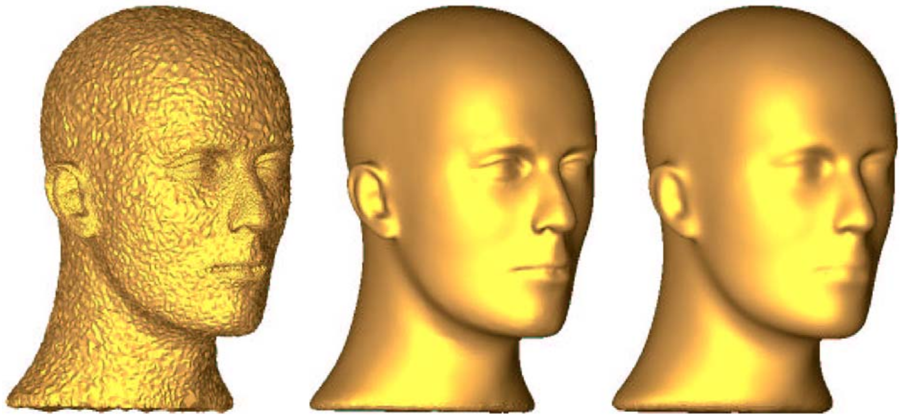

Fig. 5. The left figure represents the initial mesh with a default choice of the interpolatory vertices. The middle and right figures are the faired meshes after 3 fairing iterations with timestep $\tau = 0.0025$ and with a nonzero $\omega_I(x)$ and a zero $\omega_I(x)$, respectively.

References

[1] C. Bajaj, G. Xu, Anisotropic diffusion of subdivision surfaces and functions on surfaces, ACM Trans. Graphics 22 (1) (2003) 4–32.

[2] U. Clarenz, U. Diewald, M. Rumpf, Anisotropic geometric diffusion in surface processing, Proceedings of Viz2000, IEEE Visualization, Salt Lake City, Utah, 2000, pp. 397–505.

[3] M. Desbrun, M. Meyer, P. Schröder, A.H. Barr, Implicit fairing of irregular meshes using diffusion and curvature flow, SIGGRAPH99, 1999, pp. 317–324.

[4] M. Desbrun, M. Meyer, P. Schröder, A.H. Barr, Discrete differential-geometry operators in n*D*, http://www.multires.caltech.edu/pubs/, Caltech, 2000.

[5] M. do Carmo, Riemannian Geometry, Boston, 1992.

[6] G. Greiner, Variational design and fairing of spline surface, Comput. Graphics Forum 13 (1994) 143–154.

[7] B. Harr Romeny (Ed.), Geometry Driven Diffusion in Computer Vision. Kluwer, Boston, MA, 1994.

[8] H. Hoppe, T. DeRose, T. Duchampi, J. McDonald, W. Stuetzle, Mesh optimization, Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH93, Anaheim, 1993, pp. 19–26.

[9] A. Hubeli, M. Gross, Fairing of non-manifolds for visualization, Proceedings of Viz2000, IEEE Visualization, Salt Lake City, Utah, 2000, pp. 407–414.

[10] L. Kobbelt, S. Campagna, J. Vorsatz, H.-P. Seidel, Interactive multi-resolution modeling on arbitrary meshes, SIGGRAPH98, 1998, pp. 105–114.

[11] L. Kobbelt, T. Hesse, H. Prautzsch, K. Schweizerhof, Iterative mesh generation for FE-computation on free form surfaces, Eng. Comput. 14 (1997) 806–820.

[12] C.T. Loop, Smooth subdivision surfaces based on triangles, Master's Thesis. Technical Report, Department of Mathematices, University of Utah, 1978.

[13] J.L. Mallet, Discrete smooth interpolation in geometric modelling, Comput. Aided Des. 24 (4) (1992) 178–191.

[14] H. Moreton, C. Sequin, Functional optimization for fair surface design, ACM Computer Graphics (SIGGRAPH'92 Proceedings) (1992) 167–176.

[15] P. Perona, J. Malik, Scale space and edge detection using anisotropic diffusion, IEEE Trans. Pattern Anal. Math. Intel. 12 (1990) 629–639.

[16] T. Preußer, M. Rumpf, An adaptive finite element method for large scale image processing, J. Visual Comm. Image Repres. 11 (2000) 183–195.

[17] S. Rosenberg, The Laplacian on a Riemannian Manifold, Cambridge University Press, Cambridge, 1997.

[18] N. Sapidis, Designing fair curves and surfaces, SIAM, Philadelphia, 1994.
[19] G. Sapiro, Geometric Partial Differential Equations and Image Analysis, Cambridge University Press, Cambridge, 2001.
[20] J. Stam, Fast evaluation of loop triangular subdivision surfaces at arbitrary parameter values. SIGGRAPH '98 Proceedings, CD-ROM supplement, Orlando, 1998.
[21] J. Warren, Subdivision Methods for Geometric Design: A Constructive Approach, Morgan Kaufmann, 2002.
[22] J. Weickert, Anisotropic Diffusion in Image Processing, B.G. Teubner, Stuttgart, 1998.
[23] W. Welch, A. Witkin, Variational surface modeling, Comput. Graphics 26 (1992) 157–166.
[24] W. Welch, A. Witkin, Free-form shape design using triangulated surfaces, SIGGRAPH '94 Proceedings, Vol. 28, Orlando, July 1994, pp. 247–256.