Adaptive Surfaces Fairing by Geometric Diffusion

Chandrajit L. Bajaj[1]
Department of Computer Science, University of Texas, Austin
TX 78712, USA

Guoliang Xu[2]
The Institute of Computational Mathematics
Chinese Academy of Sciences, Beijing, China

In triangulated surface meshes, there are often very noticeable size variances (the vertices are distributed unevenly). The presented noise of such surface meshes is therefore composite of vast frequencies. In this chapter, we solve a diffusion partial differential equation numerically for noise removal of arbitrary triangular manifolds using an adaptive time discretization. The proposed approach is simple and is easy to incorporate into any uniform timestep diffusion implementation with significant improvements over evolution results with the uniform timesteps. As an additional alternative to the adaptive discretization in the time direction, we also provide an approach for the choice of an adaptive diffusion tensor in the diffusion equation.

1 Introduction

The solution for triangular surface mesh denoising (fairing) is achieved 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 (see [1, 10, 11, 17]). It is well known that the solution of heat equation $\partial_t \rho - \Delta\rho = 0$, based on the Laplacian Δ,

[1] Supported in part by NSF grants ACI-998297 and KDIDMS-9873326
[2] Support in part by NSF of China and National Innovation Fund 1770900, Chinese Academy of Sciences

at time T for a given initial image ρ_0 is the same as taking a convolution of the Gauss filter $G_\sigma(x) = \frac{1}{2\pi\sigma}\exp\left(-\frac{|x|^2}{2\sigma^2}\right)$ with standard deviation $\sigma = \sqrt{2T}$ and image ρ_0. Taking the convolution of $G_\sigma(x)$ 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 in [4, 5] and shown to be very effective even for higher-order methods [3]. The counterpart of the Laplacian Δ is the Laplace-Beltrami operator Δ_M (see [7]) for a surface M. However, 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.

The bottom row of Fig 1 illustrates this under-fairing and over-fairing effects. For the input mesh on the top-left, three fairing results are presented at three time scales. The first figure exhibits the under-fairing for the head. The last figure exhibits the over-fairing for the ears, eyes, lips and nose. Hence, a phenomena that often appears for the triangular surface mesh denoising 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 [3, 4]. However, this is far from satisfactory. The aim of this chapter is to overcome the under-fairing and over-fairing dilemma in solving the diffusion equation.

There are several situations where the produced triangular surface meshes have varying triangle density. One typical case is geometric modeling, where the detailed structures are captured by several small triangles while the simpler shapes are represented by fewer large ones. We may call such triangular meshes as *feature-adaptive*. Another case is the results of physical simulation, in which the researcher is interested in certain regions of the mesh. In these regions, accurate solutions are desired, and quite often, finer meshes are used. For example, in the acoustic pressure simulation [15], the interesting region is the ear canal for human hearing. Hence, more accurate and

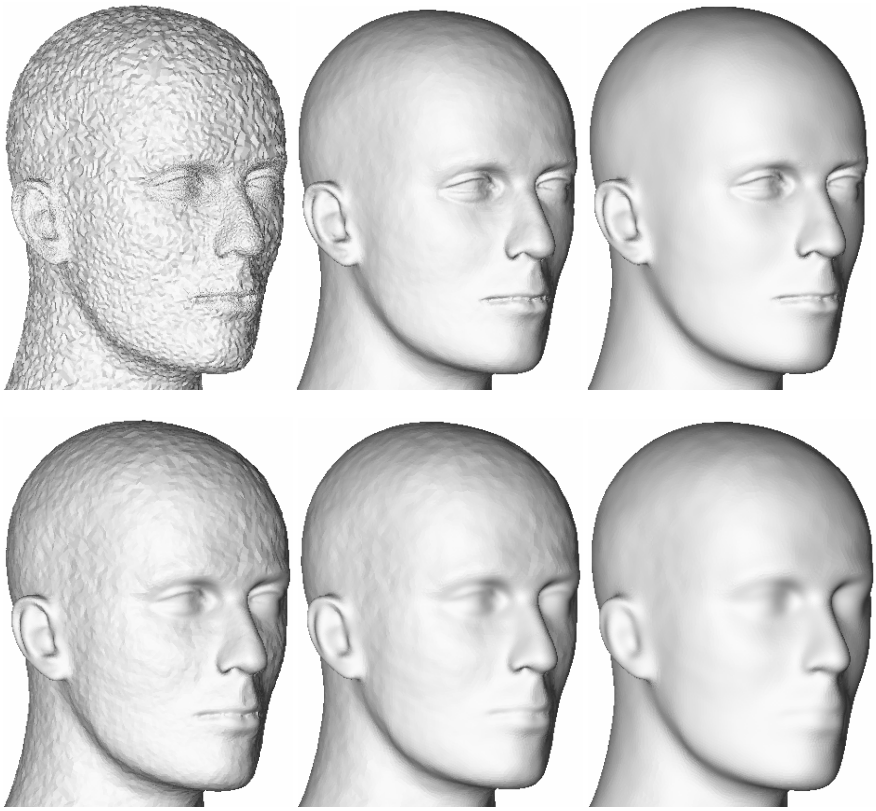


Fig. 1. *The left figure in the top row shows the initial geometry mesh. The top-middle and top-right figures illustrate the results of the adaptive timestep smoothing after 2 and 4 fairing steps with* $\tau = 0.016$. *The three figures in the bottom row are the results of the timestep* $t = 0.001$ *smoothing after 1, 2 and 4 fairing steps, respectively.*

finer meshes are used there. We call such meshes as *error-adaptive*. One additional case arises from the multiresolution representation of surfaces, for example using wavelet transforms or direct mesh simplifications. Each resolution of the representation is a surface mesh that approximates the highest resolution surface. The approximation error is usually adaptive and can vary over the entire surface. For instance, the mesh simplification scheme in [2], which is driven by the surface normal variation, results in meshes that are both feature-adaptive and error-adaptive.

Previous Work. For PDE based surface fairing or smoothing, there are several methods that have been proposed (see [3, 4, 5, 6]) recently. Desbrun et al in [5, 6] 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 in [4] generalize the Laplacian to the Laplace-Beltrami operator Δ_M, and use linear finite elements to discretize the equation. In [3], the problem is reformulated for 2-dimensional Riemannian manifold embedded in R^k aiming at smooth 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 non-adaptivity. All of them use uniform timesteps. Hence they quite often suffer from under-fairing or over-fairing problems.

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, we present a discretization in the time direction, which is mesh adaptive. We use a timestep $T(x)$, which depends on the position x of the surface. The part of the surface, that is coarse, uses larger $T(x)$. The idea is simple and it is easy to incorporate it into any uniform timestep diffusion implementation. The improvements achieved over the evolution results with uniform timestep are significant. The top row of Fig 1 shows this adaptive time evolution improvement over the uniform timestep evolution results, shown in the bottom row.

The middle and right figures in the top row are the smoothing results of the mesh on the left top after 2 and 4 fairing steps, respectively. As an alternative to the adaptive discretization in time direction, we also provide an approach for the adaptive choice of the diffusion tensor in the diffusion PDE equation.

The remaining of the chapter is organized as follows: Section 2 summarizes the diffusion PDE model used, followed by the discretization section 3. 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 adaptive. The conclusion section 4 provides examples showing the superiority of the adaptive scheme.

2 Geometric Diffusion Equation

We shall solve the following nonlinear system of parabolic differential equations (see [3, 4]):

$$\partial_t x(t) - \Delta_{M(t)} x(t) = 0, \tag{1}$$

where $\Delta_{M(t)} = div_{M(t)} \circ \nabla_{M(t)}$ 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. This equation is a generalization of heat equation $\partial_t \rho - \Delta \rho = 0$ to surfaces, where Δ is the Laplacian. To enhance sharp features, a *diffusion tensor* D, acting on the gradient, has been introduced (see [3, 4]). Then (1) becomes

$$\partial_t x(t) - div_{M(t)}(D \nabla_{M(t)} x(t)) = 0. \tag{2}$$

The diffusion tensor $D := D(x)$ is a symmetric and positive definite operator from $T_x M$ to $T_x M$. Here $T_x M$ is the tangent space of M at x. The detailed discussion for choosing the diffusion tensor can be found in [3, 4] for enhancing sharp features. In this chapter, we do not address the problem of enhancing sharp features. However, we shall use a scalar diffusion tensor for achieving an adaptive diffusion effect. The divergence $div_M \psi$ for a vector field $\psi \in TM$ is defined as the dual operator of the gradient (see [12]):

$$(div_M v, \phi)_M = -(v, \nabla_M \phi)_{TM}, \quad \forall \phi \in C_0^\infty(M), \tag{3}$$

where $C_0^\infty(M)$ is a subspace of $C^\infty(M)$, whose elements have compact support. TM is the tangent bundle, which is a collection of all the tangent spaces. The inner product $(\phi,\psi)_M$ and $(u,v)_{TM}$ are defined by the integration of $\phi\psi$ and $u^T v$ over M, respectively. The gradient of a smooth function f on M is given by

$$\nabla_M f = [t_1, t_2] G^{-1} \left[\frac{\partial (f \circ x)}{\partial \xi_1}, \frac{\partial (f \circ x)}{\partial \xi_2} \right]^T, \tag{4}$$

where

$$G^{-1} = \frac{1}{\det G} \begin{bmatrix} g_{22} & -g_{12} \\ -g_{21} & g_{22} \end{bmatrix}, \quad G = \begin{bmatrix} g_{22} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}, \quad g_{ij} = \left(\frac{\partial}{\partial \xi_1} \right)^T \frac{\partial}{\partial \xi_2},$$

$x(\xi_1, \xi_2)$ is a local parameterization of the surface. G is known as the first fundamental form. For a vector field $X = \sum_{i=1}^{2} X^i \frac{\partial}{\partial \xi_i} \in TM$, an explicit expression for the divergence is given by (see [8], page 84)

$$div_M X = \frac{1}{\sqrt{\det G}} \sum_{i=1}^{2} \frac{\partial}{\partial \xi_i} (\sqrt{\det G} X^i).$$

Then it is easy to derive that

$$div_M (h \nabla_M f) = (\nabla_M f)^T \nabla_M h + h \Delta_M f, \tag{5}$$

where f, h are smooth functions on M. From (4), (5) and the fact that $\Delta_M x = 2H(x)n(x)$, we could rewrite (2) as

$$\partial_t x(x) = \nabla_M D(x) + 2D(x)H(x)n(x), \tag{6}$$

where $H(x)$ and $n(x)$ are the mean curvature and the unit normal of M, respectively. Equation (6) implies that the motion of the surface $M(t)$ can be decomposed into two parts: One is the tangential displacement caused by $\nabla_M D(x)$, and the other is the normal displacement (mean curvature motion) caused by $2D(x)H(x)n(x)$.

Using (3), the diffusion problem (2) could be reformulated into the following variational form

$$\begin{cases} \textit{Find a smooth } x(t) \textit{ such that} \\ (\partial_t x(t), \theta)_{M(t)} + (D \nabla_{M(t)} x(t), \nabla_{M(t)} \theta)_{TM(t)} = 0, \\ M(0) = M, \end{cases} \tag{7}$$

for any $\theta \in C_0^\infty(M(t))$. This variational form is the starting point for the discretization.

We already know that equation (1) describes the mean curvature motion. Its regularization effect could be seen from the following equation (see [4, 13])

$$\frac{d}{dt} Area(M(t)) = -\int_{M(t)} H^2 dx, \quad \frac{d}{dt} Volume(M(t)) = -\int_{M(t)} H \, dx, \tag{8}$$

where $Area(M(t))$ and $Volume(M(t))$ are the area of $M(t)$ and volume enclosed by $M(t)$, respectively, H is the mean curvature. 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, have larger curvatures, which very possibly be over-faired.

3 Discretization

We discretize equation (7) 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 (7) at $t = T(x(0))$. Using a semi-implicit Euler scheme, we have the following time direction discretization:

$$\begin{cases} Find \;\; a \;\; smooth \;\; x(t) \;\; such \;\; that \\ \left(\dfrac{x(T) - x(0)}{T}, \theta \right)_{M(0)} + (D\nabla_{M(0)} x(T), \nabla_{M(0)} \theta)_{TM(0)} = 0, \end{cases} \tag{9}$$

for any $\theta \in C_0^\infty(M(t))$. If we want to go further along the time direction, we could treat the solution at $t = T(x)$ as the initial value and repeat the same process. Hence, we consider only one time step in our analysis.

3.1 Spatial Discretization

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

$$\mathrm{T} = \left\{ (\xi_1, \xi_2) \in R^2 : \xi_1 \geq 0, \xi_2 \geq 0, \xi_1 + \xi_2 \leq 1 \right\}.$$

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

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

The integration on triangle T is computed adaptively by numerical methods.

Let M_d be the given initial triangular mesh, $x_i, i=1,\cdots,m$, 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 [3] for detailed description of this). For simplicity, we call it the *Loop's basis*.

Loop's Subdivision and Finite Element Function Space

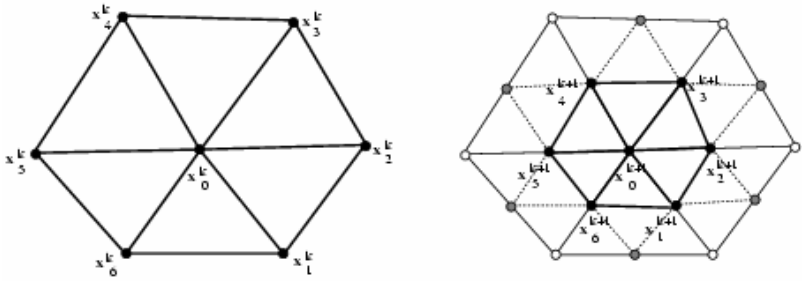


Fig. 2. *Refinement of triangular mesh around a vertex*

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 $i = 1, \cdots, n$ (see Fig 2), 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

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

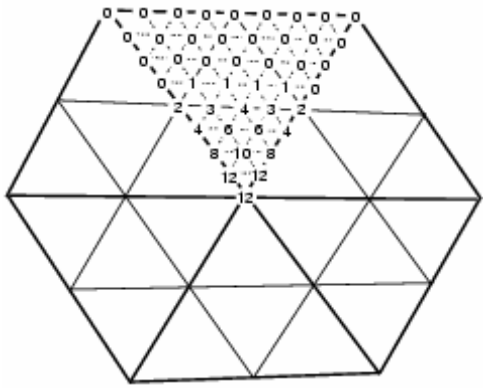
where index i is to be understood modulo n. The old vertices get new positions according to

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

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 to the later case as *extraordinary*.



F.g. 3. *The definition of base ϕ_i : The quartic Bezier coefficients (each has a factor 1/24). The Bezier coefficients on the five macro-triangles are obtained by rotating the top macro-triangle around the center to the other five positions.*

Let e_j, $j = 1, \cdots, m_i$ be the 2-ring neighborhood elements. Then if e_j is regular (meaning its three vertices have valence 6), explicit Box-spline expressions exist (see [14, 16]) for ϕ_i on e_j. Using these explicit Box-spline expressions, we derive the BB-form ex-

pression for the basis functions ϕ_i (see Fig 3). These expressions could be used to evaluate ϕ_i in forming the linear system (11). 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 by Stam [14].

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 [3].

Let $V_{M(0)}$ be the finite dimensional space spanned by the Loop's basis functions $\{\phi_i\}_{i=1}^m$. Then $V_{M(0)} \subset C^1(M(0))$. Let

$$x(0) = \sum_{i=1}^{m} x_i(0)\phi_i \in M(0), \;\; x(T) = \sum_{i=1}^{m} x_i(T_i)\phi_i, \;\; \theta = \phi_j.$$

Then equation (9) is discretized in $V^3_{M(0)}$ as

$$\begin{aligned} &\sum_{i=1}^{m} (x_i(T_i) - x_i(0))(T_i^{-1}\phi_i, \phi_j)_{M(0)} + \\ &\sum_{i=1}^{m} x_i(T_i)(D\nabla_{M(0)}\phi_i, \nabla_{M(0)}\phi_j)_{TM(0)} = 0 \end{aligned} \tag{10}$$

for $j = 1, \cdots, m$, where $x_i(0) := x_i$ is the i-th vertex of the input mesh M_d, $T_i = T(y_i(0))$ and $y_i(0)$ is a surface point corresponding to vertex $x_i(0)$. Equation (10) is a linear system for unknowns $x_i(T_i)$.

3.2 Adaptive Timestep

We first use adaptive timesteps to achieve the adaptive evolution effect. In this case the diffusion tensor D is chosen to be identity, but $T(x)$ is not a constant function. Now (10) can be written in the following matrix form:

$$(S+L)X(T) = MX(0), \tag{11}$$

where $X(T) = [x_1(T_1), \cdots, x_m(T_m)]$, $X(0) = [x_1(0), \cdots, x_m(0)]$ and

$$\begin{aligned} S &= \left((T_i^{-1}\phi_i, \phi_j)_{M(0)}\right)_{i,j=1}^{m}, \\ L &= \left((D\nabla_{M(0)}\phi_i, \nabla_{M(0)}\phi_j)_{TM(0)}\right)_{i,j=1}^{m}. \end{aligned} \tag{12}$$

Note that both S and L are symmetric. Since ϕ_1, ϕ_2, … , ϕ_m are linearly independent and have compact support, S is sparse and positive definite. Similarly, L is symmetric and nonnegative definite. Hence, $S + L$ is symmetric and positive definite.

The coefficient matrix of system (11) 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.

Defining adaptive timesteps. Now we illustrate how $T(x)$ is defined. At each vertex x_i of the mesh M_d, we first compute a value $d_i > 0$, which measures the density of the mesh around x_i. We propose two approaches for computing it:

1. d_i is defined as the average of the distance from x_i to its neighbor vertices.
2. d_i is defined as the sum of the areas of the triangles surrounding x_i.

To make the $d_i's$ relative to the density of the mesh but not the geometric size, we always resize the mesh into the box $[-3,3]^3$. The experiments show that both approaches work well, and the evolution results have no significant difference. This value d_i is used as control value for defining timestep that is the same as defining the surface point:

$$T(x) = \tau D(x),\ \ D(x) = \sum_{i=1}^{m} d_i \phi_i, \tag{13}$$

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

$$d_i^{(k+1)} = (1 - n_i l_i) d_i^{(k)} + l_i \sum_{j=1}^{n_i} d_j^{(k)}, \tag{14}$$

where $d_i^{(0)} = d_i$ for $i = 1, \cdots, m,\ \ d_j^{(k)}$ 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:

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

The smoothing rule (14) is in fact for computing the limit value of Loop's subdivision (see [9], pp 41-42) applying to the control values $d_i^{(k)}$ at the vertices. In our examples, we apply this rule three times. Experiments show that even more times of smoothing of d_i are not harmful, but the influence to the evolution results are minor. The smoothing effect of (14) could be seen by rewriting it in the following form

$$\frac{d_i^{(k+1)} - d_i^{(k)}}{n_i l_i} = \frac{1}{n_i}\sum_{j=1}^{n_i}\left(d_j^{(k)} - d_i^{(k)}\right).$$

The left-handed side could be regarded as the result of applying the *forward Euler method* to the function $d_i(t)$, the right-handed side is the umbrella operator (see [5]). Hence, (14) is a discretization of the equation $\frac{\partial D}{\partial t} = \Delta D$. Since $n_i l_i < 1$, the stability criterion for (14) is satisfied.

A different view of adaptive timestep approach

Consider the following diffusion PDE

$$\partial_t x(t) - D(x)\Delta_{M(t)} x(t) = 0, \tag{15}$$

where $D(x)$ is a function defined by (13). Again, equation (15) describes the mean curvature motion with a compression factor $D(x)$. If we use a semi-implicit Euler scheme to discretize the equation with constant timestep τ, we could arrive at the same linear system as (11). Hence, solving the equation (1) with an adaptive timestep $\tau D(x)$ is equivalent to solving equation (15) with a uniform timestep τ. But (15) may be easier to handle in the theoretical analysis.

3.3 Uniform Timestep and Adaptive Diffusion Tensor

Now we use uniform timestep τ but a non-identity diffusion tensor $D(x)$. This $D(x)$ is the same as the one defined in (13), but we should regard it as $D(x)I$, where I is the identity diffusion tensor. The discretized equation (10) then becomes

$$\begin{aligned} &\sum_{i=1}^{m} (x_i(\tau) - x_i(0))(\phi_i, \phi_j)_{M(0)} + \\ &\sum_{i=1}^{m} x_i(\tau)(\tau D \nabla_{M(0)}\phi_i, \nabla_{M(0)}\phi_j)_{TM(0)} = 0. \end{aligned} \tag{16}$$

From this, a similar linear system as (11) is obtained with

$$\begin{aligned} S &= \left((\phi_i, \phi_j)_{M(0)}\right)_{i,j=1}^{m}, \\ L &= \left((\tau D \nabla_{M(0)}\phi_i, \nabla_{M(0)}\phi_j)_{TM(0)}\right)_{i,j=1}^{m}. \end{aligned} \tag{17}$$

We know that $D(x)$ is a smooth positive function that characterizes the density of the surface mesh. The effect of this diffusion tensor is suppressing the gradient where the mesh is dense, and hence slows down the evolution speed. Comparing equation (11) with equation (16), we find that they are similar (since $\tau D = T$), though not equivalent. Indeed, if $D(x)$ is a constant on each triangle of M, then they are equivalent. In general, $D(x)$ is not a constant, but approximately a constant on each triangle, hence the observed behavior of (11) and (16) are often similar. The last two figures in Fig 4 exhibit this similarity, where the third and fourth figures are the evo-

lution results using an adaptive timestep and an adaptive diffusion tensor, respectively. Since the results of the two adaptive approaches are very close, in the other examples provided in this chapter, we use only the adaptive timestep approach.

Homogenization Effect of D

It follows from (6) that the non-constant diffusion tensor $D(x)$ causes tangential displacement of the vertices. For the diffusion tensor $D(x)$ defined in the last sub-section, we know 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 $D(x)$ is small on the triangle and larger elsewhere. This implies that the gradient of $D(x)$ 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 $D(x)$ is near a constant. Then the tangential displacement is minor. Hence, the adaptive diffusion tensor we use has homogenizing effect. Such an effect is nice and important, as it avoids producing collapsed or tiny triangles in the faired meshes.

3.4 Algorithm Summary

For a given initial mesh, stopping control threshold values $\varepsilon_i > 0$, $i = 1,2$ and $\tau > 0$, the adaptive timestep evolution algorithm could be summarized via the following pseudo-code:

```
Compute function and derivative values of φ_i on the
integration points;
  do {
     Compute d_i;
     Smooth d_i by (14);
     Compute matrices S and L by (12);
     Solve linear system (11);
     Compute H(t);
  }  while (none of (18)-(19) is satisfied);
```

Note that the evolution process does not change the topology of the mesh. Hence the basis functions could be computed before the multiple iterations.

We use two of the three stopping criteria proposed in [3] for terminating the evolution process: Let

$$H(t) = \int_{M(t)} \| H(t,x) \|^2 \, dx \Big/ \int_{M(0)} \| H(0,x) \|^2 \, dx,$$

where $H(t,x)$ is the mean curvature vector at the point x and time $tD(x)$. The stopping criteria are

$$| H'(t) | \leq \varepsilon_{1,} \text{ or} \tag{18}$$

$$H(t) \leq \varepsilon_2. \tag{19}$$

where ε_i are user specified control constants, $H'(t)$ is computed by divided differences.

4 Summary

We have proposed two simple adaptive approaches in solving the diffusion PDE by the finite element discretization in the spatial direction and the semi-implicit discretization in the time direction, aiming to solving the under-fairing/over-smooth problems that beset the uniform diffusion schemes. The implementation shows that the proposed adaptive schemes work very well.

Fig. 4. *The left figure is the initial geometry mesh. The second figure is the faired mesh after 3 fairing iterations with uniform timestep t = 0.0011. The third and fourth figures are the faired meshes after 3 fairing iterations with adaptive timestep and alternatively adaptive diffusion tensor with uniform timestep* τ *= 0.025, respectively.*

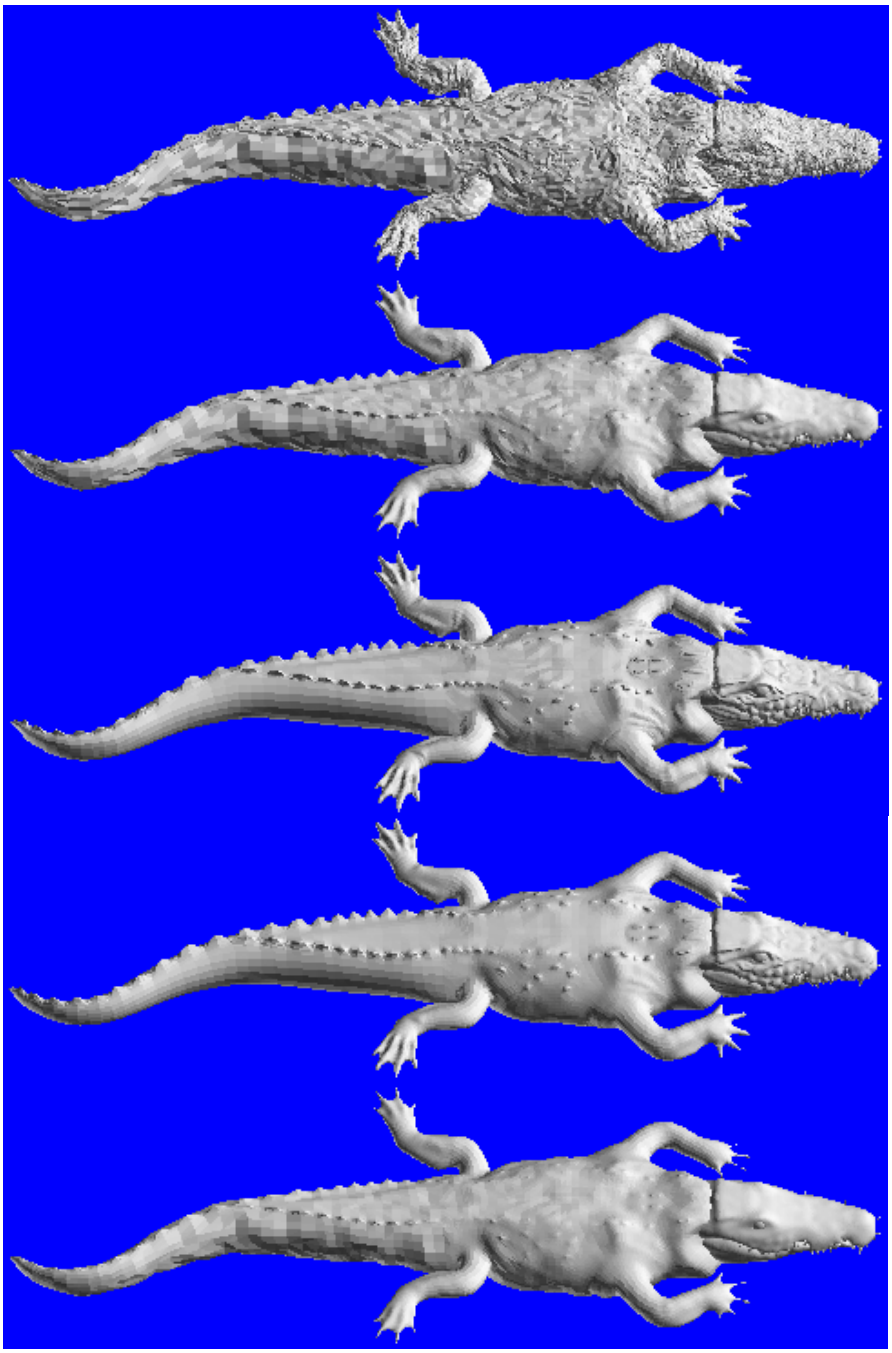


Fig. 5. *The top figure is the initial geometry mesh. The second and the third figures are the faired meshes after 2 and 4 fairing iterations with uniform timestep t = 0.0001. The last two are the faired meshes after 2 and 4 fairing iterations with adaptive timestep and* $\tau = 0.0016$.

Figure 4 and Fig 5 are used to illustrate the difference between the uniform timestep evolution and the adaptive timestep evolution. Since the adaptive timestep is not uniform, we cannot compare the evolution results for the same time. The comparing criterion we adopted here is we evolve the surface, starting from the same input, to arrive at similar smoothness for the rough/detailed features and compare the detailed/rough features. In Fig 4, the left figure is the input mesh, the second figure uses uniform timestep, the third and fourth figures use a adaptive timestep and a adaptive diffusion tensor with a uniform timestep, respectively. Comparing the three smoothing results, we can see that the large features look similar, but the toes of the foot are very different. The evolution results of the adaptive timestep and the adaptive diffusion tensor are much more desirable. Figure 5 exhibits the same effect. The top figure shows the input mesh, the next two are the results of the uniform timestep evolution. Comparing these to the bottom two figures, which are the results of the adaptive timestep 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 uniform timestep evolution (compare the tails of the crocodiles) are less fairer than that of the adaptive timestep evolution, even though the detailed features are already over-faired.

References

1. Ed B. Harr Romeny. Geometry Driven Diffusion in Computer Vision. Boston, MA: Kluwer, 1994.
2. C. Bajaj and G. Xu. Smooth Adaptive Reconstruction and Deformation of Free-Form Fat Surfaces. TICAM Report 99-08, March, 1999, Texas Institute for Computational and Applied Mathematics, The University of Texas at Austin, 1999.
3. C. Bajaj and G. Xu. Anisotropic Diffusion of Noisy Surfaces and Noisy Functions on Surfaces. TICAM Report 01-07, February 2001, Texas Institute for Computational and Applied Mathematics, The University of Texas at Austin, 2001,.

4. U. Clarenz, U. Diewald, and M. Rumpf. Anisotropic Geometric Diffusion in Surface Processing. In Proceedings of Viz2000, IEEE Visualization , pages 397-505, Salt Lake City, Utah, 2000.
5. M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow. SIGGRAPH99, pages 317-324, 1999.
6. M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Discrete Differential-Geometry Operators in nD, http://www.multires.caltech.edu/pubs/, 2000.
7. M. do Carmo. Riemannian Geometry. Boston, 1992.
8. J. Jost. Riemannian Geometry and Geometric Analysis, Second Edition. Springer, 1998.
9. C. T. Loop. Smooth subdivision surfaces based on triangles. Master's thesis. Technical report, Department of Mathematices, University of Utah, 1978.
10. P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. In IEEE Computer Society Workshop on Computer Vision, 1987.
11. T. Preuβer and M. Rumpf. An adaptive finite element method for large scale image processing. In Scale-Space Theories in Computer Vision, pages 232-234, 1999.
12. S. Rosenberg. The Laplacian on a Riemannian Manifold. Cambridge, Uviversity Press, 1997.
13. G. Sapiro. Geometric Partial Differential Equations and Image Analysis. Cambridge, University Press, 2001.
14. J. Stam. Fast Evaluation of Loop Triangular Subdivision Surfaces at Arbitrary Parameter Values. In SIGGRAPH '98 Proceedings, CD-ROM supplement, 1998.
15. T. F. Walsh. P Boundary Element Modeling of the Acoustical Transfer Properties of the Human Head/Ear. PhD thesis, Ticam, The University of Texas at Austin, 2000.
16. J. Warren. Subdivision method for geometric design, 1995.
17. J. Weickert. nisotropic Diffusion in Image Processing. B. G. Teubner Stuttgart, 1998.