Geometric Surface Smoothing via Anisotropic Diffusion of Normals

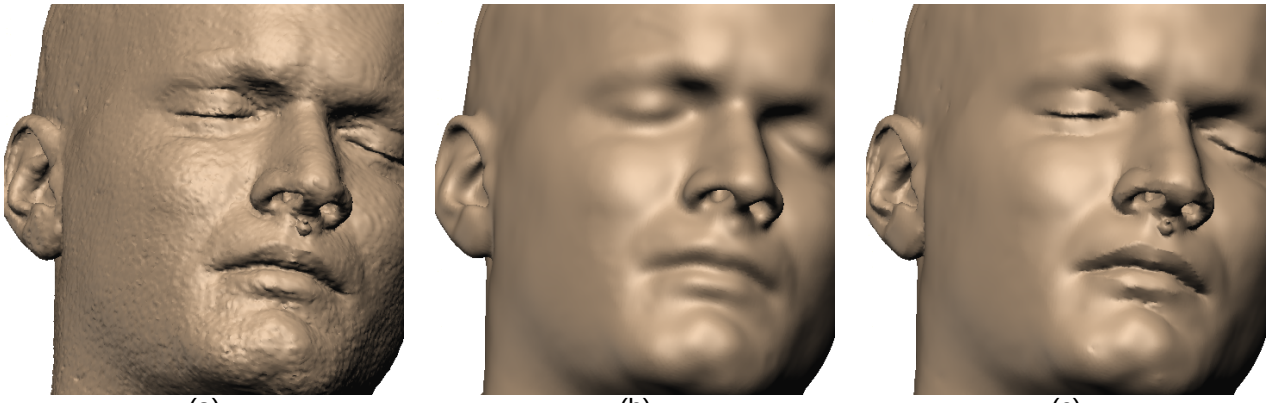
Tolga Tasdizen
School of Computing
Univ. of Utah

Ross Whitaker
School of Computing
Univ. of Utah

Paul Burchard
Dept. of Mathematics
UCLA

Stanley Osher
Dept. of Mathematics
UCLA

To appear in VIS'02



(a) (b) (c)

Figure 1: Processing results on the MRI head model: (a) original isosurface, (b) isotropic diffusion (intrinsic Laplacian of mean curvature flow), and (c) anisotropic diffusion. The small protrusion under the nose is a physical marker used for registration.

ABSTRACT

This paper introduces a method for smoothing complex, noisy surfaces, while preserving (and enhancing) sharp, geometric features. It has two main advantages over previous approaches to feature preserving surface smoothing. First is the use of level set surface models, which allows us to process very complex shapes of arbitrary and changing topology. This generality makes it well suited for processing surfaces that are derived directly from measured data. The second advantage is that the proposed method derives from a well-founded formulation, which is a natural generalization of anisotropic diffusion, as used in image processing. This formulation is based on the proposition that the generalization of image filtering entails filtering the normals of the surface, rather than processing the positions of points on a mesh.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid and object representations

Keywords: anisotropic diffusion, surface fairing, geometric surface processing, intrinsic Laplacian of curvature, level sets

1 INTRODUCTION

The fundamental principles of signal processing give rise to a wide range of useful tools for manipulating and transforming signals and images. The generalization of these principles to the processing of 3D surfaces has become an important problem in computer graphics, visualization, and vision. For instance, 3D range sensing technologies produce high resolution descriptions of objects, but they often suffer from noise. Medical imaging modalities such as MRI and CT scans produce large volumes of scalar or tensor measurements, but surfaces of interest must be extracted through some segmentation process or fitted directly to the measurements.

One of the most prevalent uses of image processing, is for *smoothing* or *denoising* images. Often denoising of images is done with a low-pass filter, which reduces noise, but also blurs sharp features and details, such as edges. The literature demonstrates a variety of nonlinear processes that reduce noise while preserving edges. Likewise, when we build *surfaces* from measured data, we would like to reduce the effects of noise on visualization or subsequent processing. However, the question of how to apply nonlinear image-smoothing processes to surfaces remains open.

In this paper, we focus on the generalization of anisotropic diffusion, a PDE-based, edge-preserving, image-smoothing technique, to surface processing. The proposed method offers two advantages over previous work on feature-preserving smoothing of surfaces. First is the use of level set surface models, which allows us to process very complex shapes of arbitrary and changing topology. This generality makes the method well suited for processing and visualizing surfaces that are derived directly from measured data. The second advantage is that the proposed method follows from a well-founded variational formulation, which is a natural generalization of anisotropic diffusion, as proposed by Perona and Malik [21] (P&M) for images. This formulation will alleviate the need for developing heuristics, which are sometimes used to achieve the qualitative effects of the P&M diffusion in the absence of a complete, mathematical generalization for surfaces [6]. Figure 1 demonstrates these concepts on an isosurface extracted from a MRI scan. The original isosurface is a good example of the kind of noisy, measured data with which this paper is concerned. Isotropic diffusion which behaves like a low pass filter on the surface normals is not particularly effective for denoising; the noise is removed but surface features are deformed or lost in the process. Anisotropic diffusion is a much better candidate for denoising as can be seen in Figure 1(c). Note that all of the surfaces in this paper are represented and processed volumetrically and rendered using the marching cubes algorithm [14].

The use of a level set formulation enables us to achieve a "black box" behavior, which is reflected in the nature of the results presented in this paper. Hence, the techniques presented in this paper offer a new set of capabilities that are especially interesting when processing measured data. Measured data can be acquired directly in volumetric form or acquired as a surface mesh and converted

into a volume [4]. In some applications, such as animation, models are manually generated by a designer and the parameterization is not arbitrary but is an important aspect of the geometric model. In these cases mesh-based processing methods offer a powerful set of tools, such as hierarchical editing [11], which are not yet possible with the proposed representation. However, in other applications, such as 3D segmentation and surface reconstruction [16, 32], the processing is data driven, surfaces can deform quite far from their initial shapes and change topology, and user intervention is not practical. Furthermore, when considering processes other than isotropic smoothing, such as nonlinear smoothing, the creation or sharpening of small features can exhibit noticeable effects of the mesh topology—features that are aligned with the mesh are treated differently than those that are not.

The proposed method relies on a novel technique for solving fourth-order flows on level set surfaces [19, 26]. Our strategy is to split the surface deformation into a two step process that (i) solves anisotropic diffusion on the normal map of the surface, and (ii) deforms the surface so that it fits the smoothed normals. This approach is based on the proposition that the natural generalization of image processing to surfaces is via the *surface normal vectors*. The variation of the normals has more intuitive meaning than the variation of points on the surface. A smooth surface is one that has smoothly varying normals and creases on a piecewise smooth surface appear as discontinuities in the normals. For instance, the definition of a crease becomes much more complicated and tied to the parameterization if the variation of points on the surface is used instead of the normals; see [27] for more details. In this light, the differences between surface processing and image processing are threefold. Normals live on a manifold (the surface) and cannot necessarily be processed using a flat metric, as is typically done with images. Normals are vector valued and constrained to be unit length; the processing techniques must accommodate this. Normals are coupled with the surface shape, and thus the normals should drag the surface along as their values are modified during processing. This general mechanism will also open the door of possibilities for mapping other types of image processing algorithms to surfaces.

The rest of this paper is organized as follows: Section 2 will present a brief overview of related work in the literature. The mathematical formulation for our approach will be discussed in Sec. 3. Examples of isotropic and anisotropic diffusion will be presented in Sec. 4. Section 5 will summarize the results of this paper and outline directions for future work. The details of the numerical implementation of our method is covered in Appendix A.

2 RELATED WORK

The majority of surface smoothing research has been in the context of *surface fairing* with the motivation of creating aesthetically pleasing surfaces using triangulated meshes. Surface fairing typically operates by minimizing a fairness or penalty function that favors smooth surfaces [17, 30]. Fairness functions can depend on the geometry of the surface or the parameterization. Geometric penalty functions make use of invariants such as principal curvatures, and therefore produce results that are not significantly affected by arbitrary decisions about the parameterization. The proposed work relies on geometric penalty functions.

One way to smooth a surface is to incrementally reduce its surface area. This can be accomplished by mean curvature flow (MCF) which is a second order geometric flow. MCF can reduce noise, but also has some unsatisfactory side effects, including the creation of singularities and shrinkage (see Sec. 3). A great deal of research has focused on modified second-order flows that produce better results than MCF. Using level set models, several authors have proposed smoothing surfaces by weighted combinations of principle curvatures. For instance, Whitaker [31] has proposed a nonlinear reweighting scheme that favors the smaller curvature and preserves cylindrical structures. Faugeraus [15] proposes a smoothing by the minimum curvature. A variety of other combinations have been proposed [24].

A similar set of curvature-based algorithms have been developed for surface meshes. For instance, Clarenz *et al.* [6] propose a modified MCF as an anisotropic diffusion of the surface. They threshold a weighted sum of the principle curvatures to determine the surface locations where edge sharpening is needed. Tangential displacement is added to the standard MCF at these locations for sharpening the edges. Although, this flow produces results that tend to preserve sharp features, it is not a strict generalization of P&M diffusion from images to surfaces. Another mesh-based modified MCF is proposed in [18] where a threshold on the mean curvature is used to stop over-smoothing. Anisotropic diffusion as a modified surface area minimization for height functions was proposed in [9]. Taubin proposes a "linear anisotropic Laplacian operator" for meshes that is based on a separate processing of the normals [29]. As with [6, 18], it is essentially a reweighting of the Laplacian. It is similar to our approach in the sense that surface positions are refitted to the processed normals; however, our approach uses level set surfaces that are refitted to the normals with a second order PDE as opposed to the least squares mesh fitting in [29].

These level set and mesh based methods are all modifications of curvature flows, and are therefore all second-order processes. Because they are based on reweightings of curvature, these methods always smooth the surface in one direction or another. They do not exhibit a sharpening of details, which is achieved by the P&M equation (for images) through an inverse diffusion process. Hence, these methods are not satisfactory generalizations of the P&M anisotropic diffusion equation.

The generalization of P&M to surfaces requires a high-order geometric flow. Using a variational framework, a useful second-order penalty function is *total curvature*

$$\int_S \kappa_1^2 + \kappa_2^2 \, dS, \tag{1}$$

where κ_1 and κ_2 are the principal curvatures. Minimizing (1) has been shown to deform surfaces into spheres [22], which are a steady state solution. Total curvature is a geometric (invariant) property of the surface. The mesh fairing approach of [30] which minimizes (1) involves fitting local polynomial basis functions to local neighborhoods for the computation of total curvature. The first variation of total curvature is intrinsic Laplacian of mean curvature (ILMCF), a fourth order equation. Instead of solving fourth order PDEs directly, Kobbelt *et al.* decouple the fourth order expression into a pair of second-order equations [25]. However, this approach works only for meshes, and relies on analytic properties of the steady-state solutions, $\Delta H = 0$, by fitting surface primitives that have those properties. Thus, it does not apply to other types of smoothing processes, such as those that minimize nonlinear feature-preserving penalties.

If we penalize the parameterization (i.e. non-geometric), total curvature becomes the thin plate energy functional. The variational derivative of thin plate energy is the linear biharmonic operator. Weighted averages of thin plate and membrane energies are used to construct mesh filters with desirable properties [28, 13]. Modifications to the mesh operators alleviate parameterization dependencies [8, 11]. In this paper, we focus on geometric surface smoothing strategies on level set surfaces using nonlinear penalty functions.

The work in this paper is also motivated by that of Chopp & Sethian [5], who derive the intrinsic Laplacian of curvature for an implicit curve, and solve the resulting fourth-order nonlinear PDE. However, they argue that the numerical methods used to solve second order flows are not practical, because they lack long term stability. They propose several new numerical schemes, but none are found to be completely satisfactory due to their slow computation

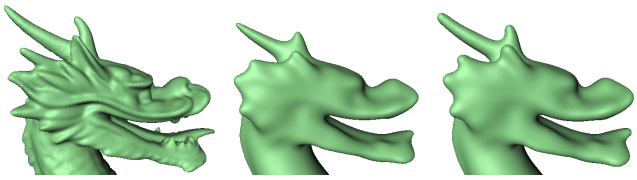


Figure 2: Second- and fourth-order surface smoothing. From left to right: Original model, mean curvature flow, and intrinsic Laplacian of mean curvature flow.

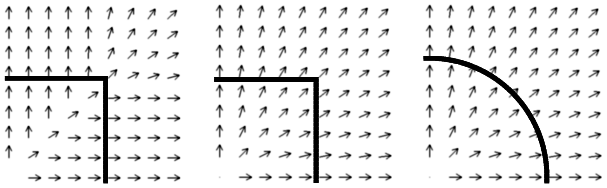


Figure 3: Shown here in 2D, the process begins with a shape and constructs a normal map from the distance transform (left), modifies the normal map according to a PDE derived from a penalty function (center), and refits the shape to the normal map (right).

and inability to handle singularities. The results in this paper allow us to solve this equation more effectively, in 3D, with an additional, nonlinear term that preserves sharp details.

3 MINIMIZING TOTAL CURVATURE

Several authors have observed the advantages of higher-order derivatives for smoothing surfaces [13, 8]. As an illustration of the importance of higher-order geometric processing, consider the results in Fig. 2, which demonstrates the differences between processing surfaces with mean curvature flow (MCF) and intrinsic Laplacian of mean curvature flow (ILMCF). The amount of smoothing for both was chosen to be qualitatively similar, and yet important differences can be observed on the smaller features of the model. MCF has shortened the horns, and yet they remain sharp—not a desirable behavior for a "smoothing" process. This behavior for MCF is well documented as a pinching off of cylindrical objects and is expected from the variational point of view: MCF minimizes surface area and therefore will quickly eliminate smaller parts of a model. Some authors [24] have proposed volume preserving forms of second-order flows, but these processes compensate by enlarging *the object as a whole*, which exhibits, qualitatively, the same behavior on small features. ILMCF, in Fig. 2, preserves the structure of these features much better *while* smoothing them.

In this section, we will introduce a method for solving general fourth order surface flows by breaking them into two second order PDEs. The specific flows we are interested in are isotropic (ILMCF) and anisotropic diffusion. This pair of equations is solved by allowing the surface shape to lag the normals as they are filtered and then refitted by a separate process. Figure 3 shows this three step process graphically in 2D for ILMCF—shapes give rise to normal maps, which, when filtered, give rise to new shapes.

3.1 Notation

To facilitate the discussion, we use the Einstein notation convention, where the subscripts indicate tensor indices, and repeated subscripts within a product represent a summation over the index (across the dimensions of the underlying space). Furthermore, we use the convention that subscripts on quantities represent derivatives, except where they are in parenthesis, in which case they refer to a vector-valued variable. Thus, ϕ_i is the gradient vector of a scalar quantity $\phi : \mathbb{R}^n \mapsto \mathbb{R}$. The Hessian is ϕ_{ij}, and the Laplacian is ϕ_{ii}. A vector field is $v_{(i)}$, where $v : \mathbb{R}^n \mapsto \mathbb{R}^n$, and the divergence of that field is $v_{(i)i}$. Scalar operators, such as differentials behave in the usual way. Thus, gradient magnitude is $\| \phi_i \| = \sqrt{\phi_i \phi_i}$ and the differential for a coordinate system is $dx_{(i)} = dx_1 dx_2 \ldots dx_n$.

Level set surface models rely on the notion of a regular surface, which is a collection of 3D points, $\mathscr{S}$, with a topology that allows each point to be modeled *locally* as a function of two variables. We can describe the deformation of such a surface using the 3D velocity of each of its constituent points, i.e., $ds_{(i)}(t)/dt$ for all $s_{(i)} \in \mathscr{S}$. If we represent the surface implicitly at each time t, then

$$\mathscr{S} = \left\{ s_{(i)}(t) \mid \phi\left(s_{(i)}(t), t\right) = 0 \right\}. \tag{2}$$

Surfaces defined in this way divide a volume into two parts: inside ($\phi > 0$) and outside ($\phi < 0$). It is common to choose ϕ to be the signed distance transform of $\mathscr{S}$, or an approximation thereof. The surface remains a level set of ϕ over time, and thus taking the total derivative with respect to time (using the chain rule) gives

$$\frac{\partial \phi}{\partial t} = -\phi_j \frac{ds_{(j)}}{dt}. \tag{3}$$

3.2 Total Curvature of Normal Maps

When using implicit representations one must account for the fact that derivatives of functions defined on the surface are computed by projecting their 3D derivatives onto the tangent plane of the surface. Let $N_{(i)} : \mathbb{R}^3 \to S^3$ be the normal map, which is a field of normals that are everywhere perpendicular to the family of embedded isosurfaces of ϕ—thus $N_{(i)} = \phi_i / \sqrt{\phi_k \phi_k}$. The 3×3 projection matrix for the implicit surface normal is $P_{(ij)} = N_{(i)} N_{(j)}$, and $P_{(ij)} V_{(i)}$ returns the projection of $V_{(i)}$ onto $N_{(i)}$. Let $I_{(ij)}$ be the identity matrix. Then the projection onto the plane that is perpendicular to the vector field $N_{(i)}$ is the *tangent projection operator*, $T_{(ij)} = I_{(ij)} - P_{(ij)}$. Under typical circumstances the normal map $N_{(i)}$ is derived from the gradient of ϕ, and $T_{(ij)}$ projects vectors onto the tangent planes of the level sets of ϕ. However, the computational strategy we are proposing allows ϕ to lag the normal map. Therefore, the tangent projection operators differ, and we use $T^{\phi}_{(ij)}$ to denote projections onto the tangent planes of the level sets of ϕ and $T^{N}_{(ij)}$ to denote projections onto planes perpendicular to the normal map.

The shape matrix [10] of a surface describes its curvature independent of the parameterization. The shape matrix of an implicit surface is obtained by differentiating the normal map and projecting that derivative onto the tangent plane of the surface. The Euclidean norm of the shape matrix is squared curvature

$$\kappa^2 = \left|\left| N_{(i)j} T^{\phi}_{(jk)} \right|\right|^2. \tag{4}$$

If $N_{(i)}$ is derived directly from ϕ, this gives

$$\kappa^2 = \left|\left| T^{\phi}_{(il)} \phi_{lj} T^{\phi}_{(jk)} \right|\right|^2. \tag{5}$$

The generic penalty function of total curvature is

$$\mathscr{G}\left(\kappa^2\right) = \int_{\mathscr{S}} g(\kappa^2) dx_{(i)}, \tag{6}$$

where $g(\kappa^2)$ is monotonically increasing scalar function. Notice, that if we take the first variation of (6) with respect to ϕ using (5) for total curvature we obtain a fourth-order PDE on ϕ. On the other hand, if we use (4) for total curvature and take the first variation

of (6) with respect to $N_{(i)}$, allowing ϕ to remain fixed, we obtain a second-order PDE on $N_{(i)}$. We take the latter approach in this paper.

As we process the normal map $N_{(i)}$, letting ϕ lag, we must ensure that it maintains the unit length constraint, $N_{(i)}N_{(i)} = 1$. This is expressed in the penalty function using Lagrange multipliers. The constrained penalty function is

$$\mathscr{G}\left(\kappa^2\right) \int_{\mathscr{S}} \lambda(x_{(l)}) \left(N_{(k)}N_{(k)} - 1\right) dx_{(l)}, \tag{7}$$

where $\lambda(x_{(l)})$ is the Lagrange multiplier at $x_{(l)}$. The next step is to derive the first variation. Curvature given by (4) can be written as

$$\kappa^2 = \left|\left|N_{(i)j}\right|\right|^2 - \frac{\left|\left|N_{(i)j}\phi_j\right|\right|^2}{\phi_k\phi_k}. \tag{8}$$

Using (8) and solving for λ in (7) introduces a projection operator $T^N_{(ij)}$ on the first variation of $\mathscr{G}$, which keeps $N_{(i)}$ unit length. Then the first variation of (7) with respect to $N_{(i)}$ is

$$T^N_{(ij)} \frac{d\mathscr{G}}{dN_{(j)}} = 2\, T^N_{(ij)} \left[g'\left(\kappa^2\right) \left(N_{(j)k} - \frac{N_{(j)l}\phi_l\phi_k}{\phi_m\phi_m} \right) \right]_k, \tag{9}$$

where g' is the derivative of g with respect to κ^2. A gradient descent on this metric $\partial N_{(i)}/\partial t = -T^N_{(ij)} d\mathscr{G}/dN_{(j)}$, results in a PDE that minimizes $g(\kappa^2)$ on the normal map. Notice that this is precisely the same as solving the constrained diffusion equation on $N_{(i)}$ using the method of solving PDEs on implicit manifolds described by [3]. We will discuss several choices for g in Sec. 4.

3.3 Surface Evolution via Normal Maps

We have shown how to evolve the normals to minimize functions of curvature; however, the final goal is to process the surface, which requires deforming ϕ. Hence, the next step is to relate the evolution of ϕ to the evolution of $N_{(i)}$. Suppose that we are given the normal map $N_{(i)}$ to some set of surfaces, but not necessarily level sets of ϕ—as is the case if we filter $N_{(i)}$ and let ϕ lag. We can manipulate ϕ so that it fits the normal field $N_{(i)}$ by minimizing a penalty function that quantifies the discrepancy. This penalty function is

$$\mathscr{D}(\phi) = \int_U \left[\sqrt{\phi_i\phi_i} - \phi_i N_{(i)} \right] dx_{(j)}, \tag{10}$$

where $U \subset \mathbb{R}^3$ is the domain of ϕ. The first variation with respect to ϕ is

$$\frac{d\mathscr{D}}{d\phi} = -||\phi_k|| \left[\left(\frac{\phi_j}{\phi_m\phi_m} \right)_j - N_{(j)j} \right] = -||\nabla\phi|| \left[\mathrm{H}^\phi - \mathrm{H}^\mathrm{N} \right] \tag{11}$$

where H^ϕ is the mean curvature of the level set surface and H^N is the induced curvature of the normal map. A gradient descent on ϕ that minimizes this penalty function is $\partial\phi/\partial t = -d\mathscr{D}/d\phi$. Thus, the surface moves as the difference between its own curvature and that of the normal field. The factor of $\| \nabla\phi \|$, which is typical with level set formulations, comes from the fact that we are manipulating the shape of the level set, which is embedded in ϕ, as in (3).

We propose to solve fourth-order flows on the level sets of ϕ by a splitting strategy, which entails processing the normals and allowing ϕ to lag and then be refitted later, in a separate process. In a related work, joint interpolation of vector fields and gray level functions was used for successfully filling-in missing parts of images in

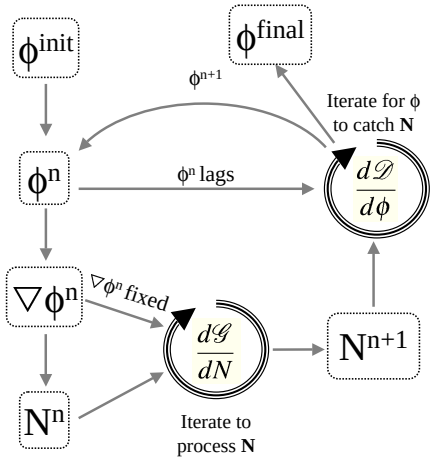


Figure 4: Flow chart

[2]. We have derived a gradient descent for the normal map based on a certain class of penalty functions that use total curvature in Sec. 3.2. This process is denoted in Fig. 4 as the $d\mathscr{G}/d\mathbf{N}$ loop. The surface refitting to the normal map is formulated in Sec. 3.3. This process is the $d\mathscr{D}/d\phi$ loop in Fig. 4. The overall algorithm shown in Fig. 4 repeats these two steps to minimize the penalty functions in terms of the surface. We refer to both of these processes, back to back, as one iteration of our algorithm. In [27] we have shown that the overall process of simultaneously solving these two PDEs as shown is equivalent to the fourth-order flow on the original surface. This establishes the mathematical foundation of the proposed method.

4 APPLICATIONS

The flexible normal map energy minimization and surface refitting methodology introduced in Sec. 3 allows us to experiment with various forms of g in (6) that give rise to different classes of penalty functions. The choice of $g(\kappa^2) = \kappa^2$ leads to ILMCF which we will refer to as isotropic diffusion. Minimizing the total curvature of a surface works well for smoothing surfaces and eliminating noise, but it also deforms or removes important features. This type of smoothing is called isotropic because it corresponds to solving the heat equation on the normal map with a constant, scalar conduction coefficient. Isotropic diffusion is not particularly effective if the goal is to denoise a surface that has an underlying structure with fine features.

Figure 1(a) showed an example of the skin surface, which was extracted, via isosurfacing, from an magnetic resonance imaging (MRI) data set. The roughness of the skin is noise, an artifact of the measurement process. This model is also topologically complex because, despite our best efforts to avoid it, the isosurfaces include many convoluted passages up in the sinuses and around the neck. As an indication of this complexity, consider that marching cubes produces 543,000 triangles from this $256 \times 256 \times 175$ volume. Isotropic diffusion, shown in Fig. 1(b), is marginally effective for denoising the head surface. Notice that the sharp edges around the eyes, nose, lips and ears are lost in this process.

The problem of preserving features while smoothing noise has been studied extensively in computer vision. Perona & Malik [21] proposed to replace Laplacian smoothing, which is equivalent to the heat equation $\partial I/\partial t = \nabla \cdot \nabla I$, with a nonlinear, anisotropic PDE

$$\partial I/\partial t = \nabla \cdot \left[g'\left(\| \nabla I \|^2 \right) \nabla I \right], \tag{12}$$

where I is generally the grey-level image. This PDE is the first

variation of

$$\int_U g\left(\| \nabla I \|^2\right)\, dx\, dy \tag{13}$$

where $g\left(\| \nabla I \|^2\right)$ is the edge stopping function, g' is its derivative with respect to $\| \nabla I \|^2$, and U is the image domain. Perona & Malik suggested using $g(x) = e^{-|\nabla I|^2/2\mu}$, where μ is a positive, free parameter that controls the level of contrast of edges that can affect the smoothing process. Notice that $g(\| \nabla I \|)$ approaches 1 for $\| \nabla I \| \ll \mu$ and 0 for $\| \nabla I \| \gg \mu$. Edges are generally associated with large image gradients, and thus diffusion across edges is stopped while regions that are relatively flat undergo smoothing. A mathematical analysis shows that solutions to (12) can actually exhibit an inverse diffusion near edges, and can enhance or sharpen smooth edges that have gradients greater than μ [24].

The generalization of P&M anisotropic diffusion to surfaces is achieved from variational principles by choosing the appropriate function of the squared curvature in (6). For instance,

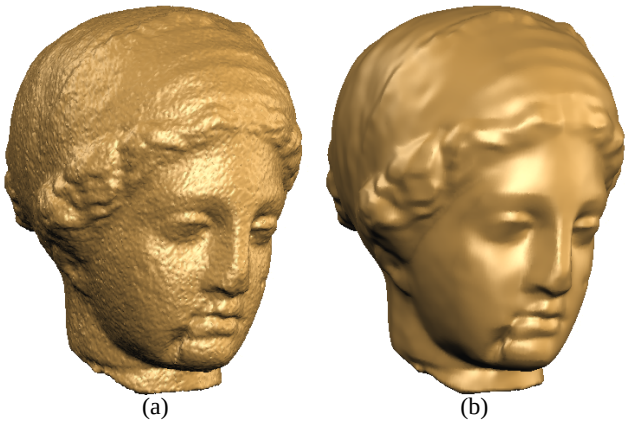
$$g(\kappa^2) = 2\mu^2\left(1 - e^{-\frac{\kappa^2}{2\mu^2}}\right), \text{ and } g'(\kappa^2) = e^{-\frac{\kappa^2}{2\mu^2}}, \tag{14}$$

where g' is the derivative of g with respect to κ^2. The first variation with respect to the surface normals gives a vector-valued anisotropic diffusion on the level set surface—a straightforward generalization of (12). This flow is a modified version of ILMCF that preserves or enhances areas of high curvature, which we will call *creases*. Creases are the generalization of edges in images to surfaces. The differences between anisotropic diffusion and isotropic diffusion can clearly be observed in Fig. 1(c). Around the smooth areas of the original model such as the forehead and the cheeks, there is no noticeable difference in the results of the two processes. However, very significant differences exist around the lips and the eyes. The creases in these areas, which have been eliminated by isotropic diffusion, are preserved by the anisotropic process. The computation time required for one iteration of the main processing loop operating on this model is approximately 20 minutes on a *1.7 Ghz Intel processor* for both isotropic and anisotropic diffusion. The results shown in Fig. 1(b) and (c) are both after 3 iterations which translates to around 60 minutes of processing.

The generality of the proposed approach comes at the cost of significant computation time. However, the method is practical with state-of-the-art computers and is well-poised to benefit from parallel computing architectures, due to its reliance on local, iterative computations. Furthermore, the computation times for the proposed method are quite competitive if one compares them to the end-to-end computation times for meshes, which can include manually establishing base meshes and/or "fixing" topological problems, which can require several hours of computation [12].

Another example of denoising by anisotropic diffusion is shown in Fig. 5. Independent Gaussian noise with standard deviation 1.0 was added to the original model which in this case is a $221 \times 221 \times 161$ volume. The noise was added to the voxel values in the volume which are distances to the surface. After 3 iterations of the main processing loop (see Fig. 4) the noise was successfully removed while preserving the features of the original model. A qualitative, visual comparison of these results with results from the same model shown in [6](Fig. 2) suggests that the results in this paper demonstrate better preservation of fine, sharp details, such as those around the eyes and in the hair. The computation times per iteration for this example are approximately 7 minutes on a *1.7 Ghz Intel processor* compared to 20 minutes per iteration for the example in Fig. 1. This is indicative of the relatively high degree of complexity of the MRI based model.

Figure 6(a) shows a different isosurface (the cortex) extracted from the same MRI scan as the model in Fig. 1. The complexity of this model, i.e. the many tightly nested folds, make it ill suited



(a) (b)

Figure 5: (a) Noisy venus head model, and (b) smoothed version after 3 iterations of anisotropic diffusion.

for mesh based deformations. Also the main cortical surface has many detached pieces, an artifact of the segmentation process. The approach proposed in this paper can automatically simplify topologically noisy features due to the level set implementation — an important aspect of denoising measured surfaces.

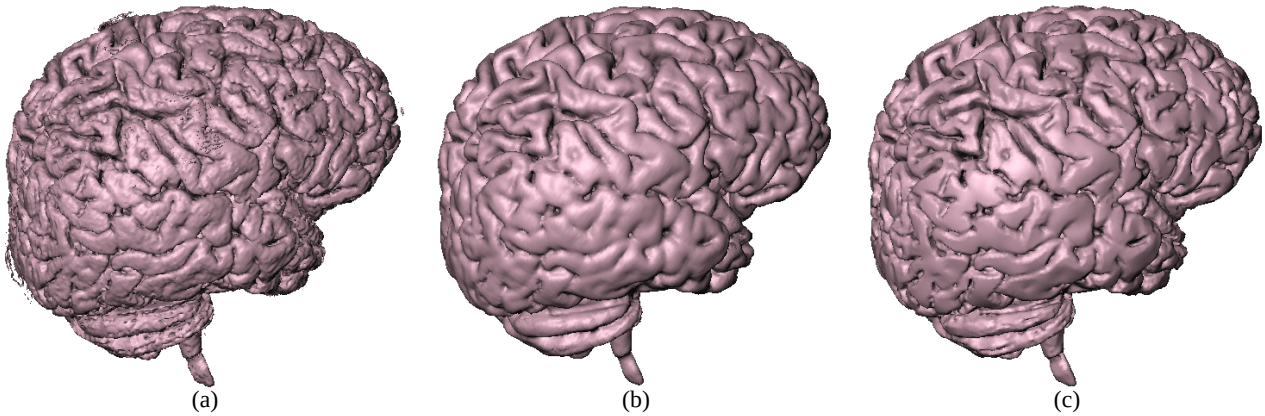
The example in Fig. 7 demonstrates another aspect of the proposed method. Although this model was constructed as a volume directly from 3D range data [7], it does not exhibit significant noise. When running the proposed method for anisotropic diffusion, however, surfaces *tend* toward solutions that have piecewise constant normals with sharp discontinuities in the normal map—analogous to the behavior of the P&M equation for intensity images. Such properties in the normal map correspond to surfaces consisting of planar patches bounded by sharp creases. Thus, the proposed method generates a feature preserving scale space, very much like that of P&M for images. These results support our proposition that processing the normals of a surface is the natural generalization of image processing. The non-linear progression of elimination of details from the smallest scale to the largest in Fig. 7 also suggests applications of this method to surface compression and multi-resolution modeling.

5 CONCLUSION

A generalization of anisotropic diffusion for surfaces leads to a smoothing process that enhances creases while reducing small-scale noise. Our approach is based on the proposition that the *natural* generalization of image processing to surfaces is via the normals. Variational expressions for the surface have corresponding variational formulations on the surface normals. This philosophy leads to a clear generalization of P&M anisotropic diffusion to surfaces. As a result the behavior for surfaces that mirrors that of images—we see piecewise smooth surface patches bounded by high curvature creases. Processing the normals separately from the surface leads to a pair of coupled second-order equations instead of a fourth-order equation.

We solve the surface deformation using level sets, an implicit representation that relies on a discrete grid, which is a volume. Because of this implementation, the proposed method applies to any shape that can be modeled as an isosurface. Consequently, our results show results with a level of surface complexity that goes beyond that of previous methods. For example, the proposed method can be used on isosurfaces that are extracted directly from 3D medical data—a difficult task for mesh based approaches.

Future work will study the usefulness of other interesting image processing techniques such as *total variation* [23]. To date, we have dealt with post processing noisy surfaces. The noise en-



(a) (b) (c)

Figure 6: (a) Original brain isosurface from MRI data set, (b) result of MCF, and (c) after 5 iterations of anisotropic diffusion.

(a) (b) (c)

Figure 7: Various stages of anisotropic diffusion: (a) original model, (b) after 10 iterations, and (c) after 20 iterations.

countered in these examples, such as the MRI scans, were additive noise in the volume elements. Other surface processing problems exhibits different types of noise, such as the line-of-sight noise in reconstruction from laser range finder data [32]. Future work will combine the proposed method with segmentation and reconstruction from range data techniques. The current shortcoming of this method is the computation time, which is significant. However, the process lends itself to parallelism, and the advent of cheap, specialized, vector-processing hardware promises significantly faster implementations. Multi-grid adaptive level sets and implicit PDE solvers are possibilities for research towards reductions in the computational complexity of the method.

REFERENCES

[1] D. Adalsteinson and J. A. Sethian. A fast level set method for propogating interfaces. *J. Computational Physics*, pages 269–277, 1995.

[2] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera. Filling-in by joint interpolation of vector fields and gray levels. *IEEE Trans. on Image Processing*, 10(8):1200–1211, August 2001.

[3] M. Bertalmio, L.-T. Cheng, S. Osher, and G. Sapiro. Variational methods and partial differential equations on implicit surfaces. *J. Computational Physics*, 174:759–780, 2001.

[4] David Breen, Sean Mauch, and Ross Whitaker. 3d scan conversion of csg models into distance volumes. In *Proceedings of the 1998 Symposium on Volume Visualization, ACM SIGGRAPH*, pages 7–14, October 1998.

[5] D. L. Chopp and J. A. Sethian. Motion by intrinsic laplacian of curvature. *Interfaces and Free Boundaries*, 1:1–18, 1999.

[6] U. Clarenz, U. Diewald, and M. Rumpf. Anisotropic geometric diffusion in surface processing. In *Proceedings of IEEE Visualization*, pages 397–405, 2000.

[7] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. *Computer Graphics (SIGGRAPH '96 Proceedings)*, July 1996.

[8] M. Desbrun, M. Meyer, M. Schroder, and A. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of SIGGRAPH'99*, pages 317–324, 1999.

[9] M. Desbrun, M. Meyer, P. Schroder, and A. H. Barr. Anisotropic feature-preserving denoising of height fields and bivariate data. In *Graphics Interface*, 2000.

[10] M. P. DoCarmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.

[11] I. Guskov, W. Sweldens, and P. Schroder. Multiresolution signal processing for meshes. In *Proceedings of SIGGRAPH'99*, pages 325–334, 1999.

[12] I. Guskov and Z. J. Wood. Topological noise removal. In *Graphics Interface*, 2001.

[13] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of SIGGRAPH'98*, pages 105–114, 1998.

[14] W. Lorenson and H. Cline. Marching cubes: A high resolution 3d surface reconstruction algorithm. In *Proceedings of SIGGRAPH'87*, pages 163–169, 1987.

[15] Liana Lorigo, Olivier Faugeras, Eric Grimson, Renaud Keriven, Ron Kikinis, Arya Nabavi, and Carl-Fredrik Westin. Co-dimension 2 geodesic active contours for the segmentation of tubular strucures. In *Proceedings of Computer Vision and Pattern Recognition*, 2000.

[16] Ravikanth Malladi, James A. Sethian, and Baba C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Trans. on PAMI*, 17(2):158–175, 1995.

[17] H. P. Moreton and C. H. Sequin. Functional optimization for fair surface design. In *Proceedings of SIGGRAPH'92*, pages 167–176, 1992.

[18] Y. Ohtake, A. G. Belyaev, and I. A. Bogaevski. Polyhedral surface smoothing with simultaneous mesh regularization. In *Geometric Modeling and Processing*, 2000.

[19] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulation. *J. Computational Physics*, 79:12–49, 1988.

[20] D. Peng, B. Merriman, S. Osher, H-K Zhao, and M. Kang. A pde based fast local level set method. *J. Computational Physics*, 155:410–438, 1999.

[21] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.

[22] A. Polden. Compact surfaces of least total curvature. Technical report, University of Tubingen, Germany, 1997.

[23] L. Rudin, S. Osher, and C. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[24] Guillermo Sapiro. *Geometric Partial Differential Equations and Image Analysis*. Cambridge University Press, 2001.

[25] R. Schneider and L. Kobbelt. Generating fair meshes with g^1 boundary conditions. In *Geometric Modeling and Processing Proceedings*, pages 251–261, 2000.

[26] J. A. Sethian. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Material Sciences*. Cambridge University Press, 1996.

[27] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher. Geometric surface processing via normal maps. Technical Report UUCS-02-03, University of Utah, January 2002.

[28] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of SIGGRAPH'95*, pages 351–358, 1995.

[29] G. Taubin. Linear anisotropic mesh filtering. Technical Report RC22213, IBM Research Division, October 2001.

[30] W. Welch and A. Witkin. Free-form shape design using triangulated surfaces. In *Proceedings of SIGGRAPH'94*, pages 247–256, 1994.

[31] Ross T. Whitaker. Volumetric deformable models: Active blobs. In Richard A. Robb, editor, *Visualization In Biomedical Computing*. SPIE, 1994.

[32] Ross T. Whitaker. A level-set approach to 3D reconstruction from range data. *Int. J. Computer Vision*, 29(3):203–231, 1998.

A NUMERICAL IMPLEMENTATION

By embedding surface models in volumes, we have converted equations that describe the movement of surface points to nonlinear PDEs defined on a volume. The next step is to discretize these PDEs in space and time. In this paper, the embedding function ϕ is defined on the volume domain U and time. The PDEs are solved using a discrete sampling with forward differences along the time axis. There are two issues with discretization: (i) the accuracy and stability of the numerical solutions, (ii) the increase in computational complexity introduced by the dimensionality of the domain.

For brevity, we will discuss the numerical implementation in 2D— the extension to 3D is straightforward. The function $\phi : U \mapsto \mathbb{R}$ has a discrete sampling $\phi[p,q]$, where $[p,q]$ is a grid location and $\phi[p,q] = \phi(x_p, y_q)$. We will refer to a specific time instance of this function with superscripts, i.e. $\phi^n[p,q] = \phi(x_p, y_q, t_n)$. For a vector in 2-space v, we use $v_{(x)}$ and $v_{(y)}$ to refer to its components consistent with the notation of Sec. 3. In our calculations, we need three different approximations to first-order derivatives: forward, backward and central differences. We denote the type of discrete difference using superscripts on a difference operator, i.e., $\delta^{(+)}$ for forward differences, $\delta^{(-)}$ for backward differences, and δ for central differences. For instance, the differences in the x direction on a discrete grid with unit spacing are

$$\begin{aligned} \delta_x^{(+)}\phi[p,q] &\triangleq \phi[p+1,q] - \phi[p,q], \\ \delta_x^{(-)}\phi[p,q] &\triangleq \phi[p,q] - \phi[p-1,q], \quad \text{and} \\ \delta_x\phi[p,q] &\triangleq (\phi[p+1,q] - \phi[p-1,q])/2, \end{aligned} \tag{15}$$

where the time superscript has been left off for conciseness. The application of these difference operators to vector-valued functions denotes componentwise differentiation.

The positions of derivatives computed with forward and backwards differences are staggered off the grid by $1/2$ pixels. For instance, $\delta_x^{(+)}\phi[p,q]$ as defined in (15) uses information from positions $[p+1,q]$ and $[p,q]$ with equal weights; hence, it exists at $[p+1/2,q]$. This is staggered by $1/2$ pixels in the x direction from the grid. To keep track of staggered locations, we will use the following notation: α, $\overset{x+}{\alpha}$, and $\overset{y+}{\alpha}$ will denote the variable α computed at $[p,q]$, $[p+1/2,q]$, and $[p,q+1/2]$, respectively.

In describing the numerical implementation, we will refer to the flow chart in Fig. 4 for one time step of the main loop. Hence, the first step in our numerical implementation is the calculation of the surface normal vectors from ϕ^n. Recall that the surface is a level set of ϕ^n as defined in (2). Hence, the surface normal vectors can be computed as the unit vector in the direction of the gradient of ϕ^n. The gradient of ϕ^n is computed with central differences as

$$\phi_i^n[p,q] \approx \begin{pmatrix} \delta_x\phi^n[p,q] \\ \delta_y\phi^n[p,q] \end{pmatrix}; \tag{16}$$

and the normal vectors are initialized as

$$N_{(i)}^{u=0}[p,q] = \phi_i^n[p,q] / \parallel \phi_k^n[p,q] \parallel . \tag{17}$$

Because ϕ^n is fixed and allowed to lag behind the evolution of $N_{(i)}$, the time steps in the evolution of $N_{(i)}$ are denoted with a different superscript, u. For this evolution, $\partial N_{(i)}/\partial t = -d\mathscr{G}/dN_{(i)}$

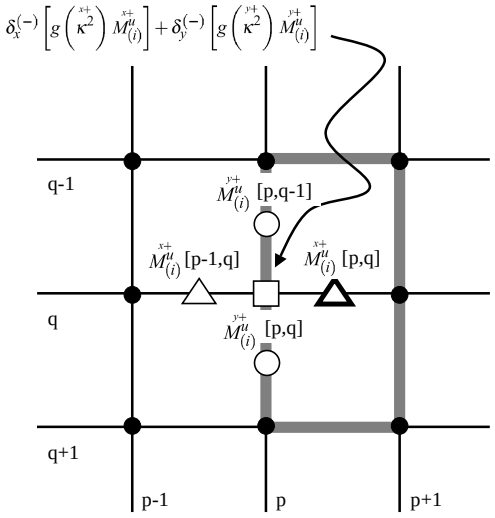


Figure 8: Computation grid

is implemented with smallest support area operators. For ILMCF and anisotropic diffusion $d\mathscr{G}/dN_{(i)}$ is given by (9) which we will rewrite here component by component. The Laplacian of a function can be applied in two steps, first the gradient and then the divergence. In 2D, the gradient of the normals produces a 2×2 matrix, and the divergence operator in (9) collapses this to a 2×1 vector. The diffusion of the normal vectors in the tangent plane of the level sets of ϕ, requires us to compute the flux in the x and y directions. The "columns" of the flux matrix are computed independently as

$$M_{(i)}^{x+u} \approx \delta_x^{(+)}N_{(i)}^u - C_{(i)}^{x+u}\left(\delta_x^{(+)}\phi^n\right), \tag{18}$$

$$M_{(i)}^{y+u} \approx \delta_y^{(+)}N_{(i)}^u - C_{(i)}^{y+u}\left(\delta_y^{(+)}\phi^n\right) \tag{19}$$

where the time index n remains fixed as we increment u, and where

$$C_{(i)}^{x+u}=N_{(i)j}^{x+u}\phi_j^{x+n}/(\phi_k^{x+n}\phi_k^{x+n}), \quad \text{and} \quad C_{(i)}^{y+u}=N_{(i)j}^{y+u}\phi_j^{y+n}/(\phi_k^{y+n}\phi_k^{y+n}). \tag{20}$$

Derivatives are computed with forward differences; therefore they are staggered, located on a grid that is offset from the grid where ϕ and $N_{(i)}$ are defined, as shown Fig. 8 for the 2D case. Furthermore, notice that since the offset is half pixel only in the direction of the differentiation, the locations of $\delta_x^{(+)}N_{(i)}$ and $\delta_y^{(+)}N_{(i)}$ are different, but are the same locations as the flux (18) and (19) respectively. To evaluate (20), derivatives of ϕ and $N_{(i)}$ must computed at $[p+1/2,q]$ and $[p,q+1/2]$, respectively. These computations are done with the smallest support area operators, using the symmetric 2×3 grid of samples around each staggered point, as shown in Fig. 8 with the heavy rectangle. For instance, the staggered gradients of ϕ are

$$\phi_j^{x+n}=\phi_j^n[p+\frac{1}{2},q] \approx \begin{pmatrix} \delta_x^{(+)}\phi[p,q] \\ \frac{1}{2}\left(\delta_y\phi[p,q]+\delta_y\phi[p+1,q]\right)\end{pmatrix},$$

$$\phi_j^{y+n}=\phi_j^n[p,q+\frac{1}{2}] \approx \begin{pmatrix} \frac{1}{2}\left(\delta_x\phi[p,q]+\delta_x\phi[p,q+1]\right) \\ \delta_y^{(+)}\phi[p,q]\end{pmatrix} \tag{21}$$

The staggered gradients of the normals are computed in the same way as (21). To evaluate (9), we also need to compute $g'(\kappa^2)$ at the precise locations where the flux (18) and (19) are located. For this, we need the total intrinsic curvature

$$\kappa^{2\,x+}=N_{(i)j}^{x+u}N_{(i)j}^{x+u}-C_{(i)}^{x+u}C_{(i)}^{x+u}, \quad \text{and} \quad \kappa^{2\,y+}=N_{(i)j}^{y+u}N_{(i)j}^{y+u}-C_{(i)}^{y+u}C_{(i)}^{y+u} \tag{22}$$

at the required locations.

Backwards differences of the flux are used to compute the divergence operation in (9)

$$\left[-\frac{d\mathscr{G}}{dN_{(i)}}\right]^u \approx \delta_x^{(-)}\left[g\left(\kappa^{2\,x+}\right)M_{(i)}^{x+u}\right]+\delta_y^{(-)}\left[g\left(\kappa^{2\,y+}\right)M_{(i)}^{y+u}\right] \tag{23}$$

Notice that these backwards differences are defined at the original ϕ grid location $[p,q]$ because they undo the forward staggering in the flux locations. Thus both components of $d\mathscr{G}/dN_{(i)}$ are located on the original grid for ϕ. Using the tangential projection operator in (9), the new set of normal vectors are computed as

$$N_{(i)}^{u+1}=N_{(i)}^u+T^N\left[\frac{d\mathscr{G}}{dN_{(i)}}\right]^u=N_{(i)}^u+\left[\frac{d\mathscr{G}}{dN_{(i)}}\right]^u\left(I_{(ij)}-N_{(j)}^uN_{(i)}^u\right). \tag{24}$$

Starting with the initialization in (17) for $u=0$, we iterate (24) for a fixed number of steps, 25 iterations for the examples in this paper. In other words, we do not aim at minimizing the energy given in (7) in the $d\mathscr{G}/dN$ loop of Fig. 4; we only reduce it. The minimization of total mean curvature as a function of ϕ is achieved by iterating the main loop in (17).

Once the evolution of N is concluded, ϕ is evolved to catch up with the new normal vectors according to (11). We denote the evolved normals by $N_{(i)}^{n+1}$. To compute (11) we must calculate H^ϕ and $H^{N^{n+1}}$. $H^{N^{n+1}}$ is the induced mean curvature of the normal map; in other words, it is the curvature of the hypothetical target surface that fits the normal map. Calculation of curvature from a field of normals is

$$H^{N^{n+1}} \approx \delta_x N_x^{n+1}+\delta_y N_y^{n+1}, \tag{25}$$

where we have used central differences on the components of the normal vectors. $H^{N^{n+1}}$ needs to be computed once at initialization as the normal vectors remain fixed during the catch up phase. Let v be the time step index in the $d\mathscr{D}/d\phi$ loop. H^{ϕ^v} is the mean curvature of the moving level set surface at time step v and is calculated from ϕ with the smallest area of support

$$H^{\phi^v} \approx \delta_x^{(-)}\delta_x^{(+)}\phi^v/(\phi_j^{x+v}\phi_j^{x+v})+\delta_y^{(-)}\delta_y^{(+)}\phi^v/(\phi_j^{y+v}\phi_j^{y+v}) \tag{26}$$

where the gradients in the denominators are staggered to match the locations of the forward differences in the numerator. The staggered gradients of ϕ in the denominator are calculated using the 2×3 neighborhood as in (21).

The PDE $\partial\phi/\partial t=-d\mathscr{D}/d\phi$ is solved with a finite forward differences, but with the up-wind scheme for the gradient magnitude [19], to avoid overshooting and maintain stability. The up-wind method computes a one-sided derivative that looks in the up-wind direction of the moving wave front, and thereby avoids overshooting. Moreover, because we are interested in only a single level set of ϕ, solving the PDE over all of U is not necessary. Different level sets evolve independently, and we can compute the evolution of ϕ only in a narrow band around the level set of interest and re-initialize this band as necessary [1, 20]. See [26] for more details on numerical schemes and efficient solutions for level set methods. Using the upwind scheme and narrow band methods, ϕ^{v+1} is computed from ϕ^v using the curvatures computed in (25) and (26). This loop is iterated until the energy in (10) ceases to decrease; let v^{final} denote the final iteration of this loop. Then we set ϕ for the next iteration of the main loop (see Fig. 4) as $\phi^{n+1}=\phi^{v^{final}}$ and repeat the entire procedure. The number of iterations of the main loop is a free parameter that generally determines the extent of processing.