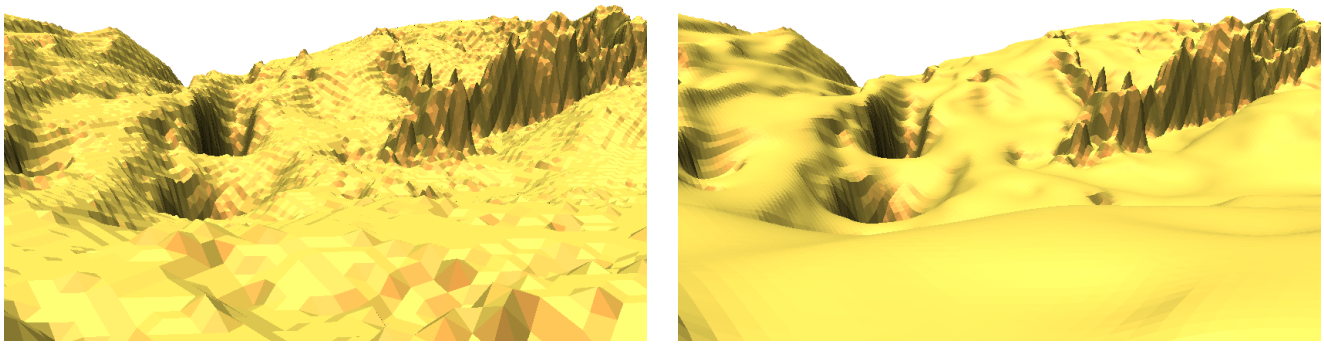
Anisotropic Feature-Preserving Denoising of Height Fields and Bivariate Data

Mathieu Desbrun[†‡] Mark Meyer[†] Peter Schröder[†] Alan H. Barr[†]

[†] Caltech - [‡] USC



(a) (b)

Figure 1: *Mars elevation map: (a) raw data, (b) smooth version after anisotropic diffusion. Notice how, with our non-uniform diffusion, the aliasing due to poor quantization is suppressed without altering the general topography of the surface (both pictures are flat-shaded).*

Abstract

In this paper, we present an efficient way to denoise bivariate data like height fields, color pictures or vector fields, while preserving edges and other features. Mixing surface area minimization, graph flow, and nonlinear edge-preservation metrics, our method generalizes previous anisotropic diffusion approaches in image processing, and is applicable to data of arbitrary dimension. Another notable difference is the use of a more robust discrete differential operator, which captures the fundamental surface properties. We demonstrate the method on range images and height fields, as well as greyscale or color images.

CR Categories: I.3.7 [Computer Graphics] Three-Dimensional Graphics and Realism; I.4.3 [Image Processing and Computer Vision] Enhancement.

Keywords: Anisotropic diffusion, Re-parameterization, Surface flows, Edge preservation, Image processing, Surface fairing.

1 Introduction

The problem of smoothing surfaces in computer graphics is closely related to the smoothing of images in computer vision. In both cases, we wish to smooth a 2-dimensional dataset, while preserving important features such as discontinuities. The discontinuities represent cliffs in height fields, and edges in images (edges in images arise when the associated height-field values—intensities of the image—change sharply with position).

To reduce the noise in images, early research has advocated the use of the laplacian as a local differential operator. Diffusing the signal using laplacian smoothing will reduce high frequency noise. Unfortunately, an unintended consequence is that the noise is diffused uniformly in screen space. Sharp edges and other fundamental features of an image are then lost, blurred away by the uniform diffusion. Consequently, anisotropic operators have been proposed. They can diffuse the signal non-uniformly to better preserve edges, while reducing noise in the signal.

The concept of noise removal with preservation of edges can be used for denoising textures acquired from images, or generalized for meshes to preserve features while removing small bumps, or even for reducing quantization effects in height fields. In this paper, we propose a technique for all these applications. Using robust differential geometry tools developed in the discrete domain for computer graphics, we define a non-linear anisotropic diffusion operator valid for any bivariate data (height fields, greyscale images, color images, tensor images). We explain the relations between our approach and previous work in image processing, and show results to demonstrate its reliability.

1.1 Anisotropic Diffusion in Image Processing

The first inhomogeneous diffusion model was introduced by Perona and Malik [PM90]. The idea was to vary the conduction spatially to favor noise removal in nearly homogeneous regions while avoiding any alteration of the signal along significant discontinuities (see [TT99] for an intuitive explanation of this technique). The change in intensity I over time was defined as:

$$I_t = div(\ g(||\nabla I||)\ \nabla I) \quad \text{with: } g(x) = \frac{1}{1+\frac{x^2}{\alpha}}. \qquad (1)$$

Many different variations on the conduction function g have been proposed [ROF92, ABBFC97, ALM92], and recently a higher order PDE has been introduced by Tumblin [TT99] in the context of displaying high contrast computer graphics pictures. Similar techniques have been used to visualize complex flow fields, as in [PR99]. All of these approaches rely on isophotes of the image (see Figure 2(a)): the anisotropic diffusion equation can be interpreted as a diffusion mainly in the direction tangential to each isophote. Therefore, discontinuities present in the orthogonal direction are not lost, as explained in [KDA97]. Typically, finite difference schemes are used to discretize the differential operators used.

Some of these approaches also use an inverse diffusion process orthogonal to the isophotes to enhance edges; this process, being very unstable by nature, requires a pre-smoothing of the gradient for the well-posedness of the problem.

However, in general, relying only on isophotes to restore a noisy image is questionable: non-uniform lighting (glares, specularity effects) often enhance our understanding of a scene while significantly affecting isophotes in complex ways. Other anisotropic diffusion models are therefore desirable.

1.2 Overview of Our Approach

In the remainder of this paper, we will develop a geometric framework for the denoising of images using a *surface-based* approach inspired by [DMSB99]. We will show how:

- a careful examination of the smoothing problem for surfaces in computer graphics,
- a straightforward adaptation to graph flows through reparameterization, and
- a penalization along edges

will allow us to define a *simple and very general denoising technique based on surface area minimization*. This will lead to 3D curvature-based models, mimicking diffusion with respect to the metric induced by the image itself instead of the usual screen space metric. Moreover, the numerical method to implement this technique will use this very nature of surface area minimization over the discrete data, ensuring a natural, accurate operator to integrate the resulting flow.

The generality of our framework enables us to use this technique on data of any dimensionality, such as greyscale or color images, with an overhead only linear in the dimension. We will also demonstrate how range images can be treated naturally with this method, offering better smoothing of a scanned surface than previous methods. More complex data, like flow fields or tensor images, can be treated in the exact same way by extending the concept of curvature for bivariate data in higher dimensions.

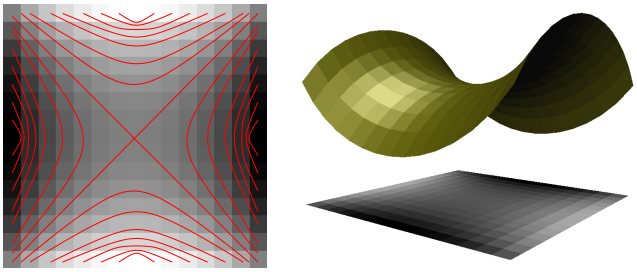
This paper is organized as follows: Section 2 establishes the main notation and discusses related work about surface flows. In Section 3, we derive a parameterization-independent flow for greyscale images before generalizing the approach to higher dimensional data such as color images (Section 4). We present a robust numerical scheme to integrate our flow in Section 5, then show results in Section 6 and conclude in Section 7.

2 Background on Surface Flows

A number of approaches for denoising in image processing research consider an image as a 2-manifold embedded in 3D: the image $I(x, y)$ is regarded as a surface $(x, y, I(x, y))$ in a three dimensional space, as depicted in Figure 2(b). Embedding the image in a higher dimension allows us to use richer and more meaningful differential operators. In this section, we review the different methods based on this idea, and give general definitions that we will build upon later.

2.1 Intensity as a 2-manifold

An image $I(x, y)$, usually considered as a function on a 2D plane (x, y), can also be regarded as a surface $(x, y, I(x, y))$ in a three dimensional space as depicted in Figure 2(b). The surface $S = (x, y, I(x, y))$ is sometimes called a Monge surface, or simply a *height field* as the intensity represents an elevation along the z direction of the (x, y, z) space. As we will see in the remainder of this



(a) (b)

Figure 2: *The intensity map $I(x, y)$ of an image can be thought of as (a) a set of isophotes, or (b) a height field $(x, y, z = I(x, y))$.*

paper, considering an image as a surface, will allow us to use some well known differential geometry properties to design appropriate differential operators.

We will denote by $\mathbf{n}$ the normal of the surface S, and will use $\mathcal{W}$, the square root of the first fundamental form determinant of the surface [DHKW92, Gra98]. This latter quantity measures at a given point on the surface the area expansion between the parameter domain and the surface itself: a surface dA on the screen (parameter domain, also called screen space in our context) will then represent a surface area of $\mathcal{W}\ dA$ on the height field. Due to the simplicity of a height field, we can write:

$$\mathcal{W} = \sqrt{1 + I_x^2 + I_y^2} \tag{2}$$

$$\mathbf{n} = \frac{1}{\mathcal{W}}(-I_x, -I_y, 1), \tag{3}$$

where I_x (resp. I_y) is the first derivative of I with respect to x (resp. y).

2.2 Laplace-Beltrami Operator

As mentioned earlier, many recent approaches have focused on diffusing 2D isophote curves. For an image regarded as an embedding in 3D, a natural extension of this idea is to consider *surface diffusion*. The 2D curvature operator is then replaced by the mean curvature in 3D, denoted κ hereafter. A differential operator measures this mean curvature: the *Laplace-Beltrami operator*. Traditionally denoted Δ_g [DHKW92], this operator gives the mean curvature normal of the surface S:

$$\Delta_g S = 2\kappa\mathbf{n}.$$

Commonly used in differential geometry [DHKW92, Gra98], it is often referred as the natural generalization of the laplacian from flat spaces to general manifolds, as it uses the induced metric of the surface itself, not the metric of the parameter domain.

Almost all surface flow techniques consider this mean curvature normal for edge-preserving denoising, albeit in significantly varying ways. We briefly review these different flows used in image processing and vision, along with their motivations:

- Malladi and Sethian [MS96] proposed: $I_t = \mathcal{W}\kappa$ to implement the geometrically natural mean curvature flow. Contrary to the conventional laplacian filtering, it is an anisotropic flow more appropriate for a scale-space. They also derive a min/max flow, thresholding the curvature locally depending on local averages.
- Extending the Perona-Malik formulation for an intensity height field, Ford and El-Fallah [FEF98] proposed an inhomogeneous diffusion with a coefficient inversely proportional to the gradient magnitude:

$$I_t = div(\frac{1}{\sqrt{1 + I_x^2 + I_y^2}}(-I_x, -I_y, 1)^t).$$

Since this expression is actually the divergence of the unit normal $\mathbf{n}$ to the surface, we can reformulate it as:

$$I_t = -2\,\kappa.$$

They show how this flow provides good experimental results for noise removal with edge preservation, and give a FD (finite difference) algorithm to implement it using the Sobel operator for the evaluation of derivatives.

- Finally, Kimmel, Malladi and Sochen [KMS97, SKM98] proposed a framework for non-linear diffusion where equations are derived by minimizing a functional. Using the extended Polyakov action, which reduces to the surface area functional for 2D greyscale images, they obtained the Beltrami operator as the associated parameterization-independent Euler-Lagrange equation. To introduce an edge preserving flow, they proposed the following technique, called Beltrami flow:

$$I_t = -\Delta_g S \cdot \mathbf{e}_z = \frac{1}{\mathcal{W}}\kappa$$

where $\mathbf{e}_z$ is the unit vector in the z (intensity) direction. We will come back to this derivation in more detail in Section 3.2.1, as our derivation follows similar lines, although resulting in a different flow.

2.3 Discussion

In all previous methods, the mean curvature plays a central role. Curvature normal flow has a known connection to *surface minimization*: Lagrange already noticed that $\kappa = 0$ is the Euler-Lagrange for the surface area functional [DHKW92], meaning that the curvature normal flow minimizes surface area. But, to the authors' knowledge this property has not been used to derive a robust numerical scheme. We present in this paper both a geometrically-sound denoising flow based on surface area minimization and a discrete integration scheme following the geometric interpretation of the flow in a natural way. The next section explains the foundations of our new approach using greyscale images, while the rest of the paper will extend this method to color images and higher dimensional data.

3 Denoising of Greyscale Images

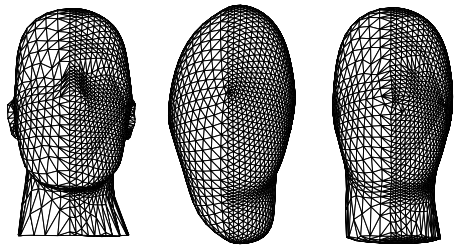
In this section, we present our first contribution in which we carefully derive a way to denoise greyscale images using surface considerations. We will show how it relates to previous work, and will demonstrate that this approach has desirable properties. Although we restrict ourselves to greyscale images in this section, the following derivation will be the backbone of our generalization to higher dimensional data.

3.1 Denoising Flow of a 3D surface

For better insight, we first explore the different methods to denoise a 3D surface. As we are about to see, crucial geometric properties have to be satisfied to achieve accurate results.

3.1.1 Curvature Flow of Arbitrary 3D Meshes

Recent work in computer graphics has demonstrated the efficiency of the mean curvature flow in removing undesirable noise from arbitrary 3D meshes [DMSB99]. Creating high-fidelity computer graphics objects using imperfectly-measured data from the real world requires an adequate smoothing technique. Earlier smoothing techniques, using a simple laplacian diffusion of the mesh, introduced large distortions in the geometry. In contrast, following the mean curvature normal $2\kappa\mathbf{n}$ at each vertex of the surface is robust with respect to differences in sampling rate, as even highly irregular meshes can be smoothed appropriately (see Figure 3(c)). Even if diffusion is a close relative of curvature flow in terms of differential equations, practical experience demonstrates undeniable advantages in using curvature flow over simple diffusion.



(a) (b) (c)

Figure 3: *Smoothing an irregularly sampled mesh: (a) Initial mesh, (b) Result of a laplacian smoothing assuming regular parameterization, (c) Result of a mean curvature flow as described in [DMSB99].*

3.1.2 Smoothing Shape vs. Parameterization

The reason for the superior performance of curvature flow over diffusion is the *parameterization-independent* nature of the Laplace-Beltrami operator. The laplacian of a mesh is always described with respect to a particular parameterization. The same geometric surface defined using another parameterization will result in a different laplacian vector. However, the notion of normal vector to a surface, or of mean curvature, is purely geometric, and as such, does not depend on the parameter space. The directions of the Laplace operator and of the Laplace-Beltrami actually coincide for the *conformal* parameter space [DHKW92]. This allows us to interpret the mean curvature normal $2\kappa\mathbf{n}$ as a special laplacian: it is a laplacian for a parameter space naturally *induced by the surface itself*.

These arguments explain a posteriori why any component in the tangent plane during the smoothing process would create distortion or local alteration of the shape of the triangulation [DMSB99]. Another insight into the nature of the tangent component can be gained if one considers a locally flat piece of a tesellated surface: any tangential component will introduce undesirable drift of the tesselation. The standard laplacian in effect fairs the parameterization of the surface as well as the shape itself. On the other hand, a purely geometric (i.e., parameterization independent) smoothing will produce the intended result of *smoothing only the shape* (notice that in Figure 3, the shapes of the initial triangles are preserved in the smoothed version).

3.2 Edge-Preserving Denoising

Starting with the denoising technique described above, we can easily introduce weights on vertices in order to preserve steep slopes of the height surface, important characteristics of the original image. Particular care must be taken to preserve the parameterization independent nature of the flow. This section explains how simple geometric considerations can be used to create a parameterization-independent, scale-invariant, edge-preserving smoothing.

3.2.1 Beltrami Flow

In the context of images, edges (i.e., sudden intensity changes) are fundamental. In denoising, any edge or boundary between different objects should be mainly preserved, while almost homogeneous regions should be smoothed quickly. Since edges in the image result

in cliffs in the z direction for the corresponding height field, a first idea is to define the following flow (see Figure 4(a)):

$$I_t = 2\kappa \mathbf{n} \cdot \mathbf{e}_z$$

The normal to the surface near edges will be almost parallel to the screen, leading to little smoothing in those regions while more uniform regions will be denoised as before. Remembering Equ. (2) and (3), we find:

$$\mathbf{n} \cdot \mathbf{e}_z = \frac{1}{\mathcal{W}}, \tag{4}$$

At this point, we recognize the Beltrami flow [KMS97]: $I_t = 2\kappa/\mathcal{W}$. However, in the rest of this section, we construct a more general approach by deriving a feature-preserving flow step by step.

3.2.2 Graph Flow

As we have briefly seen in Section 2.3, the Laplace-Beltrami operator is in a direction which minimizes surface area. Unfortunately, in the context of images, we can not easily "move" the sample points along the normal direction as it is generally not aligned with the image parameter directions. We would have to re-sample the surface back onto the pixel grid. Producing a *geometrically-equivalent flow* by only evolving the intensity field (therefore, constraining the sampling to remain the same) is then easier, and significantly faster. We will now introduce such a flow, referred to as "graph flow."

Suppose we have a surface $S(t)$ evolving in time, starting with a shape S_0. Let us define a potential $f(\mathbf{X}(t), t)$ in space such that the zero isosurface of f corresponds to S at every time t. As the evolving potential characterizes a moving isosurface, we can derive a simple differential equation satisfied by f. The path of a point $\mathbf{X}(t)$ during the evolution of the surface satisfies $f(\mathbf{X}(t), t) = 0$ for any time t, yielding:

$$\frac{\partial f}{\partial t}(\mathbf{X}(t), t) + \nabla f(\mathbf{X}(t), t) \cdot \frac{d\mathbf{X}(t)}{dt} = 0 \tag{5}$$

Note that with this equation (the typical PDE used in the level-set literature) only the normal component of $d\mathbf{X}(t)/dt$ matters since it is dotted with the gradient of f, which is along the normal to the surface. An important consequence is that *only the normal component of a surface flow really affects the shape*: since any tangential component will not be accounted for in the PDE, the potential f will only evolve according to the normal component. Adding an arbitrary tangent component to a flow field will not perturb the evolution of a surface, just modify its parameterization (as mentioned in Section 3.1.2). The preceding remark allows us to construct different particle paths that lie on the same surface family. Note that in the case of mesh smoothing, we did not wish to allow the vertices of triangulated meshes to slide. Indeed, if the triangle vertices slide tangentially in the same surface family, the interior points of the triangle faces are not guaranteed to remain close to the surface: the triangles could cut deeply across the surface.

Since we want to obtain a mean curvature flow, the graph flow needs to match the mean curvature flow after projection onto the normal, as depicted in Figure 4(a). Since $\mathbf{e}_z$ projected onto the normal introduces a factor $1/\mathcal{W}$ (see Equ. (4)), we can obtain the equivalent graph flow:

$$I_t = -2\mathcal{W}\kappa \tag{6}$$

This flow is the exact geometric equivalent of the mean curvature flow, but adapted for graphs (equivalent to the usual flow, followed by a re-sampling of the surface at the original pixel locations). Consequently, it satisfies the property of parameterization independence.

However, this flow still behaves inappropriately for height fields since edges will be smoothed significantly. No less it is an interesting anisotropic smoothing operator compared to standard laplacian smoothing (as noted by [MS96]).

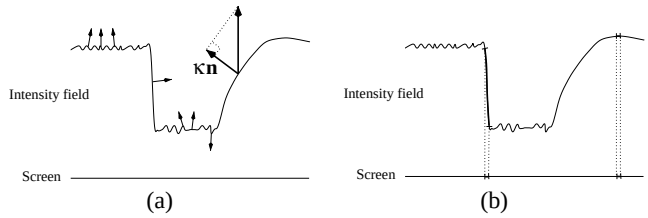


Figure 4: *(a): The left side indicates how normals are perpendicular to the screen in homogeneous, noisy areas, while parallel to the screen plane for edges. The right side shows how the graph flow is built out of the mean curvature flow by having the same magnitude once projected along the normal. (b): $\mathcal{W}$ measures the surface expansion between the parameter space (screen pixel) and the surface of the height field.*

3.2.3 Edge-Preserving Weighting

To further improve this flow and make it edge-preserving, we can now use a smoothing weight, dependent on the metric of the surface, in order to penalize the edges more than the flat regions. This corresponds to the soft constraints smoothing technique developed in [DMSB99], but this time, the appropriate smoothing factors are computed automatically, instead of being hand-chosen by the user.

Consider the term $\mathcal{W}$ (square root of the determinant of the surface metric): it measures the surface expansion between the parameter space (screen) and the surface itself (intensity field considered as a height field). Therefore, this term will be infinite along edges, while equal to one in flat regions as depicted in Figure 4(b). Its inverse is therefore a good candidate for an edge "indicator". This holds for any positive power of $\mathcal{W}$ as well. Since $\mathcal{W}$ is unitless this edge indicator is also scale-invariant. The complete edge-preserving flow can now be expressed as:

$$I_t = -\frac{2\kappa}{\mathcal{W}^\gamma} \tag{7}$$

The coefficient $\gamma \geq 0$ determines the relative penalization of small jumps in intensity versus large jumps. Values less than one only penalize large jumps, while values larger than one penalize even small jumps. It controls the linearity of our edge-preservation metric: as such, γ can be described as an *edge contrast parameter*.

The flow derived above is quite general. For $\gamma = 0$, we find the same flow used by El-Fallah and Ford [FEF98]. For $\gamma = 1$, our formulation leads to the Beltrami flow, mentioned in Section 2.2. Other values of γ offer a whole new family of denoising flows, all having the properties of parameterization-independence, scale-invariance, and feature-preservation. In the next section, we propose to generalize the above derivation to nD data. In Section 5 we will present a natural and robust numerical scheme for this PDE, which will preserve the surface area minimization nature of the flow.

4 Denoising of Arbitrary Bivariate Data

Two-dimensional data often has more than one channel of information. Color images for instance have three channels per pixel: red, green, and blue. Although a straightforward channel by channel smoothing is easily achieved by the previous method, it may not lead to optimal smoothing. Independent changes in the red, green, and blue channels result in perceptually-strong color variations in the smoothed image. Therefore, smoothing in color should be performed in the rgb space where coupling between channels results in more natural color smoothing [Sha96]. Similarly, higher dimensional data should be smoothed in its respective space, not channel-by-channel. This section demonstrates that our previous approach

can be extended easily to provide a denoising technique for higher dimensional data.

4.1 Graph Flow for Mean Curvature Smoothing

We now consider our bivariate multi-dimensional data as lying on 2-manifold embedded in nD. We can still define the Laplace-Beltrami operator as being the generalization of the mean curvature normal, or the generalization of the (parameterization-independent) surface area gradient. For the sake of simplicity, we will denote the Laplace-Beltrami operator as $\mathbf{B}$ from now on: $\Delta_g S = \mathbf{B}$. To make this flow a graph flow, we have to project this vector onto the sub-space of free parameters, such as r, g, b in the case of color images. The orthogonal projection of $\mathbf{B}$ onto this sub-space is the vector $\overline{\mathbf{B}}$. It consists of the same coordinates as $\mathbf{B}$, except for the first two components (corresponding to the x and y axes of screen space) set to zero. Therefore, we need a vector in the direction opposite to $\overline{\mathbf{B}}$ to ensure a graph flow, but such that its projection onto $\mathbf{B}$ has the same magnitude as $\mathbf{B}$ to ensure the geometric equivalence:

$$-\frac{\mathbf{B}\cdot\mathbf{B}}{\overline{\mathbf{B}}\cdot\overline{\mathbf{B}}}\ \overline{\mathbf{B}}. \tag{8}$$

Applied to color images (5D space $(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_r, \mathbf{e}_g, \mathbf{e}_b)$), the graph flow geometrically equivalent to a mean curvature flow is therefore:

$$\frac{d}{dt}\begin{pmatrix} r \\ g \\ b \end{pmatrix} = -\frac{\mathbf{B}\cdot\mathbf{B}}{\overline{\mathbf{B}}\cdot\overline{\mathbf{B}}}\begin{pmatrix} \overline{\mathbf{B}}\cdot\mathbf{e}_r \\ \overline{\mathbf{B}}\cdot\mathbf{e}_g \\ \overline{\mathbf{B}}\cdot\mathbf{e}_b \end{pmatrix}. \tag{9}$$

4.2 Edge-Preserving Flow

Following the same arguments as in Section 3, we now want to weight the features to favor smoothing of almost uniform regions. Thus, we need to find a way to measure discontinuities. Based on the same idea as in the greyscale case, we can use the ratio of surface expansion between the screen and the surface. It is directly measured by the ratio between the magnitudes of $\mathbf{B}$ and $\overline{\mathbf{B}}$, as cliffs are characterized by a normal parallel to the screen plane. Our multi-dimensional scale-invariant edge indicator can be written as: $||\overline{\mathbf{B}}||/||\mathbf{B}||$: the edge indicator will be valued 0 on sharp edges, and 1 in homogeneous regions. Adding an edge contrast parameter γ (slightly different than the previously defined γ, purely for aesthetic reasons), our feature-preserving flow becomes, for color pictures for instance:

$$\frac{d}{dt}\begin{pmatrix} r \\ g \\ b \end{pmatrix} = -\left(\frac{||\overline{\mathbf{B}}||}{||\mathbf{B}||}\right)^{\gamma}\begin{pmatrix} \overline{\mathbf{B}}\cdot\mathbf{e}_r \\ \overline{\mathbf{B}}\cdot\mathbf{e}_g \\ \overline{\mathbf{B}}\cdot\mathbf{e}_b \end{pmatrix}. \tag{10}$$

Notice that $\gamma = 0$ simplifies greatly to a Beltrami flow. The creation of higher dimension feature-preserving smoothing flows follows naturally.

4.3 Incorporating Perceptual Bias for Color

The (r, g, b) color space is not necessarily the most perceptually sound. Put simply, the human eye is not similarly sensitive to a change of red, green, or blue: what we visibly consider as a major color edge may not be considered as such in this color space, and vice-versa. Therefore, smoothing a color image in such a space may not lead to the most pleasant visual results.

Instead, we use the (L^*, U^*, V^*) color space to take some of the human color perception biases into account. This model has the advantage of being almost perceptually uniform for the human eye, and therefore, will appropriately define edges. Note that any other model and/or linear combination of existing models is straightforward to implement in our framework as only the input has to be changed.

4.4 Tuning of Global Contrast

The framework defined so far has an additional degree of freedom: the scaling of intensity/colors. Colors are usually rescaled between 0 and 1, but the real color spectrum of the image is undetermined. Unless radiometric values of the image are available, we can arbitrarily choose a scale factor α to define the global contrast of the image. Note that our surface functional for a large value of α will be equivalent, for $\gamma = 0$, to a regularized version of the L_1 norm of the intensity: therefore, our flow will be equivalent to the total variation denoising approach of [ROF92]. On the other hand, a small scale factor will tend to create a flow based on the L_2 norm [Sha96] for the same γ [KMS97].

4.5 Discussion

We have defined a scale-invariant anisotropic flow to denoise any bivariate data while preserving features. It is based on surface area minimization, well-known in 3D to provide good denoising. As this method tends to minimize surface area in nD, the smoothing between data samples is treated in a non-linear way, significantly different from a channel-by-channel smoothing. In the special case of color images, color smoothing will induce an alignment of the gradient of each channel, which does not appear in a channel by channel smoothing. Our method also applies to higher dimensional data such as tensor or vector fields, providing an interesting framework to simplify complex fluid flows or MRI tensor images. However, a discrete implementation must be defined in order for the method to be practical. The next section addresses this point.

5 Implementation of our Flow

We now turn our attention to a practical and robust implementation of our feature-preserving flow. In this section we will derive a simple discrete form of the PDE and use it to reliably integrate the flow in time. The discretization is designed to preserve the surface area minimization nature of the flow.

5.1 Usual Implementation with FD

One can approximate the mean curvature directly by finite differences. Since the mean curvature is defined as:

$$\kappa = \frac{I_{xx}(1+I_y^2) - 2I_{xy}I_xI_y + I_{yy}(1+I_x^2)}{(1+I_x^2+I_y^2)^{3/2}},$$

a quick FD implementation can be derived for greyscale images. Kimmel [KMS97] also derived a FD numerical scheme for color images, but the complexity of the computations involved increases rapidly with the dimensionality of the data. Moreover we prefer to use a more natural discretization of the mean curvature, since it will guarantee good behavior as the discrete operator mimics the continuous case perfectly. We will also see that it can easily be extended to nD with a modest computational overhead.

5.2 Definition of Mean Curvature Normal

In differential geometry, the mean curvature normal is sometimes described as a geometric property of a surface. Around a point $\mathbf{P}$, the limit of the surface area variation with respect to $\mathbf{P}$ as we take

a smaller and smaller piece of surface turns out to be the mean curvature at **P**. Therefore, the mean curvature normal can be defined through the following property [DMSB99]:

$$2\kappa\mathbf{n} = \nabla\mathcal{A}/\mathcal{A}, \tag{11}$$

where $\mathcal{A}$ is a small area around **P**, and ∇ is the derivative with respect to **P**. Similarly, the vector **B**, generalizing the mean curvature normal in nD, can be expressed as:

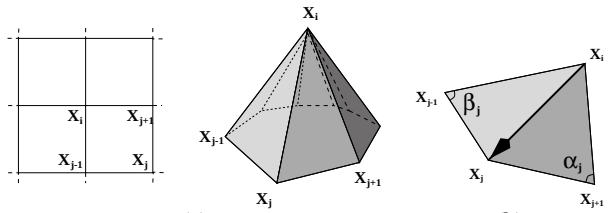
$$\mathbf{B} = \nabla\mathcal{A}/\mathcal{A}. \tag{12}$$

5.3 Definition of a Robust Discrete Operator in 3D

In [DMSB99], the authors showed a formulation of the surface area gradient of a piecewise linear 3D surface approximation (i.e., a triangle mesh) with respect to a given vertex. A direct derivation of the continuous case to the discrete case yields the formula:

$$\nabla\mathcal{A} = \frac{1}{2}\sum(cot\ \alpha_{ij} + cot\ \beta_{ij})(\mathbf{X}_i - \mathbf{X}_j), \tag{13}$$

where $\mathbf{X}_i$ is a vertex of the mesh, $\mathbf{X}_j$ an immediate neighbor, and α_{ij}, β_{ij} the two opposite angles to the edge $\mathbf{X}_i\mathbf{X}_j$, as sketched in Figure 5. Notice that this gradient is zero for any flat piece of surface, regardless of the shape or the number of triangles in it.

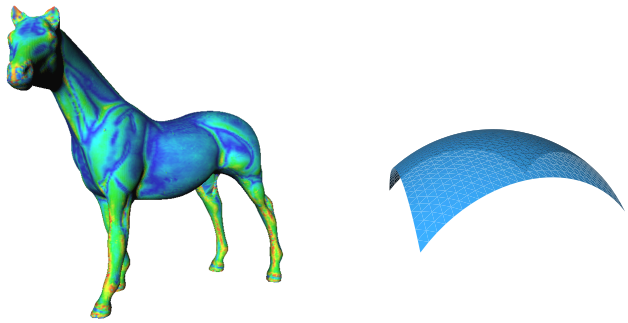


(a) (b)

Figure 5: *A vertex x_i and its neighbors on the screen and on the surface (a), and one term of its curvature normal formula (b).*

The discrete operator has been proven robust and reliable even on irregular meshes. Fig. 6(a) and 6(b) exhibit two irregular meshes and their curvature plot using the previously discussed discrete curvature normal derivation. We observe a significantly reduced amount of noise compared to previous methods of approximating the mean curvature.

We believe that the good properties of this discrete form of the mean curvature are due to the preservation of the fundamental property of the operator: surface area minimization. This formulation is *conservative* in that sense. Hence, such a discretization will provide better results than regular FD schemes in the context of image denoising since area minimization is involved. Moreover, the extension to higher dimensional data spaces is straightforward as we are about to see.



(a) (b)

Figure 6: *Curvature plot (pseudo-colors representing magnitude of mean curvature normal) of two irregular meshes, proving the robustness of the discrete operator. (a) Model of a horse, and (b) piece of a unit discrete sphere, where the mean curvature approximation is equal to 1+/-2%*

5.4 Discrete Beltrami Operator in High Dimensions

Since the previous formulation for the surface area gradient is only valid in 3D, we must start with an extension for higher dimensional data spaces. For generality's sake, we will treat the nD case, and the color image case will only be a particular example.

5.4.1 Surface Area in nD

First, we must derive the expression for a surface area in nD. The area of a triangle formed by two vectors **u** and **v** in 3D is $2\mathcal{A} = ||\mathbf{u}\times\mathbf{v}||$. Being proportional to the sine of the angle between vectors, we can also express it as:

$$\begin{aligned}\mathcal{A}(\mathbf{u},\mathbf{v}) &= \frac{1}{2}||\mathbf{u}||||\mathbf{v}||sin(\mathbf{u},\mathbf{v}) = \frac{1}{2}||\mathbf{u}||||\mathbf{v}||\sqrt{1-cos^2(\mathbf{u},\mathbf{v})} \\ &= \frac{1}{2}\sqrt{||\mathbf{u}||^2||\mathbf{v}||^2 - (\mathbf{u}\cdot\mathbf{v})^2}.\end{aligned}$$

This latter expression is now valid in nD, and is particularly easy to evaluate in any dimension.

5.4.2 Derivation of the Area Gradient

Given the expression for the surface area of a triangle, we can formally derive the gradient of the area with respect to one of its vertices. We refer the reader to the appendix for the detailed derivation of the formula. It is shown there that the cotangent Equ. (13) is actually still valid *if we extend the definition of cotangent in nD as being*:

$$cot(\mathbf{u},\mathbf{v}) = \frac{cos(\mathbf{u},\mathbf{v})}{sin(\mathbf{u},\mathbf{v})} = \frac{\mathbf{u}\cdot\mathbf{v}}{\sqrt{||\mathbf{u}||^2||\mathbf{v}||^2 - (\mathbf{u}\cdot\mathbf{v})^2}}.$$

With this definition, the implementation in nD space is straightforward and efficient, as dot products require little computation.

5.5 Practical Implementation for Denoising

The implementation of our scheme is now straightforward with the discrete operator we have just described. We will explicitly give the discretization for $\gamma = 0$ since these formulae are particularly simple. In the case of greyscale images, we change the intensity value I_i for every pixel according to:

$$\frac{dI_i}{dt} = \frac{1}{2\mathcal{A}}\sum(cot\ \alpha_{ij} + cot\ \beta_{ij})(I_j - I_i), \tag{14}$$

where the total area $\mathcal{A}$ around a vertex is just the sum of the areas of all the triangles adjacent to this vertex. As in FD schemes, we sum the contribution for all eight immediate neighbors. The cotangent is implemented efficiently by computing the two adjacent 3D vectors forming the angle considered, and computing their dot product divided by the norm of their cross product.

For color images our edge-preserving flow becomes:

$$\frac{dr_i}{dt} = \frac{1}{2\mathcal{A}}\sum(cot\ \alpha_{ij} + cot\ \beta_{ij})(r_j - r_i)$$

$$\frac{dg_i}{dt} = \frac{1}{2\mathcal{A}}\sum(cot\ \alpha_{ij} + cot\ \beta_{ij})(g_j - g_i)$$

$$\frac{db_i}{dt} = \frac{1}{2\mathcal{A}} \sum (cot\ \alpha_{ij} + cot\ \beta_{ij})(b_j - b_i)$$

Note that the coupling between channels is incorporated in the cotangents. For data of different dimensionality the "feature-preserving" flow will be very similar to this previous set of equations.

5.6 Integration of the Flow

The implementation of the flow is done by integrating the last set of PDEs in time using either an explicit or implicit Euler scheme. The user can stop the smoothing when the data is sufficiently denoised. El-Fallah and Ford proposed an improvement on the integration by computing the time step to use according to the variation of the global area [FEF98]. Indeed, if the area of the whole image changes significantly during a time step, a lot of noise was present in the image, and it is safe to take a larger time step. When the area change starts to decrease, the image structure may be significantly affected by too large a time step, thus the time step should be reduced.

5.7 Discussion

We have derived a new discrete version of our differential denoising model. Building upon a classic curvature normal flow, we weight this smoothing process in order to preserve significant features of the data while still suppressing high frequency noise. This anisotropic smoothing flow is then implemented in a discrete setting with simple discrete-geometry tools. This numerical technique has two main advantages: it preserves the nature of the flow in a discrete way, and is easy to implement in any dimension. By varying the exponent γ, we can re-create existing flows or create entirely new flows. Additionally, we can easily extend these flows to arbitrary dimensions. The next section presents our first results obtained with the above numerical scheme.

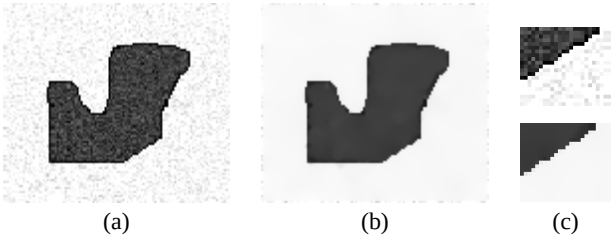
6 Results

We tested our method on several datasets. We first used computer generated images with artificially added noise. In Figure 7(a-c) we can see that our method removes the noise from a simple greyscale image while retaining the edges present in the original image. Similarly, Figure 7(d-e) shows a smoothing for a simple color picture in the presence of large amounts of noise.

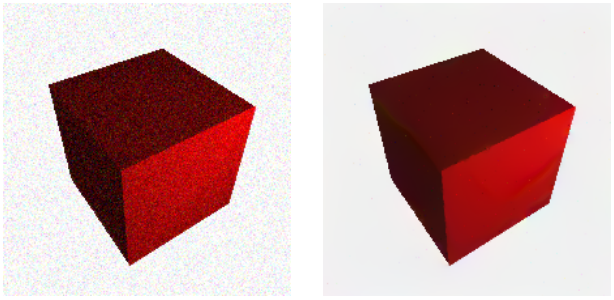
Next, we tested the method on "real world" images. The denoising technique performs well on classical test images, as demonstrated for instance in Figure 8. In Figure 10, we display a noisy image of a clock and its restored version, along with the height field representation of the images.

We also tried our technique on different depth data. Rather than using a 3D smoothing as in [DMSB99], we can take advantage of the fact that the error is only in the z direction. While former methods [Tau95, DMSB99] would make the assumption of an isotropic noise in space, our method applies better to this depth field as the noise (measurement error) mainly resides along the z axis. To demonstrate this advantage, we smoothed an elevation map of a section of Mars. Due to measurement errors and poor quantization of the original data, the height field is noisy as shown in Figure 1(a). After an anisotropic smoothing, we suppress the noise and most of the quantization effects, resulting in a smooth surface even with a flat-shaded rendering.

Finally, we demonstrate how our method behaves on range images in Figure 9. Given a noisy range image of a face, we can smooth the range image to reconstruct the face without visible noise while keeping the features in place. Once again, previous methods would have altered the shape since the assumption of isotropic noise in the data does not apply for range images.

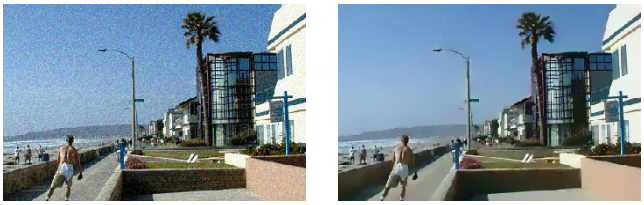


(a) (b) (c)



(d) (e)

Figure 7: *Examples of denoising for computer-generated greyscale and color images (a&d: noisy images, b&e: denoised output, c: close-up of a and b).*



(a) (b)

Figure 8: *(a) Noisy color image, (b) Denoising flow applied to (a), in 300 explicit iterations with* $dt = 1$, $\gamma = 0$.

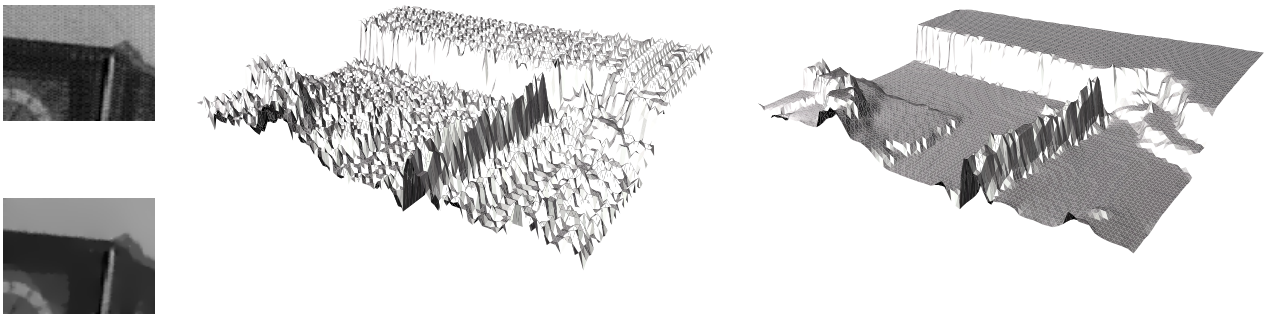
7 Conclusions and Future Work

We presented a general framework for denoising of 2D data. Derived from smoothing of meshes in 3D, our approach proposes an anisotropic diffusion of data with convenient features: our method is robust, stable, feature-preserving, scale-invariant, and easily implemented for greyscale and color images, but also any forms of multi-channel bivariate data. We demonstrated how this approach provides an elegant way to smooth height fields and range images by taking into account the way these data were acquired: knowing that the noise is mainly in the depth approximation, our method provides more accurate results than previous 3D smoothing algorithms where the noise is treated as isotropic in space.

Our current work includes more testing of these previous denoising flows for different form of data (such as tensor images from MRI data), even for volumetric data, and irregular grids. We are also working on a denoising flow based on principal curvatures for the same applications.

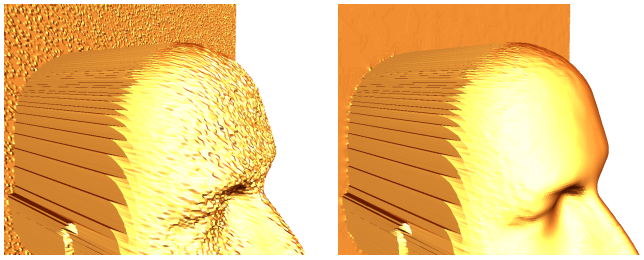
Acknowledgments

The authors are thankful to Pierre Kornprobst for help in image processing; Andrei Khodakovsky for general assistance; Martin Rumpf for important insights; JPL for the Mars topography; and the anonymous reviewers for many helpful comments. This work was supported by NSF DMS-9872890, ACI-9721349, DMS-9874082, the Packard Foundation, Alias|Wavefront, the STC, and by the Academic Strategic Alliances Program of the Accelerated Strategic Computing Initiative (ASCI/ASAP) under subcontract B341492 of DOE contract W-7405-ENG-48.



(a) (b) (c)

Figure 10: *Clock example: The initial image (a, top) contains a significant amount of noise as its height field (b) shows. Our denoising technique significantly reduces this amount of noise (a, bottom) while keeping the features in place (c).*



(a) (b)

Figure 9: *(a) Head model obtained from a noisy depth image. (b) Reconstructed model after denoising (flat-shaded).*

References

[ABBFC97] G. Aubert, M. Barlaud, L. Blanc-Feraud, and P. Charbonnier. Deterministic edge-preserving regularization in computed imaging. *IEEE Trans. Imag. Process.*, 5(12), February 1997.

[ALM92] L. Alvarez, P-L. Lions, and J-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion (II). *SIAM Journal of numerical analysis*, 29:845–866, 1992.

[Bar89] Alan H. Barr. The Einstein Summation Notation: Introduction and Extensions. In *SIGGRAPH 89 Course notes #30 on Topics in Physically-Based Modeling*, pages J1–J12, 1989.

[DHKW92] Ulrich Dierkes, Stefan Hildebrandt, Albrecht Küster, and Ortwi Wohlrab. *Minimal Surfaces I*. Grundlehren der mathematischen Wissenschaften, Springer-Verlag, 1992.

[DMSB99] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan Barr. Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow. In *SIGGRAPH 99 Conference Proceedings*, pages 317–324, August 1999.

[FEF98] G. Ford and A. El-Fallah. On mean curvature in non-linear image filtering. *Pattern Recognition Letters*, 19:433–437, 1998.

[Gra98] Alfred Gray. *Modern Differential Geometry of Curves and Surfaces with Mathematica*. CRC Press, 1998.

[KDA97] P. Kornprobst, R. Deriche, and G. Aubert. Nonlinear operators in image restoration. In *CVPR'97*, pages 325–331, Puerto-Rico, 1997.

[KMS97] R. Kimmel, R. Malladi, and N. Sochen. Images as embedding maps and minimal surfaces: Movies, color, and volumetric medical images. In *IEEE CVPR'97*, pages 350–355, 1997.

[MS96] R. Malladi and J.A. Sethian. Image processing: Flows under min/max curvature and mean curvature. *Graphical Models and Image Processing*, 58(2):127–141, March 1996.

[PM90] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.

[PR99] T. Preußer and M. Rumpf. Anisotropic Nonlinear Diffusion in Flow Visualization. In *IEEE Visuazliation'99*, pages 323–332, 1999.

[ROF92] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[Sha96] J. Shah. Curve evolution and segmentation functionals: Applications to color images. In *IEEE ICIP'96*, pages 461–464, 1996.

[SKM98] N. Sochen, R. Kimmel, and R. Malladi. A geometrical framework for low level vision. *IEEE Trans. on Image Processing*, 17(3):310–318, 1998.

[Tau95] Gabriel Taubin. A Signal Processing Approach to Fair Surface Design. In *SIGGRAPH 95 Conference Proceedings*, pages 351–358, August 1995.

[TT99] Jack Tumblin and Greg Turk. LCIS: A Boundary Hierarchy For Detail-Preserving Contrast Reduction. In *SIGGRAPH 98 Conference Proceedings*, pages 83–90, 1999.

Appendix: Area Minimization in nD

In this appendix, we use Einstein summation notation for conciseness. For an introduction, see [Bar89].

Consider 3 points (P, Q, R) in a space of arbitrary dimension $n > 2$. As mentioned in Section 5.4.1, we can write the area formed by the triangle (A, B, C) as follows:

$$\mathcal{A}^2 = \frac{1}{4}\left(PQ_iPQ_iPR_jPR_j - PQ_iPR_iPQ_jPR_j\right).$$

Differentiating term by term with respect to P we get:

$$\begin{aligned}
4\frac{\partial \mathcal{A}^2}{\partial A_k} &= -\delta_{ik}PQ_iPR_jPR_j - \delta_{ik}PQ_iPR_jPR_j \\
&\quad -\delta_{jk}PQ_iPQ_iPR_j - \delta_{jk}PQ_iPQ_iPR_j \\
&\quad +\delta_{ik}PR_iPQ_jPR_j + \delta_{ik}PQ_iPQ_jPR_j \\
&\quad +\delta_{jk}PQ_iPR_iPR_j + \delta_{jk}PQ_iPR_iPQ_j \\
&= -2PQ_kPR_jPR_j - 2PQ_iPQ_iPR_k + PR_kPQ_jPR_j \\
&\quad +PQ_kPQ_jPR_j + PQ_iPR_iPR_k + PQ_iPR_iPQ_k \\
&= 2\left[PQ_k(PQ\cdot PR - PR\cdot PR) + PR_k(PQ\cdot PR - PQ\cdot PQ)\right] \\
&= 2\left[PQ_k(QR\cdot RP) + PR_k(PQ\cdot QR)\right]
\end{aligned}$$

Additionally, we also have:

$$\frac{\partial \mathcal{A}^2}{\partial P_k} = 2\mathcal{A}\frac{\partial \mathcal{A}}{\partial P_k}$$

Therefore, using Equ. 14, and if we define the cotangent of an angle between two nD vectors $\mathbf{u}$ and $\mathbf{v}$ as:

$$cot(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}\cdot\mathbf{v}}{\sqrt{||\mathbf{u}||^2||\mathbf{v}||^2 - (\mathbf{u}\cdot\mathbf{v})^2}},$$

the gradient of the surface area can be expressed exactly as in Equ. (13), extending nicely the 3D case to nD.