Acta Math. Univ. Comenianae
Vol. LXX, 1(2001), pp. 15–31
Proceedings of Algoritmy 2000

DIFFUSION MODELS AND THEIR ACCELERATED SOLUTION IN IMAGE AND SURFACE PROCESSING

U. DIEWALD, T. PREUSSER, M. RUMPF AND R. STRZODKA

Abstract. During the last decade nonlinear anisotropic diffusion models have shown to be powerful methods not only in image processing. Moreover these methodologies can be adopted to other areas in computer vision. On the level of the continuous model one can study the qualitative aspects and properties whereas approved discretization schemes are at hand for an efficient implementation. In this paper we discuss several anisotropic diffusion methods and outline a novel technique for geometric surface processing. Moreover we will show how the solution process can be accelerated significantly by using texture hardware of modern graphics cards, making use of the much better memory bandwidth and the built-in parallelism.

1. Introduction

In the last decade PDE based models in the field of computer vision and image analysis became very popular. Here we outline how these techniques can be generalized to surface processing and visualization purposes. Furthermore, we introduce a novel concept for fast implementation of the underlying algorithms.

The nonlinear diffusion models as we know them today were first introduced by a work of Perona and Malik [**23**], who introduced a model that allows for denoising of images while retaining and enhancing edges. Analysis of the Perona Malik model showed its mathematical ill-posedness [**18, 19, 35**], that drove the derivation of a regularized model by Cattè et. al. [**7**]. This still has the edge preserving property as long as the regularization parameter is chosen appropriately. The so called scale space methods were later classified by a rigorous axiomatic theory by Alvarez et. al. [**2**]. Recovering of lower dimensional structures was analyzed by Weickert [**31**]. He considered an anisotropy depending on the so called structure tensor of images, that steers a nonlinear diffusion process taking care of tangential and normal directions on edges.

But the continuous diffusion models are not only limited to image processing. In [**24**] Preußer and Rumpf have adopted the nonlinear diffusion to visualization of vector fields – a fundamental topic in scientific visualization. Clarenz, Diewald and Rumpf picked up the idea of nonlinear diffusion and incorporated anisotropic

Received January 10, 2001.
2000 *Mathematics Subject Classification.* Primary 65M60, 65Y10.
Key words and phrases. Hardware accelerated solver, image processing, level set method, multiscale, nonlinear diffusion, surface processing, vector field visualization.

surface smoothing that retains edges on surfaces and whose diffusion tensor depends on the shape operator of the evolving surface [**8**]. Their methodology also enables visualization of vector fields on surfaces [**12**]. All the derived models profit from important properties from the original diffusion model in image processing.

Finite element methods are widely spread to discretize the underlying partial differential equations. Their convergence properties were first studied by Kačur and Mikula [**17**]. Weickert [**31**] proposed additive operator splitting techniques based on finite difference schemes to accelerate the solution process. Also multigrid methods have been considered in the past [**1**]. They allow for a good convergence as long as the anisotropic behavior of the diffusion process is not too strong. The use of adaptive finite elements has been discussed in [**4, 25**]. Moreover parallel computing has been considered in e.g. [**32**].

An alternative to the parallelization in conventional hardware is the use of modern graphics cards for computations. This has been extensively exploited in volume rendering [**9, 33, 34**] and more recently in filter applications [**15, 16**]. The advantages are mainly the very fast memory access and inherently parallel processing of data. In this paper we show how the graphics processor unit may be even used for the more complex task of solving linear systems of equations which arise in finite element schemes. This finally allows the whole application to run solely on the graphics board.

The paper is organized as follows: In Section 2 we review the general diffusion model in image processing and show various applications and adaptations to computer vision in section 3. Here we revisit the model for visualization of vector fields on Euclidean domains and on surfaces. An anisotropic parametric surface processing model and also a new geometric level set method for geometric image smoothing will be presented. In section 4 we discretize the diffusion models using finite elements. Finally we discuss the hardware acceleration in Section 5.

2. Diffusion Models in Image Processing

To begin with, let us briefly review the standard nonlinear diffusion model in image processing. We consider images as functions on a given domain $\Omega \subset \mathbb{R}^d$ having values in $\mathbb{R}^m$ with $m \in \{1, 2, 3\}$. For gray value images we clearly have $m = 1$, whereas the case $m = 2, 3$ corresponds to color images. Our image domain $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$ corresponds to 2D images in the case $d = 2$ and to 3D images that are widespread in medical applications in the case $d = 3$. The basic nonlinear diffusion problem then reads:

Find a function $\rho\colon \mathbb{R}_0^+ \times \Omega \to \mathbb{R}^m$ *which solves the parabolic problem*

$$\begin{aligned} \partial_t \rho - \operatorname{div}\left(A(\nabla \rho_\sigma)\nabla \rho\right) &= f(\rho) \text{in } \Omega, \\ \rho(0,\cdot) &= \rho_0 \quad \text{on } \partial\Omega, \\ A(\nabla \rho_\sigma)\nabla \rho \cdot \nu &= 0 \quad \text{ on } \mathbb{R}^+ \times \partial\Omega, \end{aligned} \tag{2.1}$$

for given initial density $\rho_0\colon \Omega \to \mathbb{R}^m$. Here $\rho_\sigma = \chi_\sigma * \rho$ is a convolution of the current density, which is necessary for the wellposedness of the above parabolic, boundary and initial value problem (cf. [**7**]). The function $f(\cdot)$ may serve as a penalty which forces the solution to stay close to the initial image, e. g. by choosing $f(\rho) = \gamma(\rho_0 - \rho)$ where γ is a positive constant. We regard the solution $\rho(\cdot)$ as a family of images $\{\rho(t)\}_{t\in\mathbb{R}_0^+}$, where the time t acts as a scale parameter. The parabolic equation can then be seen as a filter which for each $t \in \mathbb{R}^+$ delivers a filtered version $\rho(t)$ of the original density ρ_0.

So far we have not yet defined the nonlinear diffusion tensor $A(\nabla\rho_\sigma)$ that steers the evolution process. In the following sections we will define $A(\nabla\rho_\sigma)$ in different ways, leading to the desired models. Replacing the diffusion tensor with the identity we would get the basic heat equation model. Whereas setting $A = G(\|\nabla\rho_\sigma\|)$, leads to the regularized Perona Malik model with an edge indicator function $G\colon \mathbb{R}_0^+ \to \mathbb{R}^+$. We suppose G to be a monotone decreasing function chosen such that $\lim_{d\to\infty} G(d) = 0$ and $G(0) = \beta$ where $\beta \in \mathbb{R}^+$ is constant, e. g. $G(d) = \frac{\beta}{1+\|d\|^2/\lambda^2}$, with $\lambda > 0$. As already mentioned above the convolution avoids the non-wellposedness of the model. A widespread choice is the convolution with a Gaussian kernel [**7**], which corresponds to the solution of the heat equation evaluated at a corresponding short time. Figures 1 shows the application of a regularized Perona Malik smoothing process in 2D.

Figure 1. Different scale steps of the Perona Malik evolution of a noisy image are being shown. From left to right the noisy image, respectively the scale-steps 3, 7 and 10 are depicted.

In contrast to the nonlinear diffusion model presented so far another but purely morphological model has become fundamental in image processing. It is based on simultaneous evolution of all level sets $\mathcal{M}^c := \{x \in \Omega \big| \rho(t,x) = c\}$ of an image by mean curvature motion (cf. [**3**])

$$\partial_t\rho - \|\nabla\rho\| \operatorname{div}\left(\frac{\nabla\rho}{\|\nabla\rho\|}\right) = 0. \tag{2.2}$$

I.e. we consider the parabolic equation which is the level set variant of $\partial_t x = -H(x)N(x)$, where $H(x)$ is the mean curvature and $N(x)$ the normal to the levelset curve at a point x. From differential geometry we know that for a distinct surface $\mathcal{M} := \mathcal{M}^c$ this surface propagation is equivalent to solving

$$\partial_t x - \Delta_{\mathcal{M}} x = 0,$$

where $\Delta_{\mathcal{M}}$ denotes the Laplace Beltrami operator on a surface $\mathcal{M}$ and x is the identity on $\mathcal{M}$.

3. Applications in Computer Vision

In this section we will study various applications for which we apply the diffusion models. We will first discuss the modification to anisotropic diffusion in vector field visualization, and then carry over the multiscale concept to surface processing. After the extension of the anisotropic diffusion to visualization of vector fields on surfaces, we proceed to a new geometric level set method.

3.1. Vector Field Visualization on Euclidean Domains

The central goal in this area of scientific visualization is to define methods that allow for an intuitive reception and give an overall as well as detailed view of the underlying flow field. The simplest method of drawing vector plots often leads to visual clutter, due to the different local scaling of the field within the domain. Many other techniques have therefore been derived. For a detailed review of related work we refer to [**8, 12, 24**].

We want to define a texture that represents the flow field globally on the spatial domain, since single particle lines only very partially are capable of illuminating features of a complex flow. Our method to be presented generates streamline like patterns and in addition carries the possibility to successively coarsen those patterns. We base our method on the coherence enhancing filters from [**31**] and furthermore pick up the line integral convolution (LIC) approach as proposed by Cabral and Leedom [**6**] using the observation that the built in convolution along streamlines corresponds to solving the heat equation on the streamlines. The desired coarsening will be steered by a Perona Malik type diffusion that acts in the orthogonal direction. We consider Neumann boundary conditions as before on Section 2.

To be more precise, let $v\colon \Omega \to \mathbb{R}^d$ be a given vector field, which we assume to be continuous and non vanishing on Ω. Clearly there exists a family of orthogonal mappings $B(x)\colon \Omega \to SO(n)$ such that $B(x)v(x) = e_0$, where $\{e_i\}_{i=0,\dots,d-1}$ is the standard basis in $\mathbb{R}^d$. Hence, we consider a diffusion tensor $A(v, \nabla\rho_\sigma)$ depending on the vector field v defined by

$$A(v, \nabla\rho_\sigma) = B(x)^T \begin{pmatrix} \alpha(\|v(x)\|) & 0 \\ 0 & G(\|\nabla\rho_\sigma\|)\mathrm{Id}_{d-1} \end{pmatrix} B(x).$$

Here $\alpha\colon \mathbb{R}^+ \to \mathbb{R}^+$ controls the linear diffusion in vector field direction, i.e. along streamlines, and the above introduced edge enhancing diffusion coefficient $G(\cdot)$ acts in the orthogonal directions. In general, we choose α to be a constant function, but we may also select a monotone function with $\alpha(0) > 0$ and $\lim_{\alpha\to\infty} \alpha(s) = \alpha_{\max}$. We could run the evolution with any image density as initial data ρ_0, but to avoid aliasing artefacts, we choose a random noise of appropriate frequency range as initial data. During the evolution, patterns will diffuse along the streamlines, but there is still some diffusion perpendicular to the flow field. This supplies us with coarser representations of the flow field as the scale increases. Unfortunately, if we run the evolution with a vanishing right hand side f, the contrast of the image density will decrease, because of the linear diffusion along streamlines. Thus, we select a source term $f\colon [0,1] \to \mathbb{R}^+$ satisfying

$$f(0) = f(1) = 0, \qquad f < 0 \text{ on } (0, 0.5), \qquad f > 0 \text{ on } (0.5, 1), \tag{3.1}$$

that pushes values towards zero and 1, respectively. Well known maximum principles ensure that we do not enlarge the interval of gray values using this f. Choosing $m \in \{2, 3\}$ in the diffusion equation 2.1 provides additional asymptotic states of the process. We then select the corresponding initial data randomly distributed within the cube $[0,1]^m$, interpret the components of ρ as color-components and define the force f to work on the luminance of ρ. In Figure 2 we have depicted a scale of vector field representations.

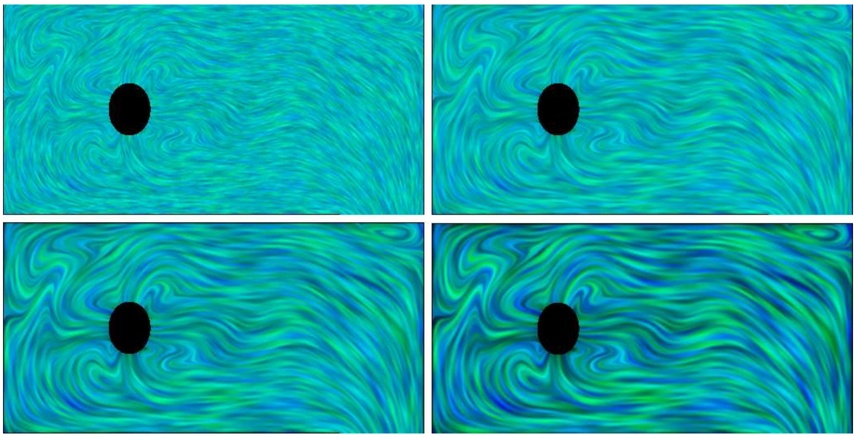

Figure 2. From top left to bottom right four successive scale-steps of the anisotropic diffusion process are depicted. The vector field visualized here results from a CFD computation, where a fluid flows from the inlet (black circle) toward an outlet on the lower right corner.

3.2. A parametric surface processing model

In this section we will review a first anisotropic diffusion model on parametric surfaces. This method extends the edge enhancing diffusion filters from [**31**] to surfaces. Other contributions to PDE based smoothing on/of surfaces have been made by [**10, 11, 20**].

To support the reading of the following sections, in Table 1 we have collected all notations used in the following. For the sake of simplicity let us assume our surfaces to be compactly embedded manifolds in $\mathbb{R}^3$ without boundary. In case of noisy parametrized surfaces $\mathcal{M}_0$ with parameterization x_0 we can proceed in analogy to diffusion in image processing and consider the corresponding anisotropic geometric evolution problem. This method first presented in [**8**] is able to preserve important features such as edges and corners on the surface and allows tangential smoothing along an edge but not in the direction perpendicular to it. The core of the method is diffusion that depends on the shape operator S, which indicates edges and corners by sufficiently large eigenvalues.

$\mathcal{TM}$	Tangent bundle on $\mathcal{M}$
$\mathcal{T}_x\mathcal{M}$	Tangent space of $\mathcal{M}$ in $x \in \mathcal{M}$
$g(\cdot,\cdot)$	Metric on $\mathcal{M}$
S	Shape operator
S^σ	Regularized shape operator
S_σ	Shape operator of a regularized image. (In general $S_\sigma \neq S^\sigma$)
$S^\sigma_{\mathcal{T}_x\mathcal{M}}$	Shape operator acting on the tangent space of $\mathcal{M}$ in $x \in \mathcal{M}$
N	Normal of a surface respectively level set
N^σ	Normal of a regularized surface respectively level set
DN	Jacobian of the normal
$X_{\mathcal{M}(t)}$	Manifold on which the differential operator X is defined

Table 1. Notations concerning geometric anisotropic diffusion.

Because the evaluation of the shape operator on a noisy surface might be misleading with respect to the original but unknown surface and its edges, we prefilter the current surface $\mathcal{M}(t)$ by straightforward "geometric Gaussian" filtering. Hence, we compute a shape operator $S^\sigma_{\mathcal{T}_x\mathcal{M}}$ on the resulting prefiltered surface $\mathcal{M}_\sigma(t)$, where σ is the corresponding filter width. Finally one obtains the following type of evolution problem:

$$\partial_t x - \mathrm{div}_{\mathcal{M}(t)}(a^\sigma_{\mathcal{T}_x\mathcal{M}}\, \nabla_{\mathcal{M}(t)} x) = f.$$

Thereby, we define the diffusion tensor $a^\sigma_{\mathcal{T}_x\mathcal{M}} = a(S^\sigma_{\mathcal{T}_x\mathcal{M}})$ with respect to the orthonormal basis of principal curvature directions on $\mathcal{M}_\sigma$ by

$$a^\sigma_{\mathcal{T}_x\mathcal{M}} = \begin{pmatrix} G(\kappa^{1,\sigma}) & 0 \\ 0 & G(\kappa^{2,\sigma}) \end{pmatrix},$$

where G is the already introduced edge indicator function. Thus, diffusion on the surface is significantly reduced in directions of high principle curvature, i.e. those perpendicular to an edge. On the other hand, a larger diffusion coefficient in the edge direction enables the tangential smoothing along the edge. The right hand side f of the considered evolution problem can be chosen such that the volume enclosed by $\mathcal{M}$ is preserved (cf. [**8**]) or one can select a simple retrieving force which avoids large deformations.

The evaluation of the shape operator will be based on an interpretation of the triangular grids of the discrete manifold as a graph over the tangent plane. A succeeding L^2 projection of the graph onto the quadratic polynomials enables the actual approximate evaluation of the shape operator, whose eigenvectors and eigenvalues we pick up for the definition of the diffusion tensor.

If we compare the new model with the anisotropic diffusion model in image processing we see a strong analogy. The difference only consists in the interpretation of the operators compared to the Euclidean case. Basically we have replaced

Euclidean differential operators by their geometric counterparts. The anisotropy in the new geometric model depends on regularized curvatures, which are based on second derivatives, whereas the Euclidean model considers a gradient based model. For a detailed description we refer the reader to [**8**].

As an example, in Figure 3 we see the evolution of a noisy laser scan surface under the anisotropic geometric diffusion method.

Figure 3. From left to right we have depicted four different scale steps of the anisotropic geometric evolution of the Venus head. The surface representation was obtained by a laser scan with additionally added noise.

3.3. Vector field visualization on surfaces

In Section 3.1 we have reviewed an anisotropic diffusion model for visualization of vector fields on domains that are subsets of the two- and three-dimensional Euclidean space. In what follows we will briefly outline how to carry over this concept to visualize vector fields on surfaces. The applications we show here will focus on vector fields in differential geometry, i.e. principal directions of manifolds, but the method is also applicable to results from meteorological computations or flow fields on stream surfaces. In analogy to the diffusion equation in the Euclidean case, we now ask for a solution $\rho\colon \mathbb{R}_0^+ \times \mathcal{M} \to \mathbb{R}^d$ of the parabolic equation

$$\partial_t \rho - \mathrm{div}_{\mathcal{M}}(A \nabla_{\mathcal{M}} \rho) = f(\rho)$$

on $\mathbb{R}_0^+ \times \mathcal{M}$ for given initial data $\rho(0, \cdot) = \rho_0$ on $\mathcal{M}$. Here we suppose A to be some positive definite symmetric endomorphism on $\mathcal{TM}$. To represent a vector field $v \in \mathcal{TM}$ we let w, for a non vanishing v, be a vector orthogonal to v in the sense of the metric, i.e. $g(v, w) = 0$, and then define with respect to the basis $\{v, w\}$ of $\mathcal{T}_x\mathcal{M}$ the diffusion tensor A as before in the Euclidean case

$$A(v, \nabla \rho_\sigma) = \begin{pmatrix} \alpha(\|v\|) & 0 \\ 0 & G(\|\nabla \rho_\sigma\|) \end{pmatrix}.$$

For the right hand side $f(\cdot)$, we take the function defined by (3.1) in Section 3.1 and again assume ρ_0 to be a random noise, either scalar or vector valued, but now on the surface $\mathcal{M}$. As an example in Figure 4 we have depicted the principal directions of curvature of Costa's surface.

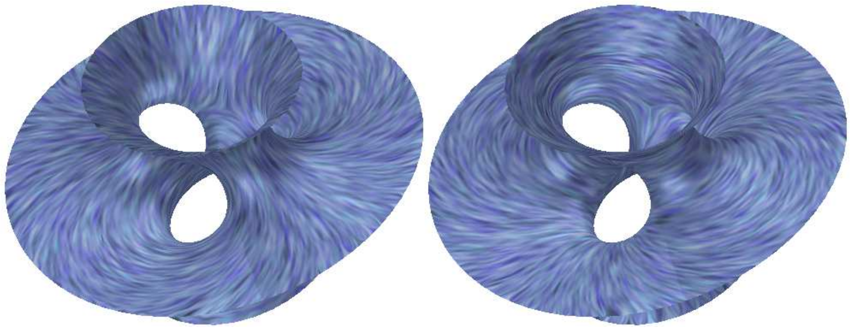

Figure 4. We depict the visualization of the principal directions of curvature of Costa's surface using the anisotropic diffusion method on surfaces.

3.4. A level set method for geometric image smoothing

The parametric surface processing model described in Section 3.2 is a well suited approach to edge preserving surface smoothing. One disadvantage, however, is the dependence upon a parameterization of the surface that makes it enormously difficult to apply the method to e.g. the isosurfaces of medical data, where parametrizations of the isosurfaces are not known. A modification of MCM was proposed by Sapiro [**28**] who considered so called self-snakes, which are steered by a coefficient depends on the image gradient. Here we present a level set approach which in addition comes along with the feature that ellipsoidal level sets remain invariant under the evolution [**26**].

As already described in Section 3.2 an edge feature is characterized by a small curvature in tangential direction along the feature and a sufficiently large curvature in the perpendicular direction in the tangent space. Let us assume $\kappa^1 \leq \kappa^2$ to be the principal curvatures and denote by v^1 and v^2 the corresponding principal directions of curvature. Hence we consider an anisotropic diffusion tensor depending on the shape operator extended to $\mathbb{R}^3$: $S := DN$. Thereby the diffusion tensor is supposed to significantly decrease the diffusion in the dominant curvature direction v^2, whereas a fixed diffusion is prescribed in the subdominant v^1 direction. This distinction will again be made via a function G applied to the principle curvatures κ^1, κ^2.

Again the evaluation of the shape operator S on a level set of a noisy image might be misleading with respect to the true but unknown level sets and edges. E.g. noise might be identified as features. Thus we have to consider a regularization and prefilter the current image $\rho(t,\cdot)$, which leads to a regularized shape operator S^σ. Either we consider an appropriate "morphological" filter which is a short timestep of the level set evolution by mean curvature, or alternatively we base it on a least square approximation of the true local level set in a suitable finite dimensional space of smooth functions and compute the shape operator on the corresponding level set. A third choice would be the convolution of the image $\rho(t,\cdot)$ with a Gaussian kernel (cf. [**7**]).

We end up with the following type of nonlinear parabolic problem. Given an initial 3D image ρ_0 on in Ω, we ask for a scale of images $\{\rho(t,\cdot)\}_{t\geq 0}$ which obey the anisotropic geometric evolution equation:

$$\begin{aligned} \partial_t \rho - \|\nabla\rho\| \operatorname{div}\left(\mathcal{A}(S^\sigma)\frac{\nabla\rho}{\|\nabla\rho\|}\right) &= 0 \qquad && \text{on } \mathbb{R}^+ \times \Omega, \\ \rho(0,\cdot) &= \rho_0(\cdot) && \text{on } \Omega, \\ \mathcal{A}(S^\sigma)\nabla\rho\cdot\nu &= 0 && \text{on } \mathbb{R}^+ \times \partial\Omega, \end{aligned}$$

where ν is the outer normal to $\partial\Omega$. $\mathcal{A}(S^\sigma)$ is supposed to be a symmetric, positive definite, linear endomorphism on $\mathbb{R}^3$, which cares about the preservation of edges and the tangential smoothing along edges. Thus we define

$$\mathcal{A}(S^\sigma) = B_\sigma^T \begin{pmatrix} G(\kappa^{1,\sigma}) & & \\ & G(\kappa^{2,\sigma}) & \\ & & 0 \end{pmatrix} B_\sigma \,.$$

Here $B_\sigma \in SO(3)$ denotes the basis transformation from the regularized frame of principal directions of curvature and the normal $\{v^{1,\sigma}, v^{2,\sigma}, N^\sigma\}$ onto the canonical basis $\{e_1, e_2, e_3\}$.

The underlying evolution turns out to be equivalent to the propagation of the level sets $\mathcal{M}^c(t)$ with speed f in normal direction N, i.e. $\partial_t x = f\, N$ with

$$f := \operatorname{tr}(\mathcal{A}(S^\sigma)(S_\sigma - S)) + (\operatorname{div}\mathcal{A}(S^\sigma))(N^\sigma - N)\,,$$

because (using the abbreviation $a^\sigma := \mathcal{A}(S^\sigma)$) we have

$$\begin{aligned} \operatorname{div}(a^\sigma N) &= (\operatorname{div} a^\sigma)N + \operatorname{tr}(a^\sigma DN) \\ &= (\operatorname{div} a^\sigma)N^\sigma + (\operatorname{div} a^\sigma)(N - N^\sigma) + \operatorname{tr}(a^\sigma S) \\ &= \operatorname{div}(a^\sigma N^\sigma) - \operatorname{tr}(a^\sigma DN^\sigma) + (\operatorname{div} a^\sigma)(N - N^\sigma) + \operatorname{tr}(a^\sigma S) \\ &= 0 - \operatorname{tr}(a^\sigma S_\sigma) + (\operatorname{div} a^\sigma)(N - N^\sigma) + \operatorname{tr}(a^\sigma S)\,. \end{aligned}$$

Here we define $S_\sigma := DN^\sigma$, where N^σ is the normal of the regularized image. Clearly S_σ coincides with the regularized shape operator S^σ if we evaluate S^σ on the level sets of a globally prefiltered image. But evaluating S^σ separately for each point $x \in \Omega$ on a locally regularized level set surface leads in general to $S^\sigma \neq S_\sigma$.

For noisy images we expect $\operatorname{tr}(\mathcal{A}(S^\sigma)(S_\sigma - S))$ to be the dominant term in the propagation speed, since in general for $x \in \Omega$

$$\|S_\sigma(x) - S(x)\| \gg \|N(x) - N^\sigma(x)\|.$$

This enables us to characterize the behavior of the anisotropic level set method. It is mainly driven by the difference of a regularized shape operator and the true shape operator weighted by the anisotropic weights given by the diffusion tensor. Furthermore, we verify that the propagation speed f vanishes if the regularization S_σ coincides with the original S, which gives reason for the invariance property stated above.

In our model above we have made extensive use of a regularized shape operator, on which we base the computation of the anisotropic diffusion tensor. Whatever process we apply to locally or globally regularize the image intensities, we have to define a discrete shape operator on level sets described by finite element functions. As long as we do not use at least quadratic elements even a definition – without thinking of the consistency problem – remains an open question [**29**]. Since typical image discretizations are based on trilinear interpolation of pixel or voxel values, we consider the second regularization variant, which is based on a local L^2 projection. Unfortunately this regularization defined in the sequel is not guaranteed to be invariant under gray value transformations. Nevertheless we expect the corresponding regularized shape operator to depend essentially only on the morphology of the local image.

We base the local regularization on a least squares fit of the image ρ onto a subspace of the polynomials $\mathcal{P}_2$. To this end let us fix a point $x \in \Omega$ and denote by $\mathcal{Q}$ the subspace of $\mathcal{P}_2$ that does not contain constant functions. It is not necessary to consider constant functions in $\mathcal{Q}$ since we can locally shift the image ρ such that $\rho(x) = 0$. The local L^2 projection $\Pi_{x,\sigma}\rho \in \mathcal{Q}$ of the intensity ρ onto $\mathcal{Q}$ is then defined by the orthogonality

$$\int\limits_{\mathcal{B}_\sigma(x)} (\rho - \Pi_{x,\sigma}\rho)\, q = 0 \qquad\qquad \forall\, q \in \mathcal{Q},$$

where $\mathcal{B}_\sigma(x)$ is a small neighborhood of x. For the ease of presentation we write ρ^σ instead of $\Pi_{x,\sigma}\rho$ for a fixed $x \in \Omega$. Now we define the shape operator $S_\sigma := D\frac{\nabla\rho^\sigma}{\|\nabla\rho^\sigma\|}$, which is symmetric and therefore is characterized by its real eigenvalues 0 and $\kappa^{j,\sigma}$, $j = 1, 2$ and the eigenvectors $\{N^\sigma, v^{1,\sigma}, v^{2,\sigma}\}$. In general we have $S_\sigma \neq S^\sigma$. For a more detailed description of the model, the regularization and a discussion of the implementation we refer to [**26**].

In Figure 5 we see the evolution of a noisy echocardiographical data set and in Figure 6 we have compared the new anisotropic level set method with other well known techniques (cf. [**14, 21**]).

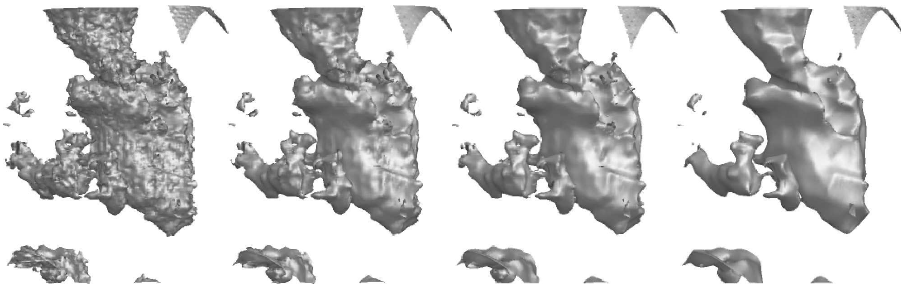

Figure 5. From left to right we depict a noisy echocardiographical data set of the left ventricle of the human heart and three scale steps of the new anisotropic level set method applied to that initial data.

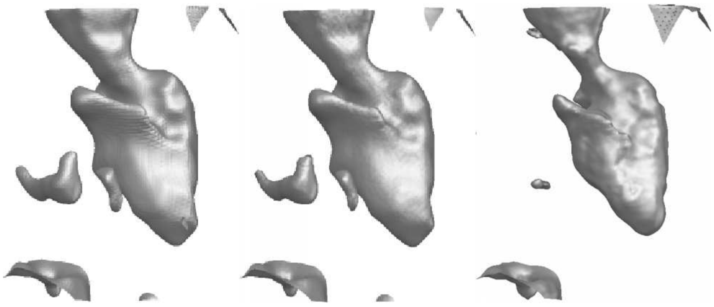

Figure 6. The results of the isotropic Perona Malik diffusion (left), anisotropic Perona Malik diffusion [**31**] (middle) and mean curvature motion (right) applied to the noisy data set from Figure 5 are shown. The new geometric diffusion clearly retains most of the edges and preserves the volume best.

4. Finite Element Discretization

In what follows, we discuss the discretization and implementation of the nonlinear diffusion methods. We will focus on domains in 2D and 3D Euclidean space and refer to [**8, 12**] for a detailed description of a discretization on manifolds.

Let us first look at the discretization of the anisotropic diffusion. The variational formulation of the diffusion problem 2.1 is obviously given by

$$(\partial_t \rho, \theta) + (A(v, \nabla \rho_\sigma)\nabla \rho, \nabla \theta) = (f, \theta),$$

for all $\theta \in C^\infty(\Omega)$, where $(\cdot,\cdot)$ denotes the L^2 product on the domain Ω. We consider a finite element discretization and a semi implicit backward Euler scheme in time. Here we have restricted ourselves to regular grids in 2D and 3D generated by recursive subdivision. On these grids we consider bilinear, respectively trilinear finite element spaces. Numerical integration is based on the lumped masses product $(\cdot,\cdot)^h$ [**30**] which approximates the L^2 product $(\cdot,\cdot)$ in the variational formulation and a one point quadrature rule for the bilinear form $(A\nabla\,\cdot, \nabla\,\cdot)$. The semi-implicit character of the scheme results in the evaluation of the nonlinearity $A(\cdot)$ and the right hand side at the old time. So finally, we have to solve a system of linear equations in each timestep of the discrete evolution. For the backward Euler discretization, we obtain

$$(M^n + \tau L^n(A^n))\bar{\rho}^{n+1} = M^n \bar{\rho}^n + \tau M^n \bar{f}^n\,.$$

Here $\bar{\rho}^n = (\bar{\rho}^n_i)_i$ denotes the vector of nodal intensity values at time $t^n = n\tau$, where τ is the selected timestep size. Furthermore, if we consider the "hat shaped" multilinear basis functions Φ_i and assume A^n to represent the diffusion tensor with respect to the discrete intensity at time t^n,

$$M^n := \left((\Phi_i, \Phi_j)^h\right)_{ij} \quad \text{and} \quad L^n(A^n) := ((A^n \nabla\Phi_i, \nabla\Phi_j))_{ij}$$

are the lumped mass matrix and stiffness matrix respectively. Finally, the components of the right hand side $\bar{f}^n$ are evaluated by $(\bar{f}^n)_i = f(\bar{\rho}^n_i)$.

In each timestep the computation of the prefiltered intensity vector $\bar{\rho}^n_\sigma$ is based on a single implicit timestep of size $\sigma^2/2$ for the corresponding discrete heat equation scheme with respect to initial data $\bar{\rho}^n$.

To discretize the anisotropic level set method, we regularize the variational formulation, since in general we cannot guarantee that $\nabla\rho \neq 0$ (cf. [**13**]). Thereby, we work with the variational formulation

$$\left(\frac{\partial_t \rho}{\sqrt{\|\nabla\rho\|^2+\epsilon^2}}, \theta\right) + \left(\mathcal{A}(S^\sigma)\,\frac{\nabla\rho}{\sqrt{\|\nabla\rho\|^2+\epsilon^2}}, \nabla\theta\right) = 0 \tag{4.1}$$

for $\epsilon > 0$ and for all test functions $\theta \in C^\infty(\Omega)$ and proceed as described above.

In our implementation the regular grids are procedurally interpreted as quadtrees, respectively octtrees. Finally no matrix is explicitly stored. The necessary matrix multiplications in the applied iterative solver are performed in successive tree traverses. Hierarchical BPX type [**5**] preconditioning is used to accelerate the convergence of the linear solver. Furthermore the code is prepared to incorporate spatial grid adaptivity if possible [**25**].

5. Hardware Accelerated Solvers

Numerical computations in graphics hardware are only slightly different from those of a computer. Graphics cards also consist of a central processor, the graphics processor unit (GPU), and memory, typically called texture memory. (The structure of some graphics cards does not exactly fit this simple scheme). The performance of nowadays standard CPUs and GPUs are comparable, whereas the texture memory offers a significantly higher bandwidth than the main memory of a computer. The main difference in computing is that the commands to be executed are not listed in the texture memory, but still in the main memory of the PC, only the operands are in the texture memory. For example to perform an addition, the corresponding command is sent to the GPU, which then gets the operands from the texture memory, processes them and writes the result back to the texture memory. We should think of the operands as large data blocks, e.g. entire images, because only then the advantages of the graphics cards can be fully exploited. For the execution of the pixelwise addition of two entire images the GPU namely needs only very few commands, the access to the operands is very fast, and the operation is performed in parallel on several components.

While using graphics hardware for computations there are two important issues which we must pay attention to. First, the number formats supported by the GPU offer only the range $[0, 1]$, which is suitable for the representation of intensities. As usually our computation will require a bigger range, we have to encode are numbers appropriately and perform any operations compliantly to the encoding. Second, the resolution of numbers in the GPU offers at most 12 bit per color component. Therefore the algorithm should be designed with the intention of reducing the necessary interval of numbers, because any extension of it leads through the encoding to a decrease in precision.

We will now consider the regularized Perona Malik model reviewed in Section 2. In Section 4 we have seen that after appropriate discretization in space and time we have to solve the following system of equations: $(M^n + \tau L^n(\rho^n_\sigma))\bar\rho^{n+1} = M^n\bar\rho^n + \tau M^n \bar f^n$. If we rescale the above equation to reference finite elements we obtain

$$\underbrace{\left(I + \frac{\tau}{h^2}\hat L(\rho^n_\sigma)\right)}_{A^n}\underbrace{\bar\rho^{n+1}}_{\bar\rho^{n+1}} = \underbrace{\bar\rho^n + \tau \bar f(\rho^n)}_{\bar R^n}.$$

Now, let us consider a simple iterative solver, like the Jacobi solver

$$F(\bar X) \quad = \quad D^{-1}(\bar R - (A-D)\bar X), \qquad D := \mathrm{diag}(A),$$

and think of the operations needed to implement it in graphics hardware. The GPU provides operations for the componentwise addition and multiplication as well as for the application of one argument functions (D^{-1}), but we cannot implement the matrix vector product straight forward. Instead we consider a splitting of operations and observe the following:

$$\begin{aligned}\left(\hat L(\rho^n_\sigma)\right)_{\alpha,\beta} &= \left(G(\|\nabla\rho^n_\sigma\|)\nabla\hat\Phi_\alpha, \nabla\hat\Phi_\beta\right)\\ &= \sum_{E\in E(\alpha)} G(\|\nabla\rho^n_\sigma\|)|_E\left(\nabla\hat\Phi_\alpha|_E, \nabla\hat\Phi_\beta|_E\right)\\ &= \sum_{E\in E(\alpha)} \bar G^n_E S^{\beta-\alpha},\end{aligned}$$

where $E(\alpha)$ is the set of elements around the node α, $\bar G^n_E := G(\|\nabla\rho^n_\sigma\|)|_E$ is the constant value of the diffusion coefficient on such an element and $S^{\beta-\alpha} := \left(\nabla\hat\Phi_\alpha|_E, \nabla\hat\Phi_\beta|_E\right)$ is a constant depending only on the index offset $\beta-\alpha$, provided we deal with equidistant meshes. Then applying $\hat L(\rho^n_\sigma)$ to an arbitrary vector $\bar X$ results in:

$$\begin{aligned}\left(\hat L(\rho^n_\sigma)\bar X\right)_\alpha &= \sum_\beta \sum_{E\in E(\alpha)} \bar G^n_E S^{\beta-\alpha}\bar X_\beta\\ &= \sum_{E\in E(\alpha)} \bar G^n_E \sum_\gamma S^\gamma \bar X_{\alpha+\gamma}.\end{aligned}$$

Here it is convenient to look upon α and β as 2 or 3 dimensional multi-indices, enumerating the nodes of the 2 or 3 dimensional grid respectively. Then $\gamma := \beta-\alpha$ is the spatial offset from node α to node β. Hence the inner sum represents a weighted sum of neighboring node values, where the weights S^γ are independent of the local position α. This discrete convolution with constant weights is directly supported by some GPUs. If not available, one can simulate it by adding shifted and weighted copies of $\bar X$. For the underlying quadtree or octtree and linear basis functions Φ_α, the index offset γ satisfies $|\gamma| \le d$, if $|\cdot|$ indicates the 1-norm on $\mathbb{R}^d$, in other words the stencil of the convolution filter is a 3 by 3 (by 3) matrix. Finally the multiplication with $\bar G^n_E$ is simply a componentwise multiplication with the vector $\bar G^n := (G^n_E)_E$.

Consequently, we list the operations required for the implementation of the Jacobi solver and their counterparts in graphics hardware using the OpenGL API [**22**]:

operation	**OpenGL**
componentwise linear combination	blending function
componentwise multiplication	blending function
componentwise function of one argument	lookup table
convolution	convolution extension

Table 2. OpenGL functionality corresponding to vector operations.

Figure 7 shows computations on the InfiniteReality2 graphics system of a SGI Onyx2 applied to the same noisy image as in Figure 1. Unlike many others, this graphics system offers display modes with 12 bit per color component, which enhances the accuracy of the calculations. We see that this precision is sufficient for the task of denoising pictures by nonlinear diffusion models. Certainly the images produced by hardware and software vary, but the overall effect of the image seems very much the same for our visual perception, which is the decisive factor in this application.

Figure 7. Nonlinear diffusion in graphics hardware (cf. Figure 1).

The performance of the InfiniteReality2 system of about 17sec for the computation of 10 timesteps à 10 Jacobi iterations which lead to the 256^2 images in Figure 7 is rather weak in this setting. This is because our algorithm strongly depends on a fast texture loading from the framebuffer, but in the time when the InfiniteReality2 graphics emerged, the graphics pipeline has only been optimized for drawing textured images into the framebuffer.

An implementation on the modern ELSA Gladiac Ultra PC graphics card powered by NVIDIA's GeForce2 Ultra chip runs through the 10 Jacobi iterations of a timestep in just 0.1sec. Here also the inverse path from the framebuffer to the texture memory has been optimized, eliminating the afore mentioned bottleneck. The lower resolution of only 8 bits per color component, however, poses additional difficulties in bounding the computational error and thus reconstructing the qualitative behavior of the analytical model.

Fortunately, the development of PC graphics hardware tends towards higher precision formats with up to 16 bits per component, such that in future even more precision sensitive algorithms will permit graphics hardware implementations. Moreover, performance is still very likely to rise by factors, because currently certain restrictions and unoptimized paths in the graphics pipeline enforce time consuming computational detours.

In this section we have outlined the implementation of the regularized Perona Malik model in graphics hardware. The availability of the basic vector operations and the reformulation of the sparse matrix vector product in graphics hardware, however, also allow the implementation of more complex models like these from sections 3.1 and 3.4. Generally speaking, the array of applications feasibly implementable in graphics hardware is restricted by the precision requirements and the homogeneity of the processed data. An example for the graphics hardware implementation of a different model than the nonlinear diffusion has been discussed in [**27**]. There, we have implemented an explicit solver for the levelset equation, and use it for the segmentation of images.

6. Conclusions

We have seen a wide range of applications for the nonlinear diffusion models, ranging from denoising techniques to vector field visualization. In either case we could experience the advantage of a multiscale of solutions which allows to choose the appropriate coarsening scale for the application. Concerning the implementation we have proposed a software and a hardware solution. The software solution uses hierarchical adaptivity and fast solvers for optimal performance with minimum memory requirements, whereas the hardware solution operates on equidistant meshes, but benefits from the higher memory bandwidth of the graphics hardware and its intrinsic parallelization. Both approaches have their pros and cons. The software solution guaranties high precision, very good scalability and implementational flexibility, but lacks ultimate performance. The graphics hardware, on the other hand, frees the CPU for other computations and still offers supreme performance, but lacks high precision and flexibility. Depending on the application, choosing the appropriate approach or a combination of both will lead to satisfactory results.

Acknowledgments. We would like to thank Stefan Turek for supplying the data depicted in Figure 2 and Matthias Hopf for very helpful information on the graphics hardware programming. We furthermore acknowledge TomTec Imaging Systems and C. Lamberti from DEIS, University of Bologna, who provided the echocardiographical data shown in figures 6 and 5.

References

1. Acton S. T., *Multigrid anisotropic diffusion*, IEEE Transactions on Image Processing **7** (1998), 280–291.

2. Alvarez L., Guichard F., Lions P. L. and Morel J. M., *Axioms and fundamental equations of image processing*, Arch. Ration. Mech. Anal. **123**(3) (1993), 199–257.

3. Alvarez L., Lions P. L. and Morel J. M., *Image selective smoothing and edge detection by nonlinear diffusion II*, SIAM J. Numer. Anal. **29** (1992), 845–866.
4. Bänsch E. and Mikula K., *A coarsening finite element strategy in image selective smoothing*, Computing and Visualization in Science **1** (1997), 53–63.
5. Bramble J. H., Pasciak J. E. and Xu J., *Parallel multilevel preconditioners*, Math. Comp. **55** (1990), 1–22.
6. Cabral B. and Leedom L., *Imaging vector fields using line integral convolution*, In Computer Graphics SIGGRAPH '93 Proceedings (J. T. Kajiya, ed.), Volume 27, pp. 263–272, Aug. 1993.
7. Catté F., Lions P. L., Morel J. M. and Coll T., *Image selective smoothing and edge detection by nonlinear diffusion*, SIAM J. Numer. Anal. **29**(1) (1982), 182–193.
8. Clarenz U., Diewald U. and Rumpf M., *Nonlinear anisotropic diffusion in surface processing*, In Proc. Visualization 2000, pages 397–405, 2000.
9. Cullip T. and Neumann U., *Accelerating volume reconstruction with 3d texture hardware*, Technical Report TR93-027, University of North Carolina, Chapel Hill N.C., 1993.
10. Deckelnick K. and Dziuk G., *A fully discrete numerical scheme for weighted mean curvature flow*, Technical Report 30, Mathematische Fakultät Freiburg, 2000.
11. Desbrun M., Meyer M., Schröder P. and Barr A., *Implicit fairing of irregular meshes using diffusion and curvature flow*, In Proc. SIGGRAPH '99, Los Angeles, CA, Aug. 1999.
12. Diewald U., Preußer T. and Rumpf M., *Anisotropic diffusion in vector field visualization on euclidean domains and surfaces*, Trans. Vis. and Comp. Graphics **6**(2) (2000), 139–149.
13. Evans L. and Spruck J., *Motion of level sets by mean curvature I*, J. Diff. Geom. **33**(3) (1991), 635–681.
14. Handlovičová A., Mikula K. and Sarti A., *Numerical solution of parabolic equations related to level set formulation of mean curvature flow*, Computing and Visualization in Science **1**(3) (1998), 179–182.
15. Hopf M. and Ertl T., *Accelerating 3d convolution using graphics hardware*, In Proc. Visualization '99, IEEE, pages 471–474, 1999.
16. ______, *Hardware Accelerated Wavelet Transformations*, In Proceedings of EG/IEEE TCVG Symposium on Visualization VisSym '00, pages 93–103, 2000.
17. Kačur J. and Mikula K., *Solution of nonlinear diffusion appearing in image smoothing and edge detection*, Appl. Numer. Math. **17**(1) (1995), 47–59.
18. Kawohl B. and Kutev N., *Maximum and comparison principle for one-dimensional anisotropic diffusion*, Math. Ann. **311**(1) (1998), 107–123.
19. Kichenassamy S., *The perona-malik paradox*, SIAM J. Appl. Math. **57** (1997), 1343–1372.
20. Kimmel R., *Intrinsic scale space for images on surfaces: The geodesic curvature flow*, Graphical Models and Image Processing **59**(5) (1997), 365–372.
21. Mikula K., Sarti A. and Lamberti C., *Geometrical diffusion in 3D-echocardiography*, In Proceedings of contributed papers and posters, ALGORITMY '97 Conference on Scientific Computing, West Tatra Mountains, pages 167–181, 1997.
22. *OpenGL Architectural Review Board (ARB)*, `http://www.opengl.org/`, OpenGL: graphics application programming interface (API), 1992.
23. Perona P. and Malik J., *Scale space and edge detection using anisotropic diffusion*, IEEE Trans. Pattern Anal. Mach. Intell. **12** (1990), 629–639.
24. Preußer T. and Rumpf M., *Anisotropic nonlinear diffusion in flow visualization*, In Proceedings Visualization 1999, pages 325–332, 1999.
25. ______, *An adaptive finite element method for large scale image processing*, Journal of Visual Comm. and Image Repres **11** (2000), 183–195.
26. ______, *A level set method for anisotropic diffusion in 3D image processing*, SIAM J. Appl. Math., submitted, 2000.
27. Rumpf M. and Strzodka R., *Level set segmentation in graphics hardware*, In Proceedings ICIP'01, 2001.

28. Sapiro G., *Vector (self) snakes: A geometric framework for color, texture, and multiscale image segmentation*, In Proc. IEEE International Conference on Image Processing, Lausanne, September 1996.

29. Schmidt A., *Computation of three dimensional dendrites with finite elements*, Journal of Computational Physics **125** (1996), 293–312.

30. Thomee V., *Galerkin – Finite Element Methods for Parabolic Problems*, Springer, 1984.

31. Weickert J., *Anisotropic diffusion in image processing*, Teubner, 1998.

32. Weickert J., Zuiderveld K., ter Haar Romeny B. and Niessen W., *Parallel implementations of AOS schemes: A fast way of nonlinear diffusion filtering*, In Proc. Fourth IEEE International Conference on Image Processing, Volume **3**, pages 396–399, Santa Barbara, CA, Oct 1997.

33. Westermann R. and Ertl T., *Efficiently using graphics hardware in volume rendering applications*, Computer Graphics (SIGGRAPH '98) **32**(4) (1998), 169–179.

34. Wilson O., van Gelder A. and Wilhelms J., *Direct volume rendering via 3d textures*, Technical Report UCSC CRL 94-19, University of California, Santa Cruz, 1994.

35. You Y.-L., Xu W., Tannenbaum A. and Kaveh M., *Behavoiral analysis of anisotropic diffusion in image processing*, IEEE Trans. Image Proc. **5** (1996), 1539–1553.

U. Diewald, Institute for Applied Mathematics, University of Bonn, Weglerstraße 6, 53115 Bonn, Germany, *e-mail*: `diewald@iam.uni-bonn.de`

T. Preusser, Applied Mathematics, University of Duisburg, Lotharstraße 65, 47048 Duisburg, Germany, *e-mail*: `tp@math.uni-duisburg.de`

M. Rumpf, Applied Mathematics, University of Duisburg, Lotharstraße 65, 47048 Duisburg, Germany, *e-mail*: `rumpf@math.uni-duisburg.de`

R. Strzodka, Applied Mathematics, University of Duisburg, Lotharstraße 65, 47048 Duisburg, Germany, *e-mail*: `strzodka@math.uni-duisburg.de`