LEARNING AN INTEGRAL EQUATION APPROXIMATION TO NONLINEAR ANISOTROPIC DIFFUSION IN IMAGE PROCESSING

Bruce Fischl and Eric L. Schwartz

December 1995

Technical Report CAS/CNS-95-033

Boston University Center for Adaptive Systems and
Department of Cognitive and Neural Systems
111 Cummington Street
Boston, MA 02215

Learning an Integral Equation Approximation to Nonlinear Anisotropic Diffusion in Image Processing. *

Bruce Fischl †

†Dept. Cognitive and
Neural Systems
Boston University
Boston, MA 02146
e-mail: fischl@cns.bu.edu

Eric L. Schwartz ‡

‡Dept. Cognitive and
Neural Systems
Boston University
Boston, MA 02146
e-mail: eric@thing4.bu.edu

December 22, 1995

*supported in part by the office of naval research (ONR N00014-95-1-0409).

Abstract

Multi-scale image enhancement and representation is an important part of biological and machine early vision systems. The process of constructing this representation must be both rapid and insensitive to noise, while retaining image structure at all scales. This is a complex task as small scale structure is difficult to distinguish from noise, while larger scale structure requires more computational effort. In both cases good localization can be problematic. Errors can also arise when conflicting results at different scales require cross-scale arbitration.

Broadly speaking, multi-scale image analysis has historically been accomplished using two types of techniques: those which are sensitive to image structure and those which are not. Algorithms in the latter category typically use a set of variously sized blurring kernels to produce images each of which retain structure at a different scale (Marr and Hildreth, 1980; Burt and Adelson, 1983; Koenderink, 1984; Hummel, 1986). The kernels used for the blurring are predefined and independent of the content of the image. Koenderink showed that if the kernels are Gaussian, then this process is equivalent to the evolution of the linear heat (or diffusion) equation. He thus transformed the integral equation representing the convolution process into the solution of a partial differential equation (PDE).

Structure sensitive multi-scale techniques attempt to analyze an image at a variety of scales within a single image (Klinger, 1971; Perona and Malik, 1987; Nitzberg and Shiota, 1992). Klinger (Klinger, 1971) proposed the quad tree, one of the earliest structure-sensitive multi-scale image representations. In this approach, a tree structure is built by recursively subdividing an image based on pixel variance in subregions. The final tree contains leaves representing image regions whose variance is small according to some measure. Recently (Perona and Malik, 1987; Perona and Malik, 1990), the PDE formalism introduced by Koenderink has been extended to allow structure-sensitive multi-scale analysis. Instead of the uniform blurring of the linear heat equation which destroys small scale structure as time evolves, Perona and Malik use a space-variant conductance coefficient based on the magnitude of the intensity gradient in the image, giving rise to a nonlinear PDE. Like the quadtree, the end result is a single image representation which contains information at all scales of interest.

The Perona and Malik approach produces impressive results, but the numerical integration of a nonlinear PDE is a costly and inherently serial process. In this paper we present a technique which obtains an approximate solution to the PDE for a specific time, via the solution of an integral equation which is the nonlinear

analog of convolution. The kernel function of the integral equation plays the same role that a Green's function does for a linear PDE, allowing the direct solution of the nonlinear PDE for a specific time without requiring integration through intermediate times. We then use a learning technique to approximate the kernel function for arbitrary input images. The result is an improvement in speed and noise-sensitivity, as well as providing a means to parallelize an otherwise serial algorithm.

1 Introduction.

Multi-scale image enhancement and representation is an important part of biological and machine early vision systems. The process of constructing this representation must be both rapid and insensitive to noise, while retaining image structure at all scales. Attempts to solve problems of this type resulted in the so-called 'scale-space' formulation of Witken (Witken, 1983) in which an image is convolved with Gaussian kernels of various sizes. Edges delineating the boundaries between objects can then be found in a number of ways, for example by tracing the zero-crossings of the Laplacian through scale-space in much the same way that Marr and Hildreth had proposed (Marr and Hildreth, 1980). This approach is problematic as the zeros can change position and disappear as scale-space is traversed. In this situation it is unclear how to arbitrate between conflicting results at different scales.

Koenderink (Koenderink, 1984) and Hummel (Hummel, 1986) have pointed out that the one-parameter family of images comprising scale-space can equivalently be viewed as snapshots of the time-evolution of the diffusion (or heat) equation:

$$I_t = c\Delta I \tag{1}$$

Where I is the intensity image, c is a diffusion constant, I_t is the partial derivative of I with respect to time, and Δ is the Laplacian operator with respect to the spatial coordinates. The solution to equation (1) can be written in terms of the Green's [1] function of the system as

[1] More accurately, the Green's function is the Gaussian multiplied by a temporal step

$$I(x,y,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x',y',0)G(x,x',y,y',t)dx'dy' \tag{2}$$

Where $I(x',y',0)$ is the initial image, and the Green's function $G(x,x',y,y',t)$ is a Gaussian kernel given by

$$G(x,x',y,y',t) = \frac{1}{\sqrt{4\pi ct}} e^{-\frac{(x-x')^2+(y-y')^2}{4ct}} \tag{3}$$

The Green's function $G(x,x',y,y',t)$ is the kernel of the integral operator which is the inverse of the diffusion operator. Thus, convolution with larger scale Gaussian kernels is equivalent to the evolution of the diffusion equation on an infinite domain for longer periods of time, with the original image as initial conditions.

The diffusion equation provides a mathematical framework with which to analyze the scale-space formalism, but it does not address the issue of cross-scale comparison. While Koenderink restricted his analysis to the isotropic diffusion characterized by the linear heat equation, Perona and Malik (Perona and Malik, 1987; Perona and Malik, 1990) suggested that a nonlinear anisotropic version of the heat equation could remedy some of the difficulties encountered in the use of linear scale-space. They proposed the following equation in which the conduction coefficient is not constant in space, but is rather a function of the magnitude of the intensity gradient of the image:

$$I_t = \nabla \cdot (c(\mid \nabla I \mid)\nabla I) \tag{4}$$

In this way, the amount of diffusion at each point in space is modulated by the function $c(|\nabla I|)$, and the image gradient at that point. They choose to make $c(\cdot)$ a decreasing function of the image gradient magnitude, so that regions of high contrast undergo less diffusion, and are therefore preserved over time. This is in contrast to the linear heat equation which blurs uniformly, destroying small scale structure as time evolves. Systems such as equation (4) are intended to yield a single intensity image which

function (Barton, 1991)

retains edge information at all scales of interest, thus obviating the need for any type of cross-scale arbitration.

The Perona-Malik equation (4) is a nonlinear partial differential equation of a type which is difficult to analyze. It has been pointed out (Nitzberg and Shiota, 1992) that (4) is unstable for some parameter regimes, although this is still a point of contention (Perona et al., 1994). Furthermore, it can amplify small scale noise which gives rise to high gradient magnitudes. Many variants of the Perona and Malik scheme have been proposed to improve its sensitivity to noise, its instability, and its equilibrium behavior (Alvarez et al., 1992; Alvarez and Mazorra, 1994; Catte et al., 1992; Cottet and Germain, 1993; Dang et al., 1994; El-Fallah and Ford, 1994; Engquist et al., 1989; Illner and Neunzert, 1993; Li and Chen, 1994; Nitzberg and Shiota, 1992; Nordstrom, 1990; Osher and Rudin, 1990; Pauwels et al., 1993; Price et al., 1990; Whitaker and Pizer, 1991; Whitaker, 1993; Kacur and Mikula, 1995).

The diffusion paradigm, while impressive in the quality of the images it produces, suffers from a number of drawbacks. The most prominent of these is the computational cost of the algorithms, coupled with their inherently serial nature. This makes them implausible from a biological standpoint, as well as impractical for use in real-time machine vision applications. The biological implausibility stems from the relatively rapid nature of perception relative to neural conduction delays and peak firing rates (<= 200Hz). Psychophysical and neurophysiological experiments indicate that perception can occur as rapidly as 100-150 msec (Thorpe and Imbert, 1989; Oram and Perrett, 1992) which is comparable to the latency of cells in primary visual cortex (Vogels and Orban, 1991). Using this figure, Thorpe and Imbert (Thorpe and Imbert, 1989) argue that the number of synaptic connections (assumed to be equivalent to the number of serial steps) used by the visual system in rapid identification tasks is somewhere between 10 and 50, although probably closer to the lower bound. Thus, while complex processing is possible, it is almost certainly parallel in nature.

In this paper we propose a method to directly learn the input-output mapping of the diffusion process. This has a number of advantages. Most importantly it parallelizes the inherently serial process of numerical integration. In addition, it obviates the need for regularization of (4) to preserve image structure at equilibrium, as an appropriate time constant

is implicitly imbedded in the system. Further, it is an order of magnitude faster than the full diffusion process, even when run on serial machines. The end result is an approximation of the kernel function in equation (2) for arbitrary input images, which we call a Green's Function Approximation (GFA) filter. While we present the algorithm in the context of anisotropic diffusion, the technique is applicable to a much wider range of image enhancements/modifications.

The remainder of this paper is organized as follows. Section 2 analyzes the task of approximating the input/output mapping of the nonlinear diffusion process and outlines the form the solution will take. Section 3 discusses the numerical implementation of a nonlinear anisotropic diffusion equation such as (4). Section 4 details the procedure for constructing the kernel function by monitoring the evolution of the diffusion process, while section 5 specifies the algorithm used to estimate the kernels for a novel image. In section 6 we present results of applying the algorithm to approximate two different diffusion processes, while section 7 is the conclusion.

2 Diffusion approximation.

The evolution of the diffusion process of equation (4) yields a three dimensional solution surface (two spatial and one temporal) when applied to an input image. From a machine vision standpoint, we are not concerned with the entire surface. Rather, we are interested in a cross-section of the surface (i.e. an image) within some temporal neighborhood which is in some sense desirable, such as one which produces sharp, noise-free edge maps. Given this restriction, it is reasonable to ask whether it is necessary to traverse the portion of the solution surface outside of our region of interest. That is, can we construct a mapping that takes an input image directly into some neighborhood of the target region? For the linear heat equation this is exactly the role that a Green's function plays. It is the kernel of the integral operator which yields an output image at any time without traversing the images at intervening times. Thus, we reformulate our question to be: can we find the kernel function $G(x, x', y, y')$ of an integral operator such that:

$$I(x,y,t) = \int\int_I I(x',y',0)G_t(x,x',y,y')dx'dy' \tag{5}$$

We subscript the kernel function $G(x,x',y,y')$ with t to emphasize the fact that different kernel functions exist for different evolution times. For computational reasons, the region of integration must be limited to a subset of the image centered around each (x,y). Equation (5) is an ill-posed problem, as an infinite number of kernel functions $G(\cdot)$ exist which satisfy it. We are not interested in arbitrary solutions of (5), but only those for which the output is continuously dependent on the input. In order to resolve these issues, we reconsider the heat equation.

The heat equation (4) can be derived using two assumptions. The first of these is the conservation principle that the time rate of change of the temperature in a given region is proportional to the divergence of the flux in that region. The second is the experimentally motivated Fourier's law of heat conduction (called Fick's law in a diffusion context) which states that the flux is proportional to the spatial temperature gradient. In the machine vision domain, the conservation law leads us to make the following observation. If we assume that image intensity $I(x,y)$ is a conserved quantity, it is possible to trace the diffusion path of each initial value in the image through space and time. This process, when carried out for each point in the image at time t, yields exactly the kernel function of the integral operator which we are seeking. The function $G_t(x,x',y,y')$ takes the form of a set of space-variant kernels, one for each point in the image. We defer further discussion of the details of the construction of $G_t(x,x',y,y')$ until section 4, as it is dependent on the form of the numerical implementation of equation (4).

Given that the kernel function has been constructed, we are capable of exactly mirroring the diffusion process for an image which has already been subject to diffusion (actually, if the support of the kernels is not the entire image, the equality in equation (5) is only approximate). By itself this is not particularly useful. In order to be a general purpose tool for use in early vision, we must be capable of approximating the kernel function for an arbitrary input image. This is a difficult problem, whose solution requires a number of steps. Even using kernels with limited support, we are still left with the task of approximating a high dimensional nonlinear mapping

from image intensity profile to kernel function. We are free to choose any image-based feature set as the domain of the mapping, while the range must allow us to reconstruct an approximate kernel function. In order to alleviate the curse of dimensionality (i.e. the exponential increase required in training set size with increasing dimension (Duda and Hart, 1973)) we employ principal components analysis (PCA) and limit ourselves to the components which account for a large ($\approx 90\%$) percentage of the variance. The range of the nonlinear mapping then becomes the coefficients of the PCA, whose dimensionality is typically between one and two orders of magnitude smaller than that of the full kernels.

3 Numerical Implementation of Anisotropic Diffusion.

The form the kernel function construction algorithm takes is dependent on the form of the numerical implementation of the anisotropic diffusion equation (4). For that reason, we first outline the simple implementation we use.

In two spatial dimensions, the anisotropic diffusion equation is given by:

$$I_t(x,y,t) = (c(x,y,t)(I_x(x,y,t)))_x + (c(x,y,t)(I_y(x,y,t)))_y \tag{6}$$

Where the x, y and t subscripts denote partial differentiation with respect to the subscripted variable. A Taylor series expansion of I around $t = t_0$ is given by:

$$I(x,y,t_0+\Delta t) = I(x,y,t_0) + \Delta t I_t(x,y,t_0) + ... \tag{7}$$

Using equations (6) and (7) we then have the first order approximation:

$$I(x,y,t_0+\Delta t) \approx I(x,y,t_0) + \Delta t((c(x,y,t)I_x(x,y,t))_x + (c(x,y,t)I_y(x,y,t))_y) \tag{8}$$

The derivatives in (8) can also be approximated using a Taylor series ex-

pansion:

$$f(x+\Delta x,y)\approx f(x,y)+\Delta x f_x(x,y)\Rightarrow f_x(x,y)\approx\frac{f(x+\Delta x,y)-f(x,y)}{\Delta x} \tag{9}$$

$$f(x-\Delta x,y)\approx f(x,y)-\Delta x f_x(x,y)\Rightarrow f_x(x,y)\approx\frac{f(x,y)-f(x-\Delta x,y)}{\Delta x} \tag{10}$$

Adding (9) and (10) yields the centered difference approximation of the derivative:

$$f_x(x,y)\approx\frac{f(x+\Delta x,y)-f(x-\Delta x,y)}{2\Delta x} \tag{11}$$

Using a discrete lattice with $\Delta x=\Delta y=1$, and considering the central pixel (x_0,y_0), and its four connected neighbors $(x_0,y_{-1}),(x_0,y_1),(x_{-1},y_0)$, and (x_1,y_0) we can apply equation (11) to approximate the derivatives in (8). Labelling these pixels with superscripts 0, N, S, W, E respectively, we have:

$$(c^0(t_0)I_x^0(t_0))_x\approx\frac{c^E(t_0)I_x^E(t_0)-c^W(t_0)I_x^W(t_0)}{2} \tag{12}$$

$$(c^0(t_0)I_y^0(t_0))_y\approx\frac{c^S(t_0)I_y^S(t_0)-c^N(t_0)I_y^N(t_0)}{2} \tag{13}$$

We use both backwards (equation (10)) and forwards (equation (9)) differences to approximate the partial derivatives with respect to the spatial variables so as to limit the domain of our numerical implementation to the four nearest neighbors of the central pixel:

$$I_x^W(t_0)=I^0(t_0)-I^W(t_0),I_x^E(t_0)=I^E(t_0)-I^0(t_0)$$

$$I_y^N(t_0) = I^0(t_0) - I^N(t_0), I_y^S(t_0) = I^S(t_0) - I^0(t_0) \tag{14}$$

Finally, substituting (12), (13) and (14) into (8) we arrive at:

$$I^0(t_0 + \Delta t) \approx I^0(t_0)(1 - 0.5\Delta t \sum_{i \neq 0} c^i(t_0)) + 0.5\Delta t \sum_{i \neq 0} c^i(t_0) I^i(t_0) \tag{15}$$

Equation (15) can equivalently be written as the correlation of the image with a set of space and time varying masks:

$$I(x, y, t_0 + \Delta t) \approx \sum_{x'} \sum_{y'} K_{x,y}^{t_0}(x', y') I(x + x', y + y', t_0) \tag{16}$$

Where the mask weights are given by:

$$K_{x,y}^{t_0} = \frac{\Delta t}{2} \begin{bmatrix} 0 & c^N(t_0) & 0 \\ c^W(t_0) & \frac{2}{\Delta t} - (\sum_{i \neq 0} c^i(t_0)) & c^E(t_0) \\ 0 & c^S(t_0) & 0 \end{bmatrix} \tag{17}$$

In 2 dimensions the two components of the spatial gradient used in the computation of the conductance function are calculated using a Sobel operator:

$$\begin{aligned} I_x(x, y, t) &= I(x, y, t) \star \begin{bmatrix} -0.25 & 0 & 0.25 \\ -0.50 & 0 & 0.50 \\ -0.25 & 0 & 0.25 \end{bmatrix} \\ I_y(x, y, t) &= I(x, y, t) \star \begin{bmatrix} -0.25 & -0.50 & -0.25 \\ 0 & 0 & 0 \\ 0.25 & 0.50 & 0.25 \end{bmatrix} \end{aligned} \tag{18}$$

Where $\star$ indicates correlation. Bounds on Δt can be computed using Fourier-von Neumann stability analysis. The numerical implementation will be stable if (Haberman, 1987):

$$\Delta t \leq \frac{(\Delta x)^2}{4c} \tag{19}$$

If we choose $c(\cdot)$ to be in the range (0,1] and let $\Delta x = 1$, we then have $\Delta t <= 0.25$. Note that we are not free to choose both Δt and Δx in this type of explicit scheme as is sometimes assumed (El-Fallah and Ford, 1994) without sacrificing stability.

In addition to stability, we require that the numerical scheme satisfy the maximum (minimum) principle. That is, the only local extrema allowable in the image are on the hyperboundary of the space-time domain defined by the evolving image. This is equivalent to what Koenderink (Koenderink, 1984) terms causality, as it implies that no new local spatiotemporal extrema can be created as time evolves. This is important as it insures that intensity values in the evolving image are constrained by the initial image values, and will therefore not grow without bound.

Defining $I^{max} = max(I^0(t_0), I^N(t_0), I^S(t_0), I^E(t_0), I^W(t_0))$, we have:

$$\begin{aligned} I^0(t_1) &= I^0(t_0)(1 - 0.5\Delta t \sum_{i \neq 0} c^i(t_0)) + 0.5\Delta t \sum_{i \neq 0} c^i(t_0) I^i(t_0) \\ &\leq I^{max}(t_0)((1 - 0.5\Delta t \sum_{i \neq 0} c^i(t_0)) + 0.5\Delta t \sum_{i \neq 0} c^i(t_0)) = I^{max}(t_0) \end{aligned} \tag{20}$$

Similarly, it is easy to show that $I^{min}(t_0) \leq I^0(t_1)$, and that thus both maximum and minimum principles hold.

The actual implementation of the diffusion equation on a discrete lattice is therefore accomplished by the iterative correlation of the initial data with a set of space-variant masks, the shapes of which are determined by the local values of the conductance function. To complete the specification, we use homogenous Neumann boundary conditions (i.e. zero flux across the border of the image). While this implementation is a simple one (for example, the use of only four points violates rotational invariance, biasing the equation towards $n\pi/2$ angles, $n = 0, 1, 2, 3$ (Niessen et al., 1994)), it is sufficient for our purposes as an example of the proposed scheme. In principle, any numerical implementation can be modified to include the kernel growth described below.

4 Kernel Function Construction.

In this section we define the algorithm used to construct the kernel function $G_t(x, x', y, y')$ of equation (5). The function will consist of an array of two dimensional kernels, which we term diffusion kernels, one for each point in the image. We denote the diffusion kernel at image point (x, y) by $C_{x,y}$. The value of this kernel at image location $(x + i, y + j)$ is $C_{x,y}(i, j)$. Similarly, we term the value of the mask associated with the image point (x, y) at location $(x + i, y + j)$ by $K_{x,y}(i, j)$, as in section 3. Since the four dimensional nature of these objects causes unnecessary notational clutter, we will proceed in one spatial dimension. The generalization to two dimensions is straightforward.

Before commencing with the details of the algorithm it is worth commenting on the different roles the masks and kernels play. The mask $K_x^t(i)$ relates the image value at time t at position $x + i$ to the image value at time $t + 1$ at position x.

$$I(x, t + 1) = \sum_i K_x^t(i) I(x + i, t) \tag{21}$$

Conversely, the kernel entry $C_x^t(j)$ prescribes the contribution of the initial image value at position $x + j$ to the image value at position x at time t (see figure (4)). That is:

$$I(x, t) = \sum_j C_x^t(j) I(x + j, 0) \tag{22}$$

Equation (22) is a discrete statement of the Green's function property of the diffusion kernels we are seeking, analogous to equation (2), while equation (21) reiterates the role of the masks in the numerical integration of the PDE.

In order to construct the diffusion kernels $C_x^t(i)$ which parallelize the diffusion, we proceed inductively. For each point x in the image, we create a kernel C_x and initialize it using a Kronecker delta function:

$$C_x^0(i) = \delta_i = \begin{cases} 1 & i = 0 \\ 0 & i \neq 0 \end{cases} \tag{23}$$

The application of this set of kernels to the initial image leaves it unchanged,

and therefore equation (22) hold for $t = 0$. We now seek a recursive update rule that will construct the diffusion kernel at time $t + 1$ assuming we have already built the kernels at time t. Combining equations (22) and (21) we find the relationship between the initial image and the image at time $t + 1$:

$$I(x, t + 1) = \sum_i K_x^t(i) \sum_j C_{x+i}^t(j) I(x + i + j, 0) \tag{24}$$

If we are seeking $C_x^{t+1}(n)$, the kernel element at some arbitrary position n, then we are only interested in the coefficients of (24) which multiply $I(x + n, 0)$. Examining (24) for values of j such that $i + j = n$, we arrive at the recursive update law:

$$C_x^{t+1}(n) = \sum_i K_x^t(i) C_{x+i}^t(n - i) \tag{25}$$

Equation (25) can be understood in the following way. The kernel value $C_{x+i}^t(n - i)$ represents the contribution of the initial intensity value at pixel $x + n$ to the pixel at $x + i$ at time t. This can be seen by noting that for $x' = x + i$, we have $C_{x+i}^t(n - i) = C_{x'}^t(n - i)$. $K_x^t(i)$ then gives the proportion of the total intensity at location x at time $t + 1$ which comes from position $x + i$. By summing over all i, we account for all possible paths the intensity value $I(x + n, 0)$ can diffuse through and arrive at position x at time $t + 1$.

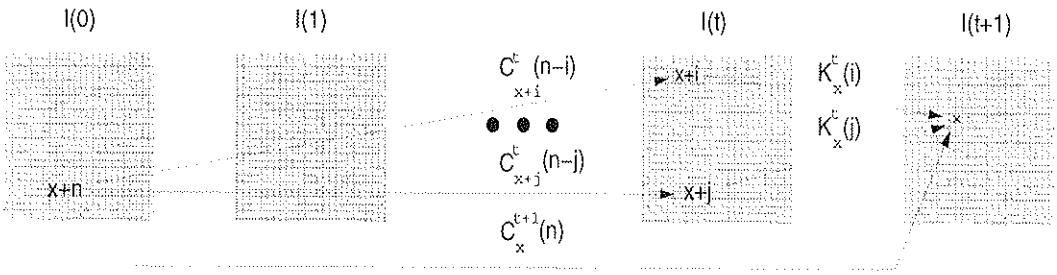

Figure 1: Illustration of the meaning of $C_{x+i}^t(j)$, $K_x^t(i)$ and $C_x^{t+1}(j)$ shown in two spatial dimensions. The composition of the two describe the contribution of an initial intensity value (far left) to one at the next time step (far right).

The limited support of the diffusion kernels C_x requires one additional modification to the algorithm. The value $C_x(j)$ can be thought of as the

percentage contribution of the initial image intensity at $x + j$ to the final image value at x. Given that the C_x are an array of percentages, their sum must be unity. If the support of the C_x is the entire image, this occurs naturally. However, when a more limited support is imposed, each C_x must be explicitly normalized after every time step in order to preserve the conservation of image intensity.

5 Kernel estimation.

Given the capability of constructing a kernel function using the process described in section 4, we are now faced with the problem of approximating an appropriate kernel function for an arbitrary image[2]. That is, given a novel input image, we want to estimate a set of kernels which will yield an image that is perceptually similar to the one that would have been obtained by integrating the diffusion equation. This process involves a number of steps. Even limiting the support of the kernels, the output of the estimation process is relatively high dimensional. Since the amount of data required to fit an n-dimensional function increases exponentially with the dimensionality (Duda and Hart, 1973), we must compress the kernel representation considerably. To accomplish this, we use principal components analysis (PCA). The result of the PCA is a set of principal components which we use as basis functions for kernel construction on an arbitrary image, and the coefficients of the expansion which must be estimated for each specific image.

5.1 Principal components analysis.

In the image processing literature principal components analysis is sometimes referred to as the Hotelling or discrete Karhunen-Loeve transformation. It is accomplished as follows (Gonzalez and Wintz, 1987). Given a set of kernels $C_{x,y}(i,j)$, we consider each n x n dimensional kernel to be one observation of n^2 random variables represented as a column vector **c**. The covariance matrix of these vectors is given by

[2]More precisely we assume the approximation is a function of the first order derivatives of image intensity, so the estimated function is of the form $\hat{G}(x, x', y, y', I_x, I_y)$

$$\Sigma_c = E\{(\mathbf{c} - \mathbf{m_c})(\mathbf{c} - \mathbf{m_c})^{\mathrm{T}}\} \tag{26}$$

Where $E\{\}$ is the expectation operator, $\mathbf{m_c} = \mathbf{E\{c\}}$ is the mean vector, and the superscript T indicates transposition. The eigenvectors $\mathbf{v_j}$ of Σ_c are the principal components we are seeking, while the eigenvalues of Σ_c reflect the amount of variance accounted for by the associated eigenvector. Since the covariance matrix is symmetric and positive definite, it is always possible to find a set of orthonormal eigenvectors (Anderson, 1971). The Hotelling transform consists of constructing a matrix A, the rows of which are the eigenvectors $\mathbf{v_j}$, and multiplying it by the centralized image vectors:

$$\mathbf{y} = \mathbf{A}(\mathbf{c} - \mathbf{m_c}) \tag{27}$$

Using the orthogonality of the rows of A, we can solve for the original image given the coefficient vector $\mathbf{y}$, and the mean vector $\mathbf{m_c}$:

$$\mathbf{c} = \mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{m_c} \tag{28}$$

Furthermore, if we construct A from the eigenvectors associated with the k largest eigenvalues ($k < n$), the reconstruction:

$$\hat{\mathbf{c}} = A_k^T\mathbf{y} + \mathbf{m_c} \tag{29}$$

is optimal in the mean-squared sense (in fact the error is given by the sum of the unused eigenvalues). The k-dimensional vectors $\mathbf{y}$ consist of the coefficients of the linear combination of principal components which optimally estimate the desired kernels.

In practice, noisy images tend to produce noisy diffusion kernels with correspondingly noisy principal components. This is due to the fact that the diffusion process fits the kernels to the exact noise present in the image. The number of principal components necessary for an accurate reconstruction can be dramatically reduced by first smoothing the diffusion kernels (we use a 7x7 Gaussian mask with a standard deviation of two pixels, although in practice almost any reasonable smoothing kernel works). The resulting

kernels yield images which are almost perceptually identical to the image derived using the unsmoothed diffusion kernels. The smoothing process also reduces the difficulty of the function approximation, as small errors in coefficient estimation do not produce dramatic differences in the derived images. An example of an image created by filtering with reconstructed kernels together with the components of the reconstruction is given in figure (2).

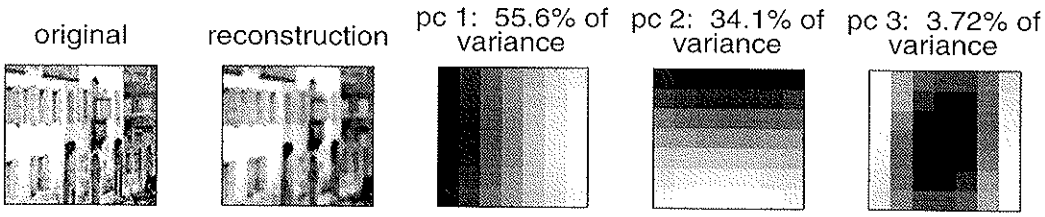

Figure 2: From left to right: the original image, the image filtered using a kernel function reconstructed from the 1st three principal components, as well as the first three principle components of the smoothed diffusion kernels, which account for $\approx 90\%$ of the variance.

5.2 Feature extraction.

The coefficients y_i discussed in the previous section constitute the targets of the function approximator. We now address the issue of its inputs. The features used to train the function approximator must have some degree of noise tolerance, while also representing the image structure in a neighborhood around the point of interest. A simple scheme which satisfies both these requirements is the use of a 3x3 window of the Gaussian smoothed image gradient magnitude based on x and y image derivatives generated by a Sobel operator. The 3x3 neighborhood gives the network both information on the dominant local orientation (if one exists), as well as the offset of any edges in the local region.

5.3 Coefficient estimation.

In order to estimate the coefficients y_i, given the multi-scale intensity gradient magnitudes, we train a multilayer perceptron using a modified[3], backpropagation algorithm (Werbos, 1974; Parker, 1985; Rumelhart et al., 1986; LeCun, 1985). Other nonlinear function estimators such as radial basis functions (Broomhead and Lowe, 1988), MARS (Friedman, 1991) or the II method (Breiman, 1991) are viable alternatives. Our task is simplified somewhat due to the uncorrelated nature of the principal components. This allows us to train separate networks for each component, obviating the need to learn nonlinear interactions between coefficients. Figure (3) illustrates the form that the coefficient images take, as well as the approximation after training.

6 Results.

We have applied the Green's function approximation or GFA algorithm to two different diffusion schemes. All results presented in this paper are from the same GFA filter except where specified. The filter is constructed by training the multilayer perceptron on coefficients derived from the single 64x64 pixel image shown in figure (2). The kernels are derived by iterating the Malik-Perona diffusion algorithm on the that image for 100 time steps. Note that all images are scaled so that their intensity values are between zero and one, all edge maps are generated using the same parameter set[4], and all diffusion generated images are created using the same number of iterations (100), and identical parameter values.

[3]In practice, the function approximation turns out to be quite sensitive, and extensive modifications of the backpropagation algorithm are required to successfully approximate the principal component coefficients. These include the use of momentum, an adaptive step size algorithm with a hard minimum, normalization of output distributions, as well as rejecting training epochs which result in an average error increase above a certain percentage.

[4]we use the Matlab 'edge' routine which employs Sobel operators and thresholded non-maximum suppression to generate edge maps.

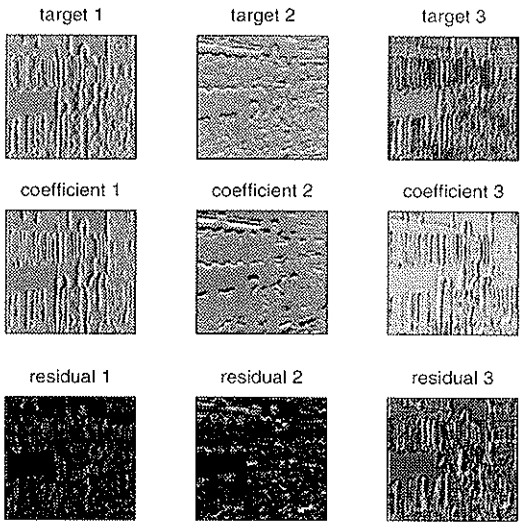

Figure 3: Top row: first three principal component coefficients at each point in the image (targets of training). Middle row: the coefficient approximation generated by the multilayer perceptron. Bottom row: residual error images generated by taking the absolute value of the difference of the images in the first two rows in the same column.

6.1 Gradient-based diffusion.

The first diffusion process we approximate is the Perona and Malik (Perona and Malik, 1987; Perona and Malik, 1990) scheme. They choose to use the simple nonlinear diffusion equation (4) with a conductance coefficient $c(\cdot)$ which is a decreasing function of the image intensity gradient magnitude. They give a number of mild constraints on the form of the function. One choice of $c(\cdot)$ they discuss is

$$c(|\nabla I|) = e^{-\left(\frac{|\nabla I|}{k}\right)^2} \tag{30}$$

Figure (4) displays the results of applying the Malik-Perona equation as well as the Green's Function Approximation (GFA) filter to a novel real-world image, corrupted with naturally occurring noise. Note the noise-enhancing property of the Malik-Perona process. Specifically, the speckle

noise in all three digits has been retained and expanded into white blocks in each character. The GFA filtered image does not evidence this effect, as the interior of all three characters is uniform.

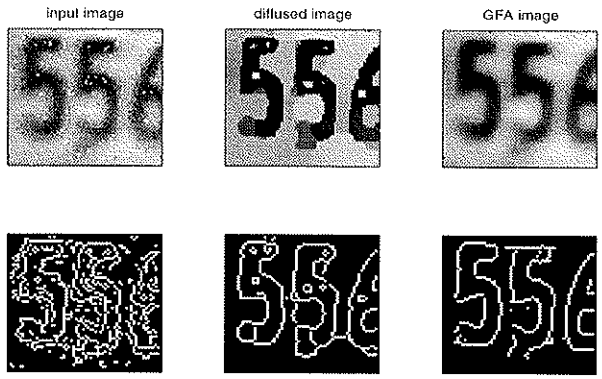

Figure 4: Top left: original image. Top center: Perona and Malik diffusion ($k = 0.05$). Top right: GFA filtered image. Bottom row: associated edge maps generated by non-maximum suppression of gradient magnitude with thresholding.

6.2 Mean curvature-based diffusion.

El-Fallah and Ford (El-Fallah and Ford, 1994) address the issue of the lack of noise-tolerance in the Malik and Perona equation. They propose a conductance function of the form:

$$c(|\nabla I|) = \frac{1}{\sqrt{1 + k^2 |\nabla I|^2}} \tag{31}$$

Equation (31) is derived by embedding the image in $\Re^3$, and setting the right hand side of (4) equal to $2H$, or twice the mean curvature of the surface. Solving this system for a conductance function yields the $c(\cdot)$ of equation (31). Figure (5) depicts the mean curvature based diffusion as well as its approximation. As can be seen, the mean curvature-based diffusion has excellent noise suppression qualities. The character outlines are quite clear, with only minor noise edges remaining. The GFA filter in this section

is generated by using kernels from the curvature-based diffusion process. It has properties similar to the filter used elsewhere in this paper. However, we have found that the edges and corners in the mean curvature diffusion tend to become slightly rounded, while characters are somewhat thickened. For these reasons, we choose to use the Malik-Perona diffusion as a model for our filter construction.

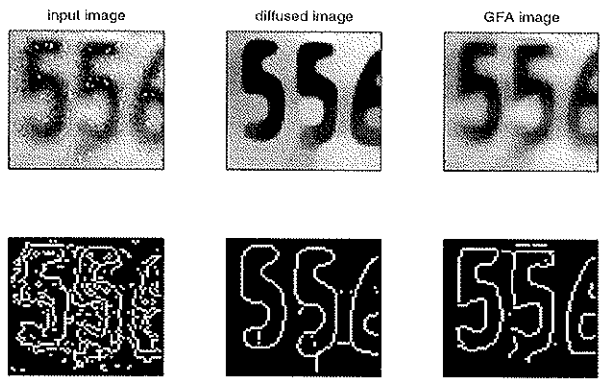

Figure 5: Top left: original image. Top center: image after mean curvature diffusion ($k = 50$). Top right: GFA filtered image. Bottom row: associated edge maps generated using the same technique as in figure (4).

6.3 Noise tolerance and comparison with median filtering.

While many filters are capable of reconstructing images for a given (known) noise distribution, the restoration problem is much more difficult if the noise characteristics are unknown or variable. The GFA filter noise-tolerance properties derive from a number of its components. The Gaussian blurred gradient magnitudes, coupled with the neural network, combine to form a robust function approximator in the presence of a wide assortment of noise profiles. In addition, the blurring of the kernels (as detailed in section 4) improves the noise tolerance considerably, as small errors in coefficient estimation result in correspondingly small deviations in the output image. In this section we compare the GFA filter to both the mean curvature diffusion process as well as a 5x5 median filter on noise corrupted images. The median filter is a simple image processing

technique[5] whose performance is generally excellent in the presence of noise.

The first noise model we use to illustrate the performance of the GFA filter is an additive Gaussian one. Figure (6) contrasts the GFA performance with that of a median filter applied to an image corrupted with zero mean, 0.25 variance additive Gaussian noise. The edge maps at the bottom indicate that the GFA filtered image contains most of the edge information of the image generated by applying the mean-curvature diffusion model at far less cost, while the median filtered image breaks down at this noise level.

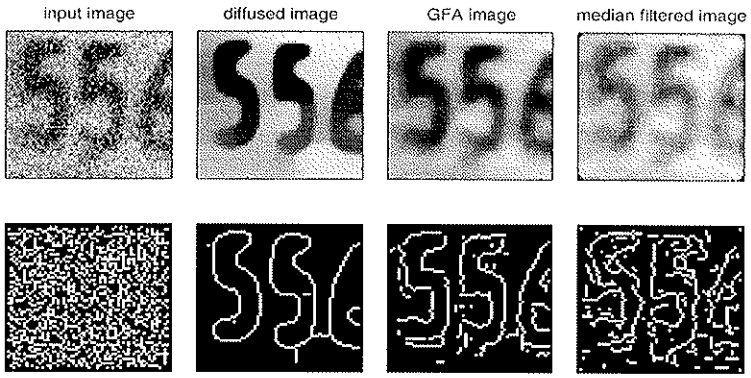

Figure 6: Comparison of the GFA filter with a median filter on an image corrupted by additive Gaussian noise (0 mean, 0.25 variance). Top row, from left to right: noisy original image, image after mean curvature diffusion ($k = 50$), GFA filtered image, median filtered image. Bottom row: associated edge maps.

Figure (7) illustrates the excellent performance of the GFA filter in the presence of multiplicative noise as compared to the median filter. Interestingly in this case, the full diffusion process washes away all detail of the fur, while the GFA filter eliminates most of the noise but retains much of the fur texture. Additionally, the diffusion generated image has fused the nose and mouth into one continuous region, while they remain separate after the application of the GFA filter.

[5]a median filter replaces the image value at each pixel with the median or central value of the intensity levels in a neighborhood of that pixel

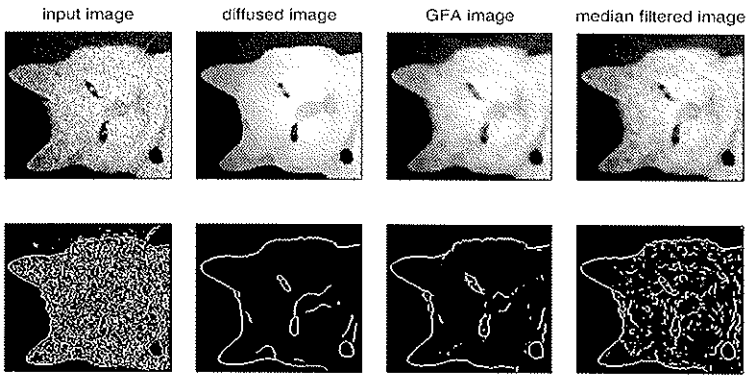

Figure 7: Comparison of the GFA filter with a median filter on an image corrupted by multiplicative (speckle) noise (amplitude=10%). Top row, from left to right: noisy original image, image after mean curvature diffusion ($k = 50$), GFA filtered image, median filtered image. Bottom row: associated edge maps.

As the last noise model, we demonstrate the GFA filter performance in the presence of salt and pepper noise (random pixel values replaced with zeros or ones). In this type of environment, the median filter produces excellent quality images, as evidenced by the building shown in figure (8). This is due to the invariance of the median filter to noise amplitude as opposed to the noise density which plagued it in the earlier examples. However, the mean curvature-based diffusion which performs so well in the presence of other types of noise, encounters difficulty with salt and pepper corruption. As can be seen, region boundaries for the triangular patches on the leftmost building have been lost, and new borders are generated which do not correspond to any real image feature. In contrast, the GFA filter retains most of the image structure, as is reflected in the edge maps at the bottom.

Finally, figures (9) and (10) present the performance of the GFA filter on a variety of images with different lighting conditions, textures, and noise types. As can be seen, the GFA filter is comparable to the full-scale diffusion process in all cases, and typically outperforms the median filter.

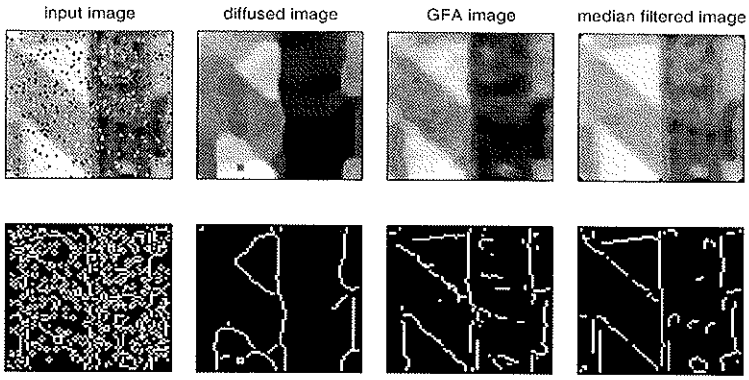

Figure 8: Comparison of the GFA filter with a median filter on an image corrupted by salt & pepper noise (density=10%). Top row, from left to right: noisy original image, image after mean curvature diffusion ($k = 50$), GFA filtered image, median filtered image. Bottom row: associated edge maps.

7 Conclusion.

Diffusion is a powerful tool of great potential utility in early vision. It unifies multi-scale processing into a simple procedure which reduces noise and integrates information at all scales of interest. However, it is a computationally costly and inherently serial process. In this paper we have presented a method which reduces both the computational cost of the algorithm as well as transforming the process into one that is amenable to parallelization. In addition, the GFA algorithm resolves many of the issues that are problematic for the Perona and Malik filtering. Regularization to improve equilibrium behavior becomes unnecessary as an appropriate time constant is implicitly imbedded in the system. Noise sensitivity can be dealt with separately in the feature extraction and training stages, and is thus no longer an issue for the PDE. Potential numerical instability of the underlying PDE is also eliminated as images for which the instability is manifest are excluded from the training set.

The Green's Function Approximation filter can also be compared with the nonlinear filtering scheme of Nitzberg and Shiota (Nitzberg and Shiota, 1992). Both construct sets of space-variant kernels based on local

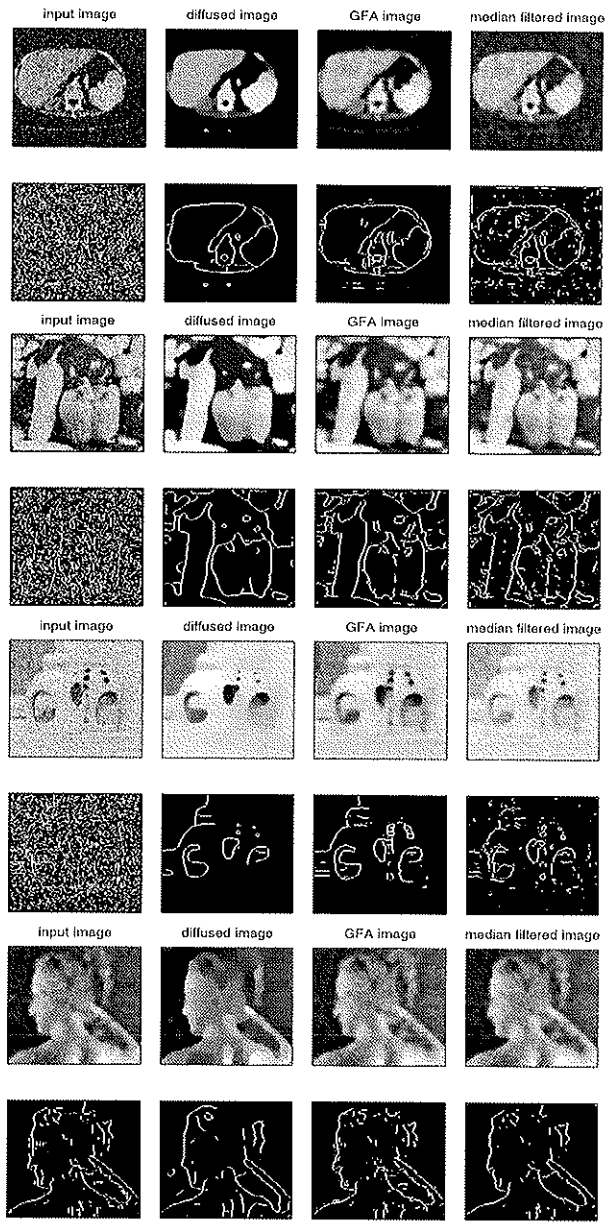

Figure 9: Comparison of the GFA filter with a median filter and curvature based diffusion on a variety of images.

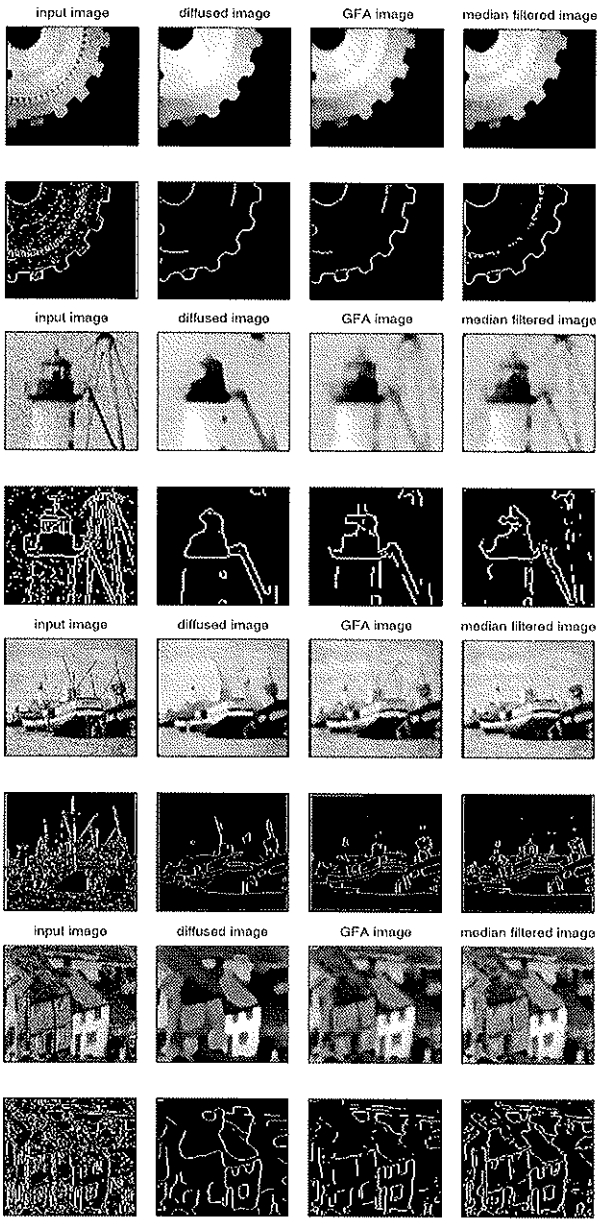

Figure 10: Comparison of the GFA filter with a median filter and curvature based diffusion on a variety of images.

image structure. The GFA filter learns the appropriate parameters from example, while the large number of constants in the Nitzberg-Shiota filter must be estimated by the user. After training, the GFA filter has no free parameters to be set by hand. The GFA filter is also capable of learning the result of repeated application of the Nitzberg-Shiota filter. Finally, if rotated, translated Gaussians are used to fit the diffusion kernels instead of PCA, the GFA would learn the parameters of the Nitzberg-Shiota filter. It would therefore provide a principled manner in which to select the many parameters of their technique based on local image structure.

From a biological standpoint, the GFA type approach seems well suited for use in the mammalian retino-cortical system. Both feature extraction via convolution and (not necessarily orthogonal) basis function expansion are natural operations for the neural substrate. The massive parallelism present in these systems allows the use of a large number of basis functions, input features, as well as units for learning and approximation, without sacrificing computational speed as on serial machines. This would dramatically increase the noise tolerance and accuracy of the process. The GFA filter is an example of a way in which a serial integration process can be parallelized either for a hardware implementation, or for possible use in a biological context.

The adaptive nature of the GFA allows us to model some developmental data as well. For example, it is known that kittens raised in an environment containing only vertical stimuli do not develop cortical cells responsive to horizontal orientations, as do normally raised animals (Blakemore and Cooper, 1970). This type of effect occurs naturally in the GFA filter. By training a filter on noisy vertical stimuli, the function approximator learns to ignore horizontal derivative information. This is illustrated in figure (11).

While many image enhancement processes yield impressive machine vision as well as psychophysical modeling results (Geman and Geman, 1984; Cohen and Grossberg, 1984; Grossberg and Mingolla, 1985) most of them suffer from the same drawbacks as the anisotropic diffusion paradigm: inherent serialism coupled with costly integration times. The methods outlined in this paper can be applied to each one of these techniques, allowing one to retain many of the benefits of the underlying procedure while improving their utility.

Many researchers have proposed modifications of the Malik-Perona

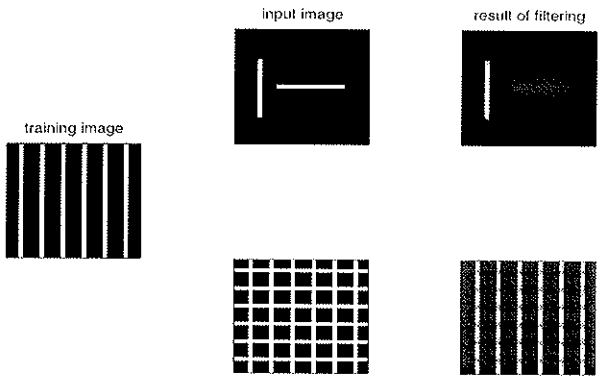

Figure 11: Example of simulated development in an environment containing only vertical stimuli (left). Subsequent horizontal stimuli (center) are ignored (right).

equation to improve its performance. For example, Whitaker (Whitaker and Pizer, 1991) suggested the use of the Gaussian smoothed image gradient in the conductance function, with the scale of the smoothing decreasing as time evolves. In this way, the Malik-Perona equation can be made noise-tolerant. Li and Chen (Li and Chen, 1994) point out that the parameter of the Malik-Perona equation (k in equation (30)) can also be made to be a function of time and possibly image structure, thereby improving the quality of images produced. Malik and Perona proposed the use of a Canny "noise estimator" (Canny, 1986) to choose an appropriate value of k. These types of modifications result in significant increases in the computational cost of the diffusion process. However, the GFA filter complexity is invariant to this type of modification of the diffusion algorithm. In fact, assuming a sufficiently rich feature set for the function approximator, arbitrarily complex local operations are possible in the conductance function without additional computational cost for the GFA.

In summary, we have presented a novel filter which learns an approximate Green's function from examples of a diffusion process. The learned filter can then directly generate output images which are good approximations of the full-scale diffusion equation. The technique is applicable to image processing algorithms other than anisotropic diffusion. The GFA filter is tolerant of a wide variety of noise environments, parameter free,

robust, and between one and two orders of magnitude faster than the full-scale anisotropic diffusion on a serial architecture, and is amenable to full parallelization.

Acknowledgments. Thanks to Giorgio Bonmassar and Michael Cohen for many helpful discussions.

References

Alvarez, L., Lions, P.-L., and Morel, J.-M. (1992). Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal of Numerical Analysis*, 29(3):845–866.

Alvarez, L. and Mazorra, L. (1994). Signal and image restoration using shock filters and anisotropic diffusion. *SIAM Journal of Numerical Analysis*, 31(2):590–605.

Anderson, T. W. (1971). *Introduction to Multivariate Statistics*. John Wiley and Sons, New York, second edition.

Barton, G. (1991). *Elements of Greens Functions*. Oxford University.

Blakemore, C. and Cooper, G. F. (1970). Development of the brain depends on visual environment. *Nature, London*, 228:477–478.

Breiman, L. (1991). The π method for estimating multivariate functions from noisy data. *Technometrics*, 33(2):125–144.

Broomhead, D. S. and Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355.

Burt, P. and Adelson, E. H. (1983). The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 9(4):532–540.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8:679–698.

Catte, F., Lions, P.-L., Morel, J.-M., and Coll, T. (1992). Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal of Numerical Analysis*, 29(1):182–193.

Cohen, M. A. and Grossberg, S. (1984). Some global properties of binocular resonances: Disparity matching, filling-in, and figure-ground synthesis. In Dodwell, P. and Caelli, T., editors, *Figural Synthesis*. Erlbaum, Hillsdale, NY.

Cottet, G.-H. and Germain, L. (1993). Image processing through reaction combined with nonlinear diffusion. *Mathematics of Computation*, 61(204):659–673.

Dang, T., Olivier, J., and Maitre, H. (1994). An image segmentation technique based on edge preserving smoothing filter and anisotropic diffusion. In *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 65–69.

Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York.

El-Fallah, A. I. and Ford, G. E. (1994). Nonlinear adaptive image filtering based on inhomogeneous diffusion and differential geometry. *SPIE Image and Video Processing II*, 2182:49–63.

Engquist, B., Lotstedt, P., and Sjogreen, B. (1989). Nonlinear filters for efficient shock computation. *Mathematics of Computation*, 52(186):509–537.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–50.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721–741.

Gonzalez, R. C. and Wintz, P. (1987). *Digital Image Processing*. Addison-Wesley Publishing Company, Reading, MA, second edition.

Grossberg, S. and Mingolla, E. (1985). Neural dynamics of form perception: Boundary completion, illusory figures, and neon color spreading. *Psychological Review*, 92(2):173–211.

Haberman, R. (1987). *Elementary applied partial differential equations.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, second edition.

Hummel, A. (1986). Representations based on zero-crossings in scale-space. In Fischler, M. and Firschein., O., editors, *Readings in Computer Vision: Issues, Problems, Principles and Paradigms.* Morgan Kaufmann, Los Angeles.

Illner, R. and Neunzert, H. (1993). Relative entropy maximization and directed diffusion equations. *Mathematical Methods in the Applied Sciences*, 17:545–554.

Kacur, J. and Mikula, K. (1995). Solution of nonlinear diffusion appearing in image smoothing and edge detection. *Applied Numerical Mathematics*, 50:47–59.

Klinger, A. (1971). Pattern and search statistics. In *Optimizing Methods in Statistics.* Academic Press, New York.

Koenderink, J. (1984). The structure of images. *Biological Cybernetics*, 50:363–370.

LeCun, Y. (1985). Une procedure d'apprentissage pour reseau a seuil assymetrique. *Cognitiva*, 85:599–604.

Li, X. and Chen, T. (1994). Nonlinear diffusion with multiple edginess thresholds. *Pattern Recognition*, 27(8):1029–1037.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London*, B 207:187–217.

Niessen, W., ter Haar Romney, B., and Viergever, M. (1994). Numerical analysis of geometry-driven diffusion equations. In Romeny, B. M. T. H., editor, *Geometry Driven Diffusion in Computer Vision*, Computational Imaging and Vision, chapter 15, pages 393–410. Kluwer.

Nitzberg, M. and Shiota, T. (1992). Nonlinear image filtering with edge and corner enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):826–833.

Nordstrom, N. K. (1990). Biased anisotropic diffusion: a unified regularization and diffusion approach to edge detection. *Image and Vision Computing*, 8(4):318–327.

Oram, M. W. and Perrett, D. I. (1992). Time course of neural responses discriminating different views of the face and head. *Journal of Neurophysiology*, 68(1):70–84.

Osher, S. and Rudin, L. I. (1990). Feature-oriented image enhancement using shock filters. *SIAM Journal of Numerical Analysis*, 27(4):919–940.

Parker, D. B. (1985). Learning-logic: Casting the cortex of the human brain in silicon. Technical Report TR-47, Center for Computational Research in Economics and Management Science, MIT, Cambridge, MA.

Pauwels, E. J., Proesmans, M., Gool, L. J. V., Moons, T., and Oosterlinck, A. (1993). Segmentation and image enhancement using coupled anisotropic diffusion. *SPIE*, 2094:836–847.

Perona, P. and Malik, J. (1987). Scale space and edge detection using anisotropic diffusion. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, pages 16–27, Miami, FL.

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

Perona, P., Shiota, T., and Malik, J. (1994). Anisotropic diffusion. In Romeny, B. M. T. H., editor, *Geometry Driven Diffusion in Computer Vision*, Computational Imaging and Vision, chapter 3, pages 73–92. Kluwer.

Price, C. B., Wambacq, P., and Oosterlinck, A. (1990). Image enhancement and analysis with reaction-diffusion paradigm. *IEE Proceedings*, 137 Pt. I(3):136–145.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature (London)*, 323:533–536.

Thorpe, S. J. and Imbert, M. (1989). Biological constraints on connectionist models. In Pfeifer, R., Schreter, Z., and Fogelman-Soulie, F., editors, *Connectionism in perspective*, pages 63–92. Elsevier, Amsterdam.

Vogels, R. and Orban, G. A. (1991). Quantitative study of striate single unit responses in monkey performing an orientation task. *Experimental Brain Research*, 84:1–11.

Werbos, P. J. (1974). *Beyond regression: new tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University.

Whitaker, R. T. (1993). Geometry-limited diffusion in the characterization of geometric patches in images. *CVGIP: Image Understanding.*, 57(1):111–120.

Whitaker, R. T. and Pizer, S. M. (1991). A multi-scale approach to nonuniform diffusion. *Computer Vision, Graphics and Image Processing.*, 57:99–110.

Witken, A. (1983). Scale-space filtering. In *International Joint Conference on Artificial Intelligence*, pages 1019–1021. Karlsruhe, West Germany.