Efficient and Consistent Recursive Filtering of Images with Reflective Extension

Ben Appleton[1] and Hugues Talbot[2]

[1] Intelligent Real-Time Imaging and Sensing Group, ITEE
The University of Queensland, Brisbane, QLD 4072, Australia
appleton@itee.uq.edu.au
[2] CSIRO Mathematical and Information Sciences,
Locked Bag 17, North Ryde, NSW 1670, Australia
hugues.talbot@csiro.au

Abstract. Recursive filters are commonly used in scale space construction for their efficiency and simple implementation. However these filters have an initialisation problem which either produces unusable results near the image boundaries or requires costly approximate solutions such as extending the boundary manually.
In this paper, we describe a method for the recursive filtering of reflectively extended images for filters with symmetric denominator. We begin with an analysis of reflective extensions and their effect on non-recursive filtering operators. Based on the non-recursive case, we derive a formulation of recursive filtering on reflective domains as a linear but time-varying implicit operator. We then give an efficient method for decomposing and solving the linear implicit system. This decomposition needs to be performed only once for each dimension of the image. This yields a filtering which is both stable and consistent with the ideal infinite extension. The filter is efficient, requiring the same order of computation as the standard recursive filtering.
We give experimental evidence to verify these claims.

1 Introduction

Recursive filters such as those described in [1] are commonly used in scale space construction for their efficiency and simple implementation. However these filters have an initialisation problem which produces unusable results near the image boundaries or requires costly approximations such as extending the boundary manually. These problems are exacerbated by the high scales encountered in linear scale space construction.

Martucci [2] investigated the relationships between symmetric convolution and the various Discrete Trigonometric Transforms. He demonstrated that DTTs could be used for an efficient implementation of the linear filtering of finite signals with reflective boundary conditions. However as Deriche observed in [3], methods relating to the Fast Fourier Transform may take many orders of magnitude greater computational effort than a direct implementation for recursive filters of low order.

A reflectively extended signal of length N may be treated as a periodic signal of length $2N$. Cunha [4] considered the problem of computing the initial values for a recursive filter on a periodically extended domain. He noted that the solution required more effort than the actual filtering. Smith and Eddins [5] suggest explicitly computing the impulse response of the recursive filter and using this to compute the initial values. As given, their solution is restricted to filters that have only single poles. Both methods suffer from the additional disadvantage that they must be repeated for each row of the image being filtered.

In [6] Weickert, ter Haar Romeny and Viergever give a method for recursive Gaussian filtering on a reflective domain. They derive their method from the relationship between linear diffusion filtering and Gaussian convolution, using a first order semi-implicit approximation to a linear diffusion PDE on a domain with Neumann boundary conditions. Their method requires a number of iterations proportional to the variance of the Gaussian. Their implementation of a single iteration is similar to the general method developed here.

In this paper, we describe a method for the recursive filtering of reflectively extended images. In Section 2 we introduce discrete filtering on infinite and finite domains. Section 3 develops a theory of filtering on finite domains with reflective extension. This theory forms the basis of the method for recursive filtering presented in Section 4. Here we analyse the existence and numerical stability of the scheme and propose an efficient implementation. Section 5 gives results on the accuracy and timings.

2 Discrete Filtering

2.1 Filtering on Infinite Domains

Linear filtering is widely used in image analysis for feature extraction, coding, enhancement and transformation. Here we briefly introduce non-recursive and recursive filtering, focussing in particular on the formulation of linear filtering as a system of explicit or implicit linear equations.

The simplest form of linear filter is the non-recursive filter, also known as the moving average or finite impulse response filter. Consider a signal x an element of the real vector space $V(\mathbb{Z})$, and a filter h. Then the filtering of x by h is defined by the convolution

$$(h \star x)[i] = \sum_{j=-\infty}^{\infty} h[j]x[i-j]$$

where h is known as the *kernel* of the filter. Here we assume h is stable in the sense that a bounded input produces a bounded output.

For a filter $h = (\ldots, h_{-2}, h_{-1}, h_0, h_1, h_2, \ldots)$ we may express this relationship as the (infinite) matrix-vector product

$$y = Hx \tag{1}$$

where y is the result of the convolution. Here $H_{ij} = h[i-j]$. For a filter h of length b, H is a banded Toeplitz matrix with total bandwidth b.

Recursive filters, also known as autoregressive or infinite impulse response filters, are expressed implicitly via a convolution. Let x be the input to a recursive filter with kernel h and let y be the output. We solve the implicit system

$$x = h \star y$$

to obtain y. Unfortunately the term 'recursive filter' is also used to describe the combination of a non-recursive filter and a recursive filter. Their sequential separability allows us to focus on the implementation of purely recursive filters with the natural extension to more general filters implicit throughout.

As with non-recursive filtering, we may implicitly define recursive filtering in matrix notation

$$x = Hy \tag{2}$$

where we solve the implicit system to obtain y. Typically a recursive filter will require only low order to approximate a desired impulse response, producing a narrowly banded matrix H [7].

2.2 Filtering on Finite Domains

The definitions given above apply only to signals on infinite discrete domains. When presented with a finite signal defined on the discrete interval $[1, N]$ we must define the action of a filter on that signal. To do so we define an extension of the signal to the domain $V(\mathbb{Z})$ and induce the definition of filtering from the choice of extension.

Zero Extension The simplest extension is the zero extension. This extension assigns the signal zero value outside of $[1, N]$. For the filter $h = (\ldots, h_{-2}, h_{-1}, h_0, h_1, h_2, \ldots)$ we obtain the filter matrix

$$H_0 = \begin{bmatrix} h_0 & h_{-1} & h_{-2} & \ldots & h_{2-N} & h_{1-N} \\ h_1 & h_0 & h_{-1} & \ldots & h_{3-N} & h_{2-N} \\ h_2 & h_1 & h_0 & \ldots & h_{4-N} & h_{3-N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{N-2} & h_{N-3} & h_{N-4} & \ldots & h_0 & h_{-1} \\ h_{N-1} & h_{N-2} & h_{N-3} & \ldots & h_1 & h_0 \end{bmatrix}$$

Periodic Extension Another possibility is to periodically extend the signal beyond the original finite domain. The resulting filter operation is a circular

convolution. It may be expressed in matrix notation using Equation 1 where

$$H_P = \begin{bmatrix} h_0 & h_{-1} & h_{-2} & \dots & h_2 & h_1 \\ h_1 & h_0 & h_{-1} & \dots & h_3 & h_2 \\ h_2 & h_1 & h_0 & \dots & h_4 & h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{-2} & h_{-3} & h_{-4} & \dots & h_0 & h_{-1} \\ h_{-1} & h_{-2} & h_{-3} & \dots & h_1 & h_0 \end{bmatrix}$$

is a circulant matrix. Circulant matrices have a close connection to the Discrete Fourier Transform which may be used for fast multiplication in Equation 1 or fast inversion in Equation 2 [8].

3 Reflective domains

3.1 Indexing

Both zero extension and periodic extension have undesirable effects on the image border. Zero extension introduces large discontinuities on the order of the signal magnitude at the image borders. Periodic extension introduces a large discontinuitiy when connecting the two endpoints of the signal.

As a more appropriate alternative, the reflective extension of a signal can be described as placing a mirror at each end of the finite domain. In the case of Gaussian blurring this is equivalent to imposing an adiabatic boundary on the equivalent linear diffusion problem. This maintains the sum of intensities in the image, a desirable property in scale space construction. In the discrete case this is equivalent to reflective extension with repetition of the endpoints, known in the context of the Discrete Cosine Transforms as type-2 symmetric extension [7].

It is instructive in the following to consider the algebra of reflective indexing. For a signal defined on a finite discrete interval $I = \{1, 2, \dots, N\}$ we define the following reflectively extended domain:

$$R = I \times \{+, -\}$$

with the group $\mathbb{Z}_{2N}$ acting on it by addition. For $j \in \mathbb{Z}_{2N}$, $i \in I$, we have

$$\begin{aligned} j + (i, +) &= (i + j, +) \\ j + (i, -) &= (i - j, -) \\ (i, +) &= (2N + 1 - i, -) \end{aligned}$$

Observe that the action of $\mathbb{Z}_{2N}$ on R is isomorphic to the additive group $\mathbb{Z}_{2N}$. The following diagram depicts the increment action on R:

$$\begin{array}{ccccccc} (1,+) & \overset{+1}{\rightarrow} & (2,+) & \overset{+1}{\rightarrow} & \dots & \overset{+1}{\rightarrow} & (N-1,+) & \overset{+1}{\rightarrow} & (N,+) \\ \uparrow{\scriptstyle +1} & & & & & & & & \downarrow{\scriptstyle +1} \\ (1,-) & \overset{+1}{\leftarrow} & (2,-) & \overset{+1}{\leftarrow} & \dots & \overset{+1}{\leftarrow} & (N-1,-) & \overset{+1}{\leftarrow} & (N,-) \end{array}$$

3.2 Convolution

The definition of the algebra of indices in R allows us to define a convolution in this space. For a signal $g \in V(R)$ and a convolution kernel $h \in V(\mathbb{Z}_{2N})$ we define the convolution as

$$(h \star g)[r] = \sum_{j=1}^{2N} g[r]h[j-r]$$

the standard periodic convolution on a domain of size $2N$. As noted earlier, the operator $(h\star)$ may also be expressed as the circulant matrix $H_{ij} = h[i-j]$.

We seek to obtain a natural definition for the reflective convolution $h\odot$ in $V(I)$. To do so, we define an epimorphism ϕ from $V(I)$ to $V(R)$ and a projection π from $V(R)$ to $V(I)$. The definition for the reflective convolution is then induced by the following diagram

$$\begin{array}{ccc} V(I) & \xrightarrow{\phi} & V(R) \\ \downarrow h\odot & & \downarrow h\star \\ V(I) & \xleftarrow{\pi} & V(R) \end{array}$$

Let $\phi : V(I) \to V(R)$ be defined as follows. If $f \in V(I)$ then

$$\phi(f)(i,+) = \phi(f)(i,-) = f[i]$$

Then ϕ is the canonical *reflective extension* epimorphism from $V(I)$ into $V(R)$. We may express ϕ as the $2N \times N$ block matrix:

$$\Phi = \begin{bmatrix} I_N \\ J_N \end{bmatrix}$$

where I_N is the $N \times N$ identity matrix and J_N is the $N \times N$ reflection matrix

$$J_N = \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix}$$

Then $\phi(f) = \Phi f$.

Let $\pi : V(R) \to V(I)$ be defined as follows: if $g \in V(R)$ then

$$\pi(g)[i] = \frac{1}{2}\left(g(i,+) + g(i,-)\right)$$

Then π is a projection from $V(R)$ to $V(I)$. Here we have chosen π such that $\phi\pi$ is the identity mapping and π acts symmetrically in the indices of R. We may express π in matrix form as

$$\Pi = \frac{1}{2}\Phi^T = \frac{1}{2}\begin{bmatrix} I_N & J_N \end{bmatrix}$$

We then obtain the definition for reflective convolution:

$$h \odot f = \pi \left(h \star \phi \left(f \right) \right)$$

Observe that, as the composition of linear operators, $(h \odot)$ is linear. We express it in matrix form as

$$H_{\odot} = \Pi H \Phi = \frac{1}{2} \Phi^T H \Phi \tag{3}$$

3.3 Equivalence to Symmetric Filtering

We now analyse the structure of $H_{\odot}$. Recall that H is the circulant $2N \times 2N$ matrix corresponding to periodic convolution in $V(R)$ by the kernel h. We partition it into a 2×2 block matrix as follows:

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

As H is cyclic it is also block cyclic, so $H_{11} = H_{22}$ and $H_{12} = H_{21}$. We can now compute $H_{\odot}$ as follows:

$$\begin{aligned} H_{\odot} &= \frac{1}{2} \begin{bmatrix} I_N & J_N \end{bmatrix} \begin{bmatrix} H_{11} & H_{12} \\ H_{12} & H_{11} \end{bmatrix} \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\ &= \frac{1}{2} \left(H_{11} + H_{12} J_N + J_N H_{12} + J_N H_{11} J_N \right) \\ &= \frac{1}{2} \left(H_{11} + H_{11}^T \right) + \frac{1}{2} \left(H_{12} + H_{12}^T \right) J_N \end{aligned}$$

We observe that $H_{\odot}$ is symmetric. In fact, consider the decomposition of our filter $h \in V(\mathbb{Z})$ into the sum of an odd and an even function $h = h^{\text{even}} + h^{\text{odd}}$. The equivalent circulant matrix is $H = H^{\text{even}} + H^{\text{odd}}$, where $\left(H^{\text{even}}\right)^T = H^{\text{even}}$ and $\left(H^{\text{odd}}\right)^T = -H^{\text{odd}}$. Computing H gives

$$\begin{aligned} H &= \frac{1}{2} \left(\left(H_{11}^{\text{even}} + \left(H_{11}^{\text{even}}\right)^T + H_{11}^{\text{odd}} + \left(H_{11}^{\text{odd}}\right)^T \right) \right) \\ &\quad + \frac{1}{2} \left(\left(H_{12}^{\text{even}} + \left(H_{12}^{\text{even}}\right)^T + H_{12}^{\text{odd}} + \left(H_{12}^{\text{odd}}\right)^T \right) J_N \right) \\ &= H_{11}^{\text{even}} + H_{12}^{\text{even}} J_N \end{aligned}$$

We see that the anti-symmetric component of our filter h does not contribute to the reflective convolution in $V(I)$. By our choice of projection π all reflective recursive filters are equivalent to an odd-length symmetric filter. Consequently, in the following we assume without loss of generality that h is symmetric about the index $0 \in \mathbb{Z}$ and hence that H is symmetric. Practically, *this constraint applies only to the recursive component of the filter.*

4 Recursive Filtering in a Reflective Domain

4.1 Existence

We now demonstrate that recursive filtering in $V(I)$, defined as the solution of Equation 2 for the filter matrix $H_\odot$, is always possible for an invertible kernel h.

If $(h\star)$ is invertible in $V(R)$ then H is invertible as a matrix. Observe that if H is invertible then so is $H_\odot$, with its inverse given by:

$$H_\odot^{-1} = \Phi^T H^{-1} \Pi^T$$

as

$$\begin{aligned}
H_\odot^{-1} H_\odot &= \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} \begin{bmatrix} I_N \\ J_N \end{bmatrix} \begin{bmatrix} I_N & J_N \end{bmatrix} H \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\
&= \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} \left(I_{2N} + J_{2N}\right) H \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\
&= \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} H \begin{bmatrix} I_N \\ J_N \end{bmatrix} + \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} J_{2N} H \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\
&= \frac{1}{2} I_N + \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} H^T J_{2N} \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\
&= \frac{1}{2} I_N + \frac{1}{4} \begin{bmatrix} I_N & J_N \end{bmatrix} H^{-1} H \begin{bmatrix} I_N \\ J_N \end{bmatrix} \\
&= I_N
\end{aligned}$$

4.2 Stability

From this we may bound the condition number κ_∞ of $H_\odot$, a measure of the numerical stability of the corresponding implicit system [8].

$$\begin{aligned}
\kappa_\infty\left(H_\odot\right) &= \left\|H_\odot\right\|_\infty \left\|H_\odot^{-1}\right\|_\infty \\
&= \left\|\Pi H \Phi\right\|_\infty \left\|\Phi^T H^{-1} \Pi^T\right\|_\infty \\
&\leq \frac{1}{4} \left\|\Phi^T\right\|_\infty \left\|H\right\|_\infty \left\|\Phi\right\|_\infty \left\|\Phi^T\right\|_\infty \left\|H^{-1}\right\|_\infty \left\|\Phi\right\|_\infty \\
&\leq \left\|H\right\|_\infty \left\|H^{-1}\right\|_\infty \\
&\leq \kappa_\infty\left(H\right)
\end{aligned}$$

Here we have noted that $\left\|\Phi\right\|_\infty = 2$ and $\left\|\Phi^T\right\|_\infty = 1$. So we see that the inversion of the reflective convolution operator $(h\odot)$ on $V(I)$ is numerically at least as stable as the inversion of the periodic convolution operator $(h\star)$ on $V(R)$.

4.3 LDL^T Decomposition and Solution

Let $A \in I\!R^{n\times n}$ be a symmetric, positive definite matrix with real coefficients. Then A may be decomposed into the product $A = LDL^T$ where L is a unit

lower triangular matrix and D a diagonal matrix. An algorithm for the numerical implementation of this decomposition is described in [8]. For an $N \times N$ matrix with total bandwidth b it requires $O(Nb^2)$ computation and $O(Nb)$ storage.

We will not always be dealing with positive definite matrices. The condition of positive definiteness may be relaxed to a more general existence criterion for the LDL^T decomposition of a symmetric matrix A: All principal submatrices of A must have non-zero determinant. So

$$\det A(1:s, 1:s) \neq 0 \quad 1 \leq s < n$$

where $A(1:s, 1:s)$ denotes the intersection of the first s rows and columns of A. To show this consider the following: $A(1:s, 1:s) = L(1:s, 1:s)D(1:s, 1:s)L(1:s, 1:s)^T$ as L is lower triangular, so

$$\begin{aligned} \det A(1:s, 1:s) &= \det L(1:s, 1:s) \det D(1:s, 1:s) \det L^T(1:s, 1:s) \\ &= \det D(1:s, 1:s) \\ &= \prod_{i=1}^{s} D_{ii} \end{aligned}$$

where we have observed that $\det L(1:s, 1:s) = 1$ as L is unit lower triangular. So

$$D_{ii} = \frac{\det A(1:i, 1:i)}{\det A(1:i-1, 1:i-1)}$$

where we take $D_{11} = A_{11}$. As we will be solving the system in Equation 2 using the LDL^T decomposition we require that $\det A(1:s, 1:s) \neq 0$ for $1 \leq s \leq n$.

Once the LDL^T decomposition has been computed it is simple to solve Equation 2 to recursively filter a signal x to obtain the filtered signal y. In sequence solve:

$$x = Lu \tag{4}$$

$$u = Dv \tag{5}$$

$$v = L^T y \tag{6}$$

The solution to Equations 4, 5, and 6 may be considered as the sequential application of a causal filtering, a point-scaling, and an anti-causal filtering. This requires $O(Nb)$ computation for a kernel h of length b and may be solved in place. In fact, it requires the same number of arithmetic operations per pixel as the standard recursive filtering on an infinite domain.

When implementing the LDL^T decomposition in finite numerical precision the stability criterion is very simple to verify. As each D_{ii} is computed simply check that it is sufficiently far from zero:

$$|D_{ii}| > \eta$$

where η is a user-defined threshold for numerical stability, eg. $\eta = 10^{-7}$.

We have observed that quite surprisingly the LDL^T decomposition of $H_\odot$ appears to always exist for an invertible filter h. All filters which produce a non-decomposable matrix $H_\odot$ are not invertible in $V(Z)$. While the authors have proved this for low filter orders the general proof remains an open problem.

4.4 A Recursive Filtering Algorithm for Reflective Domains

Here we give an algorithm for the recursive filtering of images with reflective extension.

Algorithm For each image axis:

1. Form the $H_{\odot}$ matrix along the current axis according to Equation 3
2. Compute the decomposition $LDL^T = H_{\odot}$
3. For each row of data x, solve in place Equations 4, 5, 6.

The decomposition is only computed once per image axis. The total amount of computation on a d-dimensional image of sidelength N is $O(N^d b + Nb^2)$. When applied to images of dimension $d \geq 2$ with a typical recursive filter of low order $b \ll N^{d-1}$, this reduces to $O(N^d b)$. This is the same order of computation as the standard implementation of a recursive filter. The algorithm requires auxiliary storage of $O(Nb)$ which is trivial when compared to the size of the image.

5 Application to Deriche's Recursive Gaussian Approximations

Here we consider Deriche's approximations to filtering by Gaussians and their derivatives, which require scale-invariant computation in theory [3]. However on a finite domain the standard implementation manually extends the boundary by an amount proportional to the scale of the Gaussian in order to minimise border effects. As a result the application of these filters to images is not truly invariant to scale and is doubly approximate, approximating both the Gaussian's impulse response and its effect on a finite domain.

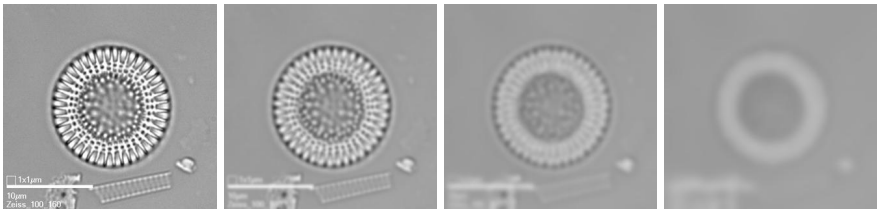
We apply the new filtering scheme developed here to both symmetric and anti-symmetric filters. Figure 1 demonstrates the application of Deriche's 4th order Gaussian approximation to a microscope image of a diatom, while Figure 2 demonstrates the application of Deriche's 4th order derivative of Gaussian approximation to computing its spatial gradient. The application of the new scheme to other filters is straightforward.

Deriche's 4th order approximation to a Gaussian of scale σ for $x \geq 0$ is:

$$e^{-\frac{x^2}{2\sigma^2}} \approx \left(1.68\cos\left(0.6318\frac{x}{\sigma}\right) + 3.735\sin\left(0.6318\frac{x}{\sigma}\right)\right)e^{-1.783\frac{x}{\sigma}} - \left(0.6803\cos\left(1.997\frac{x}{\sigma}\right) + 0.2598\sin\left(1.997\frac{x}{\sigma}\right)\right)e^{-1.723\frac{x}{\sigma}}$$

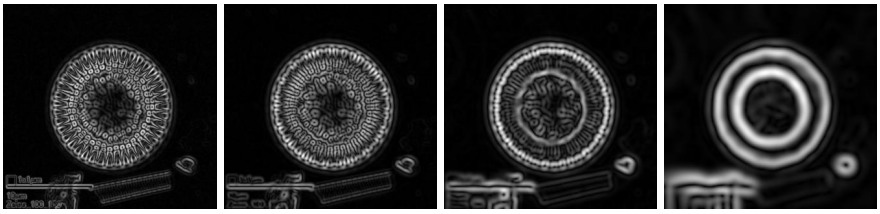
His 4th order approximation to the first derivative of a Gaussian is:

$$xe^{-\frac{x^2}{2\sigma^2}} \approx \left(-0.6472\cos\left(0.6719\frac{x}{\sigma}\right) - 4.531\sin\left(0.6719\frac{x}{\sigma}\right)\right)e^{-1.527\frac{x}{\sigma}} + \left(0.6494\cos\left(2.072\frac{x}{\sigma}\right) + 0.9557\sin\left(2.072\frac{x}{\sigma}\right)\right)e^{-1.516\frac{x}{\sigma}}$$



(a) (b) (c) (d)

Fig. 1. Symmetric filtering: Gaussian blurring of *Cyclostephanos Dubius* (329×303). (a) The original image. (b) $\sigma = 2$. (c) $\sigma = 4$. (d) $\sigma = 8$.



(a) (b) (c) (d)

Fig. 2. Anti-symmetric filtering: the gradient magnitudes of *Cyclostephanos Dubius*. (a) $\sigma = 1$. (b) $\sigma = 2$. (c) $\sigma = 4$. (d) $\sigma = 8$.

Deriche shows how these one-sided approximations may be used to compute a causal and anti-causal filter whose sum approximates a Gaussian of the desired scale. The causal filter is of the form:

$$H_+(z^{-1}) = \frac{n_{00}^+ + n_{11}^+ z^{-1} + n_{22}^+ z^{-2} + n_{33}^+ z^{-3}}{1 + d_{11} z^{-1} + d_{22} z^{-2} + d_{33} z^{-3} + d_{44} z^{-4}}$$

while the anti-causal filter is of the form:

$$H_-(z) = \frac{n_{11}^- z + n_{22}^- z^2 + n_{33}^- z^3 + n_{44}^- z^4}{1 + d_{11} z + d_{22} z^2 + d_{33} z^3 + d_{44} z^4}$$

These may be combined to produce a symmetric filter

$$\begin{aligned} H(z^{-1}) &= H_+(z^{-1}) + H_-(z) \\ &= \frac{n_0 + n_1(z^{-1} + z) + n_2(z^{-2} + z^2) + n_3(z^{-3} + z^3)}{d_0 + d_1(z^{-1} + z) + d_2(z^{-2} + z^2) + d_3(z^{-3} + z^3) + d_4(z^{-4} + z^4)} \end{aligned}$$

or an anti-symmetric filter

$$H(z^{-1}) = H_+(z^{-1}) - H_+(z)$$

$$= \frac{n_1(z^{-1} - z) + n_2(z^{-2} - z^2) + n_3(z^{-3} - z^3) + n_4(z^{-4} - z^4)}{d_0 + d_1(z^{-1} + z) + d_2(z^{-2} + z^2) + d_3(z^{-3} + z^3) + d_4(z^{-4} + z^4)}$$

Observe that the denominators are symmetric in both cases. For any anti-symmetric filter realisable by a stable recursive filter we may always manipulate the filter into a form with symmetric denominator. Anti-symmetric denominators are not considered because they are unstable, having a pole at $z = 1$.

Due to Deriche's construction of the anti-causal filter from the causal filter to ensure symmetry, 4th order terms in the numerator of $H(z^{-1})$ cancel to reduce the order of the symmetric filtering. Likewise in the anti-symmetric case the constant term in the numerator cancels. These improve the efficiency of the filter and may be considered a benefit of solving an inherently symmetric problem in a symmetric manner.

The non-recursive and recursive components of Deriche's filtering are performed in sequence. The non-recursive component is performed by manually extending the image by the highest power in the numerator. The recursive component uses the method proposed in this paper.

6 Results

All tests were performed on a 700MHz Toshiba P-III laptop with 192MB of RAM under the Linux operating system. The algorithm presented here has been implemented in double precision floating point arithmetic in C and has not been optimised significantly.

6.1 Accuracy

We apply Deriche's 4th order recursive Gaussian approximation to blur the image of Figure 3 and compare this to a ground truth result. The ground truth result is obtained by symmetrically extending the image before filtering with a zero extension implementation of Deriche's 4th order recursive Gaussian. The border is extended by 20σ so that border artifacts do not contribute measureably to the error. For a Gaussian of scale $\sigma = 10$ the error has root mean square magnitude 4.8×10^{-8} with peak magnitude 2.1×10^{-7}. This error is trivially small compared both to the amplitude of the image and to the error of Deriche's 4th order Gaussian approximation with relative root mean squared magnitude of 2.93×10^{-4}.

6.2 Timing

Here we compare the running time of the algorithm proposed in this paper with the standard implementation via border extension. Borders are manually extended by 4σ as a reasonable tradeoff between additional computation and border effects. We consider a range of image sizes and scales for 2D and 3D images. Although we have chosen here for simplicity to test square and cubic

(a) (b) (c) (d)

Fig. 3. Gaussian blurring of a golf cart image (548×509). (a) The original image. (b) Ground truth, $\sigma = 10$. (c) Proposed algorithm, $\sigma = 10$. (d) Difference between (b) and (c), scaled by 1.2×10^9 in order to be visible.

images the filter decomposition has been repeated for each image axis. Results for 2D images are given in Tables 1 and 2 while results for 3D images are given in Tables 3 and 4. Over the range of image sizes considered we observe that in 2D this method is faster for scales greater than or equal to one fifth of the image size, while in 3D it is faster for all scales. Finally we observe that our method has a constant computing time irrespective of scale σ.

7 Conclusion

In this paper we have described a method for the recursive filtering of reflectively extended images by filters with symmetric denominator. This method is efficient and consistent with the infinite reflectively extended case. It is based on a time-varying formulation of convolution derived by considering the algebra of reflective extensions. In the recursive case this leads to an implicit linear system whose solution for invertible filters was shown to exist and have the same numerical stability as the corresponding recursive filter in the symmetrically extended domain. An efficient algorithm was given for the decomposition and solution of this implicit system. The solution requires the same order of computation as a standard recursive filtering while the matrix decomposition needs to be performed only once along each axis.

The method was applied to both symmetric and anti-symmetric filters. It has been demonstrated on Deriche's recursive approximations to filtering by a Gaussian and its first derivative. Results demonstrate that the proposed method has excellent numerical accuracy. Its speed is similar to a standard implementation of recursive filtering on 2D images and is faster on 3D images.

Acknowledgements

The diatom image in Figures 1 and 2 was taken from the ADIAC public data web page:

Table 1. Running times (ms) for Deriche's 4th order Gaussian approximation on 2D images, implemented with manual extension by 4σ.

Sidelength \ σ	10	20	50	100	200	500	1000
50	2.5	4	5.5	9	17	74	109.5
100	9.5	10	12.5	19.5	32.5	148.5	216.5
200	28	28.5	37.5	50.5	74.5	307	449
500	151.5	155	173	207	267.5	852.5	1214
1000	647.5	644.5	689	753.5	876.5	2058.5	2942.5

Table 2. Running times (ms) for the algorithm presented here on 2D images.

Sidelength \ σ	10	20	50	100	200	500	1000
50	3.5	2	3	2	1.5	3	2.5
100	7.5	7.5	7.5	6.5	8	5.5	8
200	31	30	30.5	31	30.5	30	29.5
500	199	194.5	194	193.5	195	196.5	195
1000	875.5	859.5	857.5	857.5	856	867.5	858.5

Table 3. Running times (ms) for Deriche's 4th order Gaussian approximation on 3D images, implemented with manual extension by 4σ.

Sidelength \ σ	2	5	10	20	50	100
10	3.5	3	3.5	7.5	13	22
20	12	13	17	26.5	50	89
50	130	143	168.5	218.5	368	618.5
100	998.5	1063.5	1160	1351.5	1973.5	2952

Table 4. Running times (ms) for the algorithm presented here on 3D images.

Sidelength \ σ	2	5	10	20	50	100
10	2	2.5	2.5	1.5	3.5	4
20	7.5	8.5	8	6	9	7
50	109	108.5	107	109	111	109
100	919	922.5	919	918	929.5	926

http://www.ualg.pt/adiac/pubdat/pubdat.html (CEC contract MAS3-CT97-0122). The author would like to thank Peter Kootsookos of the University of Queensland and David Chan and Carolyn Evans of CSIRO Mathematical and Information Sciences for interesting discussions and assistance in proof-reading this paper.

References

1. Deriche, R.: Fast algorithms for low-level vision. IEEE Tr. on Pattern Analysis and Machine Intelligence **12** (1990) 78–87
2. Martucci, S.A.: Symmetric convolution and the discrete sine and cosine transforms. IEEE Transactions on Signal Processing **42** (1994) 1038–1051
3. Deriche, R.: Recursively implementing the gaussian and its derivatives. Technical Report 1893, Programme 4 - Robotique, Image et Vision, INRIA - Institut National en Informatique et en Automatique (1993)
4. da Cunha, A.M.: Espaços de escala e detecção de arestas. Master's thesis, IMPA, Rio de Janeiro (2000) http://www.visgraf.impa.br/escala.html.
5. Smith, M.J.T., Eddins, S.L.: Analysis/synthesis techniques for subband image coding. IEEE Transactions on Acoustics, Speech, and Signal Processing **38** (1990) 1446–1456
6. Weickert, J., ter Haar Romeny, B.M., Viergever, M.A.: Efficient and reliable schemes for nonlinear diffusion filtering. IEEE Transactions on Image Processing **7** (1998) 398–410
7. Oppenheim, A.V., with John R. Buck, R.W.S.: Discrete-Time Signal Processing. second edn. Prentice-Hall (1999)
8. Golub, G.H., Loan, C.F.V.: Matrix computations. third edn. Johns Hopkins University Press (1996)