Recursive Filtering of Images with Symmetric Extension

Ben Appleton*

School of Information Technology and Electrical Engineering

The University of Queensland, Brisbane, QLD 4072, Australia

appleton@itee.uq.edu.au

Hugues Talbot

CSIRO Mathematical and Information Sciences,

Locked Bag 17, North Ryde, NSW 1670, Australia

hugues.talbot@csiro.au

9th February 2005

Abstract

Recursive filters are widely used in image analysis due to their efficiency and simple implementation. However these filters have an

*Corresponding author. Tel.: +61 7 3365 4510; Fax.: +61 7 3365 4999

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 symmetrically extended images for filters with symmetric denominator. We begin with an analysis of symmetric extensions and their effect on non-recursive filtering operators. Based on the non-recursive case, we derive a formulation of recursive filtering on symmetric domains as a linear but spatially-varying implicit operator. We then give an efficient method for decomposing and solving the linear implicit system, along with a proof that this decomposition always exists.

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 less computation than the standard recursive filtering. We give experimental evidence to verify these claims.

Keywords

Image processing, recursive filtering, symmetric extension, Neumann boundary conditions

1 Introduction

Recursive filters are widely used in image analysis due to their efficiency and simple implementation. They replace the value of each pixel in an image by the weighted combination of a large neighbourhood of pixels. However this causes problems near the boundary of the image, as the output of the filter may depend upon the unknown values of pixels which lie outside of the image. Existing solutions to this problem produce either unusable results near the image boundaries or require costly, approximate solutions such as manually extending the image beyond the boundary before filtering. These problems are most significant on filters with small bandwidth, such as those encountered in linear scale space construction [1].

Figure 1 demonstrates the process of blurring an image with identical filters using two different boundary conditions. The first boundary condition is zero extension, which treats the pixels outside the image as having zero value. This produces dark results near the boundary of the image which may cause problems in later processing. The second boundary condition is symmetric extension, which treats the borders of the image as mirrors. In this case the results near the image boundary are quite sensible. While it is impossible to know what lies outside of the image, a simple but effective image extension method such as symmetric extension allows us to obtain useful results.

The Fast Fourier Transform is commonly used to efficiently implement

(a) (b) (c)

Figure 1: The effect of border extension on image filtering. (a) A piece of jewellery and a coin. (b) Blurring of (a) with zero extension at the image boundaries. (c) Blurring of (a) with symmetric extension at the image boundaries.

linear filtering. It assumes that the image domain is periodic, which unfortunately causes significant filtering artifacts near the borders of the image. Martucci addressed this problem by considering the symmetric extension of the image borders [2]. He investigated the efficient implementation of *symmetric convolution* using the various Discrete Trigonometric Transforms (DTTs). Martucci demonstrated that DTTs could be used for an efficient implementation of the linear filtering of finite signals with symmetric (reflective) boundary conditions. However as Deriche observed in [1], methods based on the Fast Fourier Transform take significantly greater computational effort than a direct implementation for recursive filters of low order.

Cunha [3] observed that a reflectively extended signal can be treated as a periodic signal with twice the length. This work considered the approach 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. In a related approach, Smith and Eddins [4] suggest explicitly computing the impulse response of the recursive filter and using this to compute the initial values. Hoever as presented this solution is restricted to filters that have only single poles. Both of these methods suffer from the additional disadvantage that they must be repeated for each row of the image being filtered, leading to inefficient filtering implementations.

In [5] Weickert, ter Haar Romeny and Viergever give a method for recursive Gaussian filtering on an image with symmetric extension of the borders. This method is derived from the relationship between linear diffusion filtering and Gaussian convolution, using a first order semi-implicit approximation to a linear diffusion partial differential equation on a domain with Neumann boundary conditions. Unfortunately their method requires a number of iterations proportional to the square of the scale of the Gaussian.

Appleton and Talbot [6] have presented a method for recursive filtering of finite images for a wide class of recursive filters. This is the class of filters which are dimensionally separable and whose recursive component is symmetric. This method computes the correct result up to numerical precision of recursive filtering on a symmetrically extended domain and requires fewer operations than existing recursive filters. However it relied upon a decomposition which was not proven to exist for the class of filters considered.

In this paper, we extend the work presented in [6] for recursive filtering on symmetrically extended images. In Section II we recall discrete filtering on infinite and finite domains. Section III investigates the properties of filtering

on symmetrically extended domains and consequently proposes an efficient algorithm. Section IV describes the application of the proposed algorithm to a filter commonly used in image analysis. Section V gives results on accuracy and timings.

2 Discrete Filtering

2.1 Filtering on infinite domains

Here we briefly review non-recursive and recursive filtering, focussing in particular on the formulation of linear filtering as a system of explicit or implicit linear equations. For a more detailed presentation of linear filtering we refer the reader to [7].

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

$$y[i] = (h \star x)[i] = \sum_{j=-\infty}^{\infty} h[j]x[i-j] \tag{1}$$

where h is known as the *kernel* of the filter. Here we assume that the operator $h\star$ is stable in the sense that all bounded inputs produce a bounded output.

We may express this relationship as the (infinite) matrix-vector product

$$y = Hx \tag{2}$$

Here $H_{ij} = h[i-j]$. For a filter h of length b, H is a banded Toeplitz matrix with total band width b.

Recursive filters, also known as autoregressive or infinite impulse response filters, may be 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

$$x = h \star y \tag{3}$$

to obtain y. As in the non-recursive case, we may implicitly define recursive filtering in matrix notation

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

where we solve the implicit system to obtain y. A low order recursive filter is usually sufficient to approximate a desired impulse response, producing a narrowly banded matrix H [7].

Unfortunately the term 'recursive filter' is also used to describe the more general combination of a non-recursive filter and a recursive filter. As such filters are sequentially separable, we will treat them separately in this paper.

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 then induce the definition of filtering from the choice of extension.

2.2.1 Periodic extension

One 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 2. For the filter

$$h = (\ldots,\ h_{-2},\ h_{-1},\ h_0,\ h_1,\ h_2,\ \ldots) \tag{5}$$

we obtain the $N \times N$ circulant filter matrix

$$H_\star = \begin{bmatrix} h_0 & h_{-1} & h_{-2} & \ldots & h_2 & h_1 \\ h_1 & h_0 & h_{-1} & \ldots & h_3 & h_2 \\ h_2 & h_1 & h_0 & \ldots & h_4 & h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{-2} & h_{-3} & h_{-4} & \ldots & h_0 & h_{-1} \\ h_{-1} & h_{-2} & h_{-3} & \ldots & h_1 & h_0 \end{bmatrix} \tag{6}$$

Here for clarity of expression it has been assumed that the filter kernel h is shorter than the domain length N.

2.2.2 Symmetric extension

Periodic extension can have undesirable effects near the image border, introducing a large discontinuity when connecting the two endpoints of the signal. As a more appropriate alternative, the symmetric extension of a signal ensures continuity at the boundary. In the case of linear scale space construction this has been observed to be equivalent to imposing an adiabatic boundary on the related diffusion process, maintaining the total brightness of an image [8]. In the case of discrete filtering this is equivalent to symmetric extension of the original image with repetition of the endpoints, known in the context of the Discrete Cosine Transforms as type-II symmetric extension [7].

Following [6] we may express reflective filtering in matrix notation using Equation 2 with the filter matrix

$$H_{\odot} = \frac{1}{2}\Phi^T H_{\star}\Phi \tag{7}$$

Here $H_{\odot}$ is the $N \times N$ reflective filtering matrix. $H_{\star}$ is the $2N \times 2N$ periodic filtering matrix corresponding to filtering by h in the reflectively extended domain $[1, 2N]$. The construction of this matrix follows Section 2.2.1. Φ is

the canonical reflective extension matrix

$$\Phi = \begin{bmatrix} I_N \\ J_N \end{bmatrix} \tag{8}$$

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

$$J_N = \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix} \tag{9}$$

$H_\odot$ is simply the combination of:

1. A reflective extension of the signal from the original domain $[1, N]$ to the domain $[1, 2N]$ such that it becomes periodic
2. A periodic convolution in the extended domain
3. A projection back to the original signal domain

It may also be noted that the projection $\frac{1}{2}\Phi^T$ in Equation 7 is effectively the same as filtering the symmetrically extended image in the causal direction as well as the anticausal direction before averaging the results.

2.3 Singular value decompositions of filtering operators

The singular value decompositions of the periodic filtering matrix $H_\star$ and the reflective filtering matrix $H_\odot$ have been extensively researched [9, 10]. Here we restate the key observations:

2.3.1 $H_\star$ and the Discrete Fourier Transform

The circulant matrix $H_\star$ corresponding to periodic filtering may be diagonalised by the Discrete Fourier Transform (DFT):

$$H_\star = F^* \Lambda F \tag{10}$$

Here F is the DFT matrix in orthonormal form:

$$F_{kn} = \frac{1}{\sqrt{N}} W_N^{kn} \tag{11}$$

where $W_N = e^{-j2\pi/N}$. Λ is then the diagonal matrix whose entries Λ_{ii} are the DFT coefficients of $h\star$. Note that F^* denotes the Hermitian or conjugate transpose of F.

2.3.2 $H_\odot$ and the Discrete Cosine Transform

The matrix $H_\odot$ corresponding to filtering with symmetric extension may be diagonalised by the Discrete Cosine Transform (DCT):

$$H_\odot = C^T \Gamma C \tag{12}$$

Here C is the DCT matrix given in orthonormal form by Ng, Chan and Tang [10]. Γ is the diagonal matrix whose entries Γ_{ii} are the corresponding DCT coefficients of $h\odot$.

Both the DFT and the DCT may be computed in $O(N \log N)$ time, allowing for rapid inversion of the matrices $H_\star$ and $H_\odot$. [7] presents the well known algorithm for computing a periodic convolution via the DFT, while [10] presents an algorithm for computing a reflective convolution via the DCT. Here we restate the DCT algorithm for completeness:

Algorithm 1 (Filtering via the DCT). *To solve Equation 2 or 4:*

1. *Compute the DCT of the filter $h\odot$*
2. *Compute the DCT of the image x*
3. *Pointwise multiply (2) or divide (4) the DCT coefficients of x by those of $h\odot$*
4. *Compute the inverse DCT of the result*

However as noted by Deriche in [11], methods based on the Fourier transform are significantly slower than a direct implementation of recursive filtering in the usual case of a kernel with narrow support. Unfortunately the faster alternative, the standard implementation of a recursive filter with infinite impulse response, is necessarily approximate on a finite domain.

3 Recursive Filtering in a Symmetrically Extended Domain

In this section we examine the properties of the matrix $H_\odot$ corresponding to filtering in a symmetrically extended domain, and present a number of new results. We demonstrate that the solution of the linear system described by $H_\odot$ always exists and that its computation is numerically stable. Furthermore we prove that $H_\odot$ is a symmetric positive definite matrix, and give a direct decomposition which may be used to solve the linear system. Finally, we propose a novel recursive filtering algorithm which is both theoretically consistent with the symmetric extension of the finite domain and faster than existing methods.

3.1 Equivalence to a symmetric filter

We observed in [6] that $H_\odot$ is symmetric and that the anti-symmetric component of the kernel h makes no contribution. This has also been described

in [10] in the context of filtering by the type-2 Discrete Cosine Transform, one of the Discrete Trigonometric Transforms (DTTs). Therefore all reflective 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 and hence that $H_\star$ is a symmetric matrix.

It is important to note that in practice this symmetry constraint applies only to the recursive component of the filter, and does not prevent the use of filters with asymmetric (or anti-symmetric) non-recursive components. Examples will be given of the application of this theory to anti-symmetric filters such as computing the first derivative.

3.2 Existence and stability

$H_\star$ is an invertible matrix if and only if $h\star$ is an invertible filter. Observe then that if the symmetric filter $h\star$ is invertible then so too is the filter matrix $H_\odot$, with its inverse given by [6]:

$$H_\odot^{-1} = \frac{1}{2}\Phi^T H_\star^{-1}\Phi \tag{13}$$

We analyse the reflective filtering matrix $H_\odot$ in terms of the periodic filtering matrix $H_\star$ using the condition number corresponding to the 2-norm. The condition number gives a measure of the numerical error which is ex-

pected to be introduced when solving a matrix equation [9].

$$\begin{aligned}\kappa_2(H_\odot) &= \frac{1}{4}\left\|\Phi^T H_\star \Phi\right\|_2 \left\|\Phi^T H_\star^{-1}\Phi\right\|_2 \\ &\leq \frac{1}{4}\|\Phi^T\|_2\|H_\star\|_2\|\Phi\|_2\|\Phi^T\|_2\|H_\star^{-1}\|_2\|\Phi\|_2 \\ &= \frac{1}{2}\|H_\star\|_2\|H_\star^{-1}\|_2 \\ &= \frac{1}{2}\kappa_2(H_\star) \end{aligned} \tag{14}$$

Here we have noted that $\|\Phi\|_2^2 = 2$ and $\|\Phi^T\|_2^2 = 1$. Therefore recursive filtering by the matrix $H_\odot$ in the symmetrically extended domain $V(R)$ is more accurate than recursive filtering by the matrix $H_\star$ in the periodic domain $V(\mathbb{Z}_{2N})$.

3.3 $H_\odot$ is symmetric positive definite

Theorem 1. *For a real, symmetric filter kernel h which is both finite and invertible, the derived matrix $H_\star$ is symmetric positive definite or symmetric negative definite.*

Proof. A symmetric positive definite matrix has all eigenvalues real and positive. The eigenvalues of $H_\star$ are the values of the Z-transform of h:

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n}, \tag{15}$$

evaluated at the N roots of unity $z = W_N^k$ on the unit circle $H(e^{j\omega})$. Therefore

it suffices to demonstrate that the Z-transform of h is real and uniformly positive or negative on the unit circle.

As h is real and symmetric, $H(z)$ is real. As $h\star$ is invertible, $H(z)$ has no zeros on the unit circle. For a finite impulse response h, $H(z)$ is differentiable everywhere. So the periodic function $H(e^{j\omega})$ is real, smooth and does not pass through zero. Therefore it is uniformly positive or uniformly negative, and hence the matrix $H_\star$ is symmetric positive definite or symmetric negative definite. Up to multiplication by -1 then, $H_\star$ is symmetric positive definite. □

Note that this is equivalent to saying that, if the symmetric filter kernel h has finite length and is invertible, then it is the autocorrelation function of a filter with real coefficients.

Corollary 1. *For a real and symmetric filter kernel h which is both finite and invertible, the derived matrix $H_\odot$ is symmetric positive definite up to multiplication by -1.*

Proof. We recall the definition $H_\odot = \frac{1}{2}\Phi^T H_\star \Phi$ (Equation 7). $H_\star$ is a symmetric positive definite matrix up to scaling, and Φ has full column rank. Therefore $H_\odot$ is also symmetric positive definite up to multiplication by -1. □

3.4 LDL^T decomposition and solution

Let $A \in \mathbb{R}^{N \times N}$ be a symmetric, positive definite matrix with real elements. Then A may be decomposed uniquely 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 [9]. For an $N \times N$ matrix with total band width b it may be performed in $O(Nb^2)$ time and $O(Nb)$ space.

The LDL^T decomposition of a symmetric positive definite matrix is equivalent to the well known Cholesky decomposition $GG^T = A$ with $G = LD^{1/2}$. However the LDL^T decomposition has two practical benefits: it may computed without performing any square roots, and its inversion requires N fewer divisions. Both the Cholesky decomposition and the LDL^T decomposition have the property that a banded matrix will decompose into banded matrices. This leads to very efficient implementations in practice.

Once the LDL^T decomposition has been computed it is simple to solve Equation 4 to recursively filter a signal x and obtain the filtered signal y. In sequence solve:

$$x = Lu \tag{16}$$

$$u = Dv \tag{17}$$

$$v = L^T y \tag{18}$$

The solution to Equations 16, 17, and 18 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 direct implementation of recursive filtering on an infinite domain.

3.5 A recursive filtering algorithm for reflective domains

Here we give an algorithm for the recursive filtering of images with symmetric boundary conditions.

Algorithm 2 (Recursive filtering with symmetric extension). *For each image axis:*

1. *Form the $H_\odot$ matrix along the current axis according to Equation 7*
2. *Compute the decomposition $LDL^T = H_\odot$, which always exists according to Corollary 1.*
3. *For each row of data x, solve in place Equations 16, 17, 18.*

The decomposition only needs to be 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 $d \geq 2$ 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.

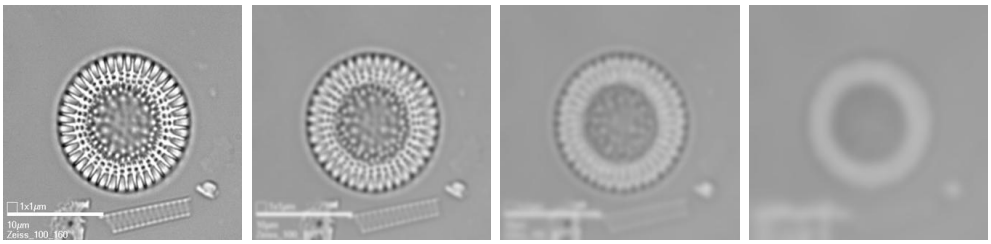
4 Application to Deriche's Recursive Gaussian Approximations

Here we consider Deriche's approximations to filtering by Gaussians and their derivatives, which in theory require an amount of computation which is independent of the scale of the Gaussian [11]. 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 2 demonstrates the application of Deriche's 4th order Gaussian approximation to a microscope image of a diatom, while Figure 3 demonstrates the application of Deriche's 4th order approximation to the derivative of a Gaussian to computing its spatial gradient. The application of the new scheme to other filters is straightforward.

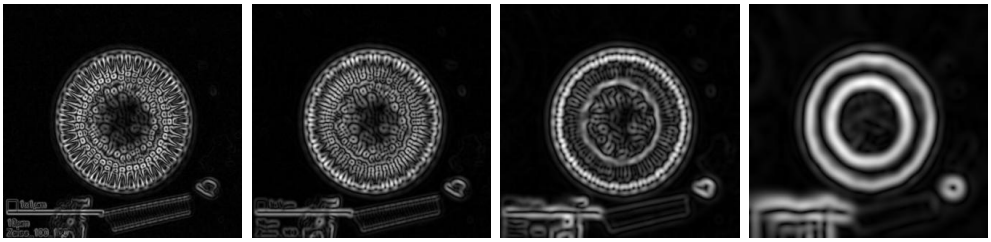
Deriche's 4th order approximations are of the form

$$h_a(x) \approx \left(\alpha_0 \cos(w_0 \frac{x}{\sigma}) + \beta_0 \sin(w_0 \frac{x}{\sigma})\right) \exp(\gamma_0 \frac{x}{\sigma}) + \left(\alpha_1 \cos(w_1 \frac{x}{\sigma}) + \beta_1 \sin(w_1 \frac{x}{\sigma})\right) \exp(\gamma_1 \frac{x}{\sigma})$$



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

Figure 2: 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)

Figure 3: Anti-symmetric filtering: the gradient magnitudes of *Cyclostephanos Dubius*. (a) $\sigma = 1$. (b) $\sigma = 2$. (c) $\sigma = 4$. (d) $\sigma = 8$.

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

$$\begin{array}{llll} \alpha_0 = & 1.680 & \alpha_1 = & \text{-}0.6803 \\ \beta_0 = & 3.735 & \beta_1 = & \text{-}0.2598 \\ \gamma_0 = & \text{-}1.783 & \gamma_1 = & \text{-}1.783 \\ w_0 = & 0.6318 & w_1 = & 1.9970 \end{array}$$

Similarly, Deriche's 4th order approximation to the first derivative of a

Gaussian is given by:

$$\begin{array}{llll} \alpha_0 = & \text{-0.6472} & \alpha_1 = & 0.6494 \\ \beta_0 = & \text{-4.531} & \beta_1 = & 0.9557 \\ \gamma_0 = & \text{-1.527} & \gamma_1 = & \text{-1.516} \\ w_0 = & 0.6719 & w_1 = & 2.072 \end{array}$$

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:

$$H_+(z^{-1}) = \frac{n_{00}^+ + n_{11}^+ z^{-1} + n_{22}^+ z^{-2} + n_{33}^+ z^{-3}}{1 + d_{11} z^{-1} + d_{22} z^{-2} + d_{33} z^{-3} + d_{44} z^{-4}} \tag{19}$$

while the anti-causal filter is of the form:

$$H_-(z) = \frac{n_{11}^- z + n_{22}^- z^2 + n_{33}^- z^3 + n_{44}^- z^4}{1 + d_{11} z + d_{22} z^2 + d_{33} z^3 + d_{44} z^4} \tag{20}$$

These may be combined to produce a symmetric filter

$$\begin{aligned} H(z^{-1}) &= H_+(z^{-1}) + H_-(z) \\ &= \frac{n_0 + \sum_{k=1}^{3} n_k (z^{-k} + z^k)}{d_0 + \sum_{k=1}^{4} d_k (z^{-k} + z^k)} \end{aligned}$$

or an anti-symmetric filter

$$\begin{aligned} H(z^{-1}) &= H_+(z^{-1}) - H_+(z) \\ &= \frac{\sum_{k=1}^{4} n_k(z^{-k} - z^k)}{d_0 + \sum_{k=1}^{4} d_k(z^{-k} + z^k)} \end{aligned}$$

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 need not be 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 filter are applied 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.

5 Results

All tests were performed on a 700MHz Toshiba P-III laptop with 192MB of RAM under the Linux operating system. Both the algorithm presented here and the reference implementation have been implemented in double precision floating point arithmetic in the C language and compiled using the GNU C compiler with standard optimisation flags. Where results are presented for the DCT algorithm of [10] the FFTW library has been used [12].

5.1 Timing

Here we compare the running time of the algorithm proposed in this paper with the standard implementation via border extension and with the DCT algorithm. Borders are manually extended by 4σ in the standard implementation as a reasonable tradeoff between additional computation and border effects. It is not necessary to extend the borders in either the proposed algorithm or the DCT algorithm. We consider a range of image sizes and scales for 2D and 3D images. Image sizes are powers of two in order to show the DCT algorithm at its best. Although for simplicity we have chosen to test square and cubic images the filter decomposition required for the method proposed in this paper has been repeated for each image axis, *i.e.* twice for 2D images and three times for 3D images.

Timings for 2D images are given in Tables 1, 2, 3 while timings for 3D images are given in Tables 4, 5, 6. Figure 4 compares the algorithm proposed

in this paper to the standard implementation over a range of scales σ for several image sizes in 2D and 3D. Figure 5 compares the algorithm proposed in this paper to the DCT algorithm over a range of image sizes in 2D and 3D.

We observe that the method proposed in this paper is faster than the standard implementation. It requires constant computation irrespective of scale σ as there is no need to extend the image borders, whereas the standard implementation requires more computation for larger scales. The new method is also approximately 4 times faster than the DCT algorithm on images whose sidelengths are powers of 2. This suggests that the new method is faster than the DCT algorithm for filters of order up to 16 over the range of image sizes considered. For image sizes which are not a power of 2 this margin increases substantially.

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

$\sigma =$	10	20	50	100
Sidelength				
128	11	16	19	30
256	50	53	62	74
512	190	191	207	243
1024	870	876	911	982

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

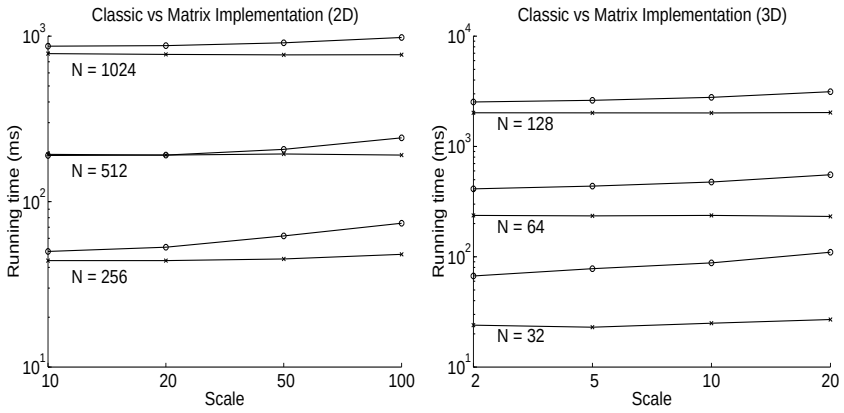
$\sigma =$	10	20	50	100
Sidelength				
128	10	7	8	9
256	44	44	45	48
512	193	191	194	191
1024	784	776	771	772

Table 3: Running times (ms) for the DCT algorithm.

$\sigma =$	10	20	50	100
Sidelength				
128	38	38	37	33
256	170	165	162	163
512	896	894	890	890
1024	3278	3251	3246	3257

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

$\sigma =$	2	5	10	20
Sidelength				
32	67	78	88	110
64	413	437	476	554
128	2531	2622	2784	3138



(a) (b)

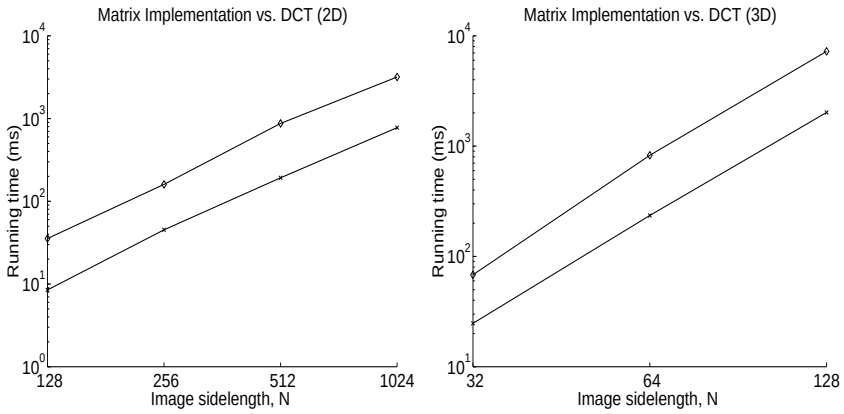
Figure 4: Graphs comparing the running times of the algorithm presented here (crosses) to the standard implementation of recursive filtering (circles). (a) 2D images. (b) 3D images.

Table 5: Running times (ms) for the algorithm presented here on 3D images.

$\sigma =$	2	5	10	20
Sidelength				
32	24	23	25	27
64	237	235	237	232
128	2019	2016	2013	2030

Table 6: Running times (ms) for the DCT algorithm.

$\sigma =$	2	5	10	20
Sidelength				
32	72	71	71	67
64	840	840	845	855
128	7139	7176	7227	7973



(a) (b)

Figure 5: Graphs comparing the running times of the algorithm presented here (crosses) to reflective filtering using the DCT (diamonds). (a) 2D images. (b) 3D images.

6 Conclusion

In this paper we have described a new method for the recursive filtering of symmetrically extended images by filters with symmetric denominator. Unlike the standard implementation of recursive filtering this method is simultaneously efficient and theoretically exact. It is based on a spatially-varying formulation of convolution previously derived in [6] by considering the algebra of reflective extensions. In the recursive case this leads to an implicit linear system whose solution was proven to exist and have improved numerical stability compared to the original filter in a periodic 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 has been applied to both symmetric and anti-symmetric filters. It has been demonstrated on Deriche's recursive filters approximating Gaussians and their first derivatives. Results demonstrate that the proposed method has excellent numerical accuracy. In contrast to standard convolution-based implementations it requires a constant amount of computation irrespective of scale. It is faster than the standard implementation of recursive filtering and substantially faster than implementations based on the FFT, even for images whose sidelengths are powers of 2.

Acknowledgements

The diatom image in Figures 2 and 3 were taken from the ADIAC public data web page:
http://www.ualg.pt/adiac/pubdat/pubdat.html
(CEC contract MAS3-CT97-0122).

The authors would like to acknowledge the anonymous reviewers for their helpful comments which improved this paper.

References

[1] R. Deriche, Fast algorithms for low-level vision, IEEE Tr. on Pattern Analysis and Machine Intelligence 12 (1) (1990) 78–87.

[2] S. A. Martucci, Symmetric convolution and the discrete sine and cosine transforms, IEEE Transactions on Signal Processing 42 (5) (1994) 1038–1051.

[3] A. M. da Cunha, Espaços de escala e detecção de arestas, Master's thesis, IMPA, Rio de Janeiro, http://www.visgraf.impa.br/RefBib/Data/PS_PDF/student-msc-2000-10-anderson-cunha/thesis.pdf (October 2000).

[4] M. J. T. Smith, S. L. Eddins, Analysis/synthesis techniques for subband image coding, IEEE Transactions on Acoustics, Speech, and Signal Processing 38 (8) (1990) 1446–1456.

[5] J. Weickert, B. M. ter Haar Romeny, M. A. Viergever, Efficient and reliable schemes for nonlinear diffusion filtering, IEEE Transactions on Image Processing 7 (3) (1998) 398–410.

[6] B. Appleton, H. Talbot, Efficient and consistent recursive filtering of images with reflective extension, in: L. Griffin, M. Lillholm (Eds.), Proceedings of the IVth International Conference on Scale-Space theories in Computer Vision, British Machine Vision Association, Springer-Verlag, 2003, pp. 699–712.

[7] A. V. Oppenheim, R. W. Schafer, J. R. Buck, Discrete-Time Signal Processing, 2nd Edition, Prentice-Hall, 1999.

[8] T. Lindeberg, Scale-space for discrete signals, IEEE Transaction on Pattern Analysis and Machine Intelligence 12 (3) (1990) 234–254.

[9] G. H. Golub, C. F. V. Loan, Matrix computations, 3rd Edition, Johns Hopkins University Press, 1996.

[10] M. K. Ng, R. H. Chan, W.-C. Tang, A fast algorithm for deblurring models with Neumann boundary conditions, SIAM Journal of Scientific Computing 21 (3) (1999) 851–866.

[11] R. Deriche, Recursively implementing the gaussian and its derivatives, in: Proc. Second Intern. Conf. on Image Processing, Singapore, 1992, pp. 263–267.

[12] M. Frigo, S. G. Johnson, FFTW: An adaptive software architecture for the fft, in: Proc. ICASSP 1998, Vol. 3, 1998, pp. 1381–1384.

Biographical Sketch

BEN APPLETON is currently undertaking a PhD in the area of Image Analysis at the University of Queensland, Brisbane, Australia. He received his bachelor degrees in Science and Electrical Engineering from the University of Queensland in 2001 and was awarded a university medal. His research interests include active contours and energy minimisation algorithms.

HUGUES TALBOT received the engineering degree from École Centrale de Paris in 1989, the D.E.A. (Masters) from University Paris VI in 1990 and the Ph.D from École des Mines de Paris in 1993., under the guidance of Dominique Jeulin and Jean Serra. He has been affiliated with the Commonwealth Scientific and Industrial Research Organisation (CSIRO), Mathematical and Information Sciences since 1994. He has worked on numerous applied projects in relation with industry, he has contributed more than 30 research papers in international journals and conferences and he has co-edited two sets of international conference proceedings on image analysis. He has also taught image processing at the University of Sydney, and his research interests include image segmentation, linear structure analysis, texture analysis and algorithms.