Fast Anisotropic Gauss Filtering

Geusebroek, J.M.; Smeulders, A.W.M.; van de Weijer, J.

Published in:
7th European Conference on Computer Vision (ECCV)

Fast Anisotropic Gauss Filtering

Jan-Mark Geusebroek*, Arnold W. M. Smeulders, and Joost van de Weijer

Intelligent Sensory Information Systems, Department of Computer Science,
University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands;
geusebroek@science.uva.nl

Abstract. We derive the decomposition of the anisotropic Gaussian in a one dimensional Gauss filter in the x-direction followed by a one dimensional filter in a non-orthogonal direction φ. So also the anisotropic Gaussian can be decomposed by dimension. This appears to be extremely efficient from a computing perspective. An implementation scheme for normal convolution and for recursive filtering is proposed. Also directed derivative filters are demonstrated.
For the recursive implementation, filtering an 512×512 image is performed within 65 msec, independent of the standard deviations and orientation of the filter. Accuracy of the filters is still reasonable when compared to truncation error or recursive approximation error.
The anisotropic Gaussian filtering method allows fast calculation of edge and ridge maps, with high spatial and angular accuracy. For tracking applications, the normal anisotropic convolution scheme is more advantageous, with applications in the detection of dashed lines in engineering drawings. The recursive implementation is more attractive in feature detection applications, for instance in affine invariant edge and ridge detection in computer vision. The proposed computational filtering method enables the practical applicability of orientation scale-space analysis.

1 Introduction

One of the most fundamental tasks in computer vision is the detection of edges and lines in images. The detection of these directional structures is often based on the local differential structure of the image. Canny's edge detector examines the magnitude of the first order image derivatives [3]. A well-founded approach for line detection is given by [16, 20], where line structures are detected by examining the eigenvectors of the Hessian matrix, the Hessian being given by the local second order derivatives. Robust measurement of image derivatives is obtained by convolution with Gaussian derivative filters, a well known result from scale-space theory [6, 13, 15, 21].

The difficulty of edge and line detection is emphasized when the structures run close together or cross each other, as is the case in engineering drawings or two-dimensional projections of complex (3D) scenes. In these cases, isotropic filtering strategies as used in e.g. [2, 3, 8, 12, 20] are not sufficient. Isotropic smoothing causes parallel lines to be blurred into one single line. Crossing lines are not

* This work was supported by the ICES Multimedia Information Analysis Project

well detected by isotropic filters [17], due to the marginal orientation selectivity of the Gaussian filter.

The task of edge and line detection is considered not trivial when disturbing influences have to be ignored, like shadows and shading, gaps in dashed lines, object borders locally blending together, partly occlusion of the object, or clutter in the background. In these cases, one would often like to have a detection method which ignores the distorting data aside the edge or line, while accumulating evidence of the edge or line data along its orientation. Hence, taking advantage of the anisotropic nature of lines and edges [1, 10].

The use of isotropic Gaussians is for historical reasons, imposed by simplicity of derivation and efficiency in computation [13]. The assumption of isotropy for front-end vision [6, 13, 15, 21] does not imply the scale-space operator to be isotropic, rather imposes the complete sampling of all possible orientations of the scene. The notion of orientation sampling suggest a combined scale and orientation space [12, 14, 22]. For a linear orientation scale-space, the anisotropic Gaussian is the best suited causal filter. We propose a method for fast filtering by anisotropic Gaussian's to construct an orientation scale-space.

Orientation analysis is often approached by steerable filters. Freeman and Adelson [7] put forward the conditions under which a filter can be tuned to a specific orientation by making a linear combination of basis filters. Their analysis included orientation tuning of the xy-separable first order isotropic Gaussian derivative filter. According to their framework, no exact basis exists for rotating an anisotropic Gaussian. Van Ginkel *et al.* proposed a deconvolution scheme for improving the angular resolution of the Gaussian isotropic filter. Under a linearity assumption on the input image, a steerable filter with good angular resolution is obtained. The method involves a Fourier based deconvolution technique, which is of high computational complexity. Perona [17] derived a scheme for generating a finite basis which approximates an anisotropic Gaussian. The scheme allowed both steering and scaling of the anisotropic Gaussian. However, the number of basis filters is large, and the basis filters are non-separable, requiring high computational performance.

We show the anisotropic Gaussian filter to be separable along two directions, not necessarily orthogonal. One of the axes is in a fixed, but in a freely selectable direction. The latter axis depends on the filter parameters $(\sigma_u, \sigma_v, \theta)$, the standard deviations and orientation, respectively. We show the resulting one-dimensional filters to be Gaussian filters. Hence, fast algorithms [4, 5, 23, 24] can be used to calculate the orientation smoothed image and its derivatives. Van Vliet and Young propose [23, 24] a recursive implementation of one-dimensional Gaussian filters. This is a great advantage over [5], as combining the one-dimensional filters into two-dimensional Gaussians will not introduce bias along the filter directions. The recursive filters need only 7 multiplications and 6 additions per pixel, independent of the standard deviation σ of the Gaussian filter. Moreover, the filter coefficients are simple to calculate.

In this paper, we show the decomposition of the anisotropic Gaussian in two Gaussian line filters in non orthogonal directions (Sect. 2). Choosing the

x-axis to decompose the filter along turns out to be extremely efficient from a computing perspective. We combine the decomposition with the recursive algorithms proposed in [23, 24], yielding a constant calculation time with respect to the Gaussian scales and orientation (Sect. 3). We give timing results and compare accuracy with two-dimensional convolution in Sect. 4. Implementation of Gaussian derivative filters is given in Sect. 5.

2 Separation of Anisotropic Gaussian

A simple case of the isotropic Gaussian convolution filter in two dimensions is given by

$$g_\circ(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left\{-\frac{1}{2}\left(\frac{x^2+y^2}{\sigma^2}\right)\right\} \quad . \tag{1}$$

Anisotropy is obtained when scaling differently in the x- and y-direction. Then, an elliptic Gaussian with axes aligned along the coordinate system (see Fig. 1a) is given by

$$g_\perp(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left\{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)\right\} \quad . \tag{2}$$

Rotation of the coordinate system (x, y) over θ,

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{3}$$

results in the general case of oriented anisotropic Gaussian (Fig. 1b),

$$g_\theta(x, y; \sigma_u, \sigma_v, \theta) = \\ \frac{1}{2\pi\sigma_u\sigma_v} \exp\left\{-\frac{1}{2}\left(\frac{(x\cos\theta + y\sin\theta)^2}{\sigma_u^2}+\frac{(-x\sin\theta + y\cos\theta)^2}{\sigma_v^2}\right)\right\} \tag{4}$$

the u-axis being in the direction of θ, and the v-axis being orthogonal to θ.

From standard Fourier theory we have the convolution theorem,

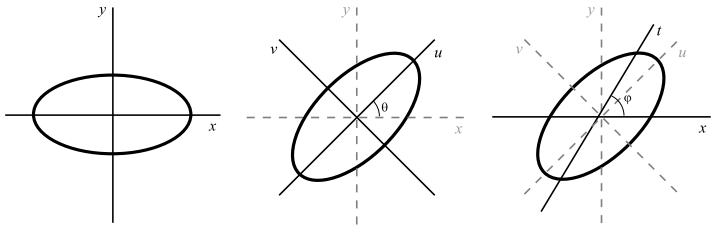
$$f(x, y) * h(x, y) \quad \overset{\mathcal{F}}{\Leftrightarrow} \quad F(\omega_x, \omega_y) H(\omega_x, \omega_y) \quad . \tag{5}$$

A linear filter is separable into two subsequent convolutions iff its Fourier transform can be written as a multiplication of two functions, one depending on ω_x, the other depending on ω_y,

$$h(x, y) = h_x(x) * h_y(y) \quad \overset{\mathcal{F}}{\Leftrightarrow} \quad H(\omega_x, \omega_y) = H_{\omega_x}(\omega_x) H_{\omega_y}(\omega_y) \quad . \tag{6}$$

The Fourier transform of $g_\theta(.)$ (Eq. (4)) is given by

$$G_\theta(\omega_x, \omega_y; \sigma_u, \sigma_v, \theta) = \\ \exp\left\{-\frac{1}{2}\left((\omega_x\cos\theta + \omega_y\sin\theta)^2 \sigma_u^2 + (-\omega_x\sin\theta + \omega_y\cos\theta)^2 \sigma_v^2\right)\right\} \quad . \tag{7}$$



(a) (b) (c)

Fig. 1. Ellipse and its axes systems. An example of an anisotropic Gaussian with aspect ratio 1:2 and orientation $\theta = \frac{\pi}{4}$. **a.** Cartesian xy-aligned Gaussian. **b.** Principal axes uv-aligned Gaussian. **c.** uv-aligned Gaussian in a non-orthogonal xt-axes system. Axis t is rotated over $\varphi \approx \frac{\pi}{6}$ with respect to the x-axis.

The exponential is separable into two products if its argument is separable into sums ($e^{a+b} = e^a e^b$). From Eq. (7) it is easy to derive that the anisotropic Gaussian may be separated along the major axes u and v with (u, v) given by Eq. (3),

$$G_\theta(\omega_x, \omega_y; \sigma_u, \sigma_v, \theta) = \exp\left\{-\frac{1}{2}\left({\omega_u}^2\sigma_u^2 + {\omega_v}^2\sigma_v^2\right)\right\} \quad . \tag{8}$$

As we are interested in a convenient basis from a computational perspective, separation in u and v is uninteresting. What is needed is the decomposition into a filter in the x-direction and a filter along another direction. Therefore, we consider

$$\left(\omega_x \cos\theta + \omega_y \sin\theta\right)^2 \sigma_u^2 + \left(-\omega_x \sin\theta + \omega_y \cos\theta\right)^2 \sigma_v^2 \tag{9}$$

as the quadratic form in ω_x and ω_y,

$$a_{11}{\omega_x}^2 + 2a_{12}\omega_x\omega_y + a_{22}{\omega_y}^2 \tag{10}$$

where the coefficients are given by

$$\begin{aligned} a_{11} &= \sigma_u^2 \cos^2\theta + \sigma_v^2 \sin^2\theta \\ a_{12} &= \left(\sigma_u^2 - \sigma_v^2\right)\cos\theta\sin\theta \\ a_{22} &= \sigma_v^2 \cos^2\theta + \sigma_u^2 \sin^2\theta \quad . \end{aligned} \tag{11}$$

We aim at separating the anisotropic Gaussian filter into a filter in the x-direction, followed by a filter along a line $t : y = x\tan\varphi$,

$$\begin{aligned} f(x,y) * h(x,y) = f(x,y) * h_x(x) * h_\varphi(t) \quad &\overset{\mathcal{F}}{\Leftrightarrow} \\ F(\omega_x, \omega_y)H(\omega_x, \omega_y) &= F(\omega_x, \omega_y)H_x(\omega_x)H_\varphi(\omega_\varphi) \end{aligned} \tag{12}$$

To achieve our goal, we collect in Eq. (10) all terms dependent on ω_y and separate out all terms independent of ω_y. Hence, Eq. (10) may be rewritten as

$$\left(a_{11} - \frac{{a_{12}}^2}{a_{22}}\right){\omega_x}^2 + a_{22}\left(\omega_y + \frac{a_{12}}{a_{22}}\omega_x\right)^2 \quad . \tag{13}$$

Substitution in Eq. (7) yields

$$G_\theta(\omega_x, \omega_y; \sigma_u, \sigma_v, \theta) = \exp\left\{-\frac{1}{2}\left(\left(a_{11} - \frac{{a_{12}}^2}{a_{22}}\right){\omega_x}^2 + a_{22}\left(\omega_y + \frac{a_{12}}{a_{22}}\omega_x\right)^2\right)\right\} \quad . \tag{14}$$

Separation of the exponential sum results in

$$G_\theta(\omega_x, \omega_y; \sigma_u, \sigma_v, \theta) = \exp\left\{-\frac{1}{2}\left(a_{11} - \frac{{a_{12}}^2}{a_{22}}\right){\omega_x}^2\right\}\exp\left\{-\frac{1}{2}a_{22}\left(\omega_y + \frac{a_{12}}{a_{22}}\omega_x\right)^2\right\} \quad . \tag{15}$$

Back transformation to the spatial domain gives the separated anisotropic Gaussian filter,

$$g_\theta(x, y; \sigma_u, \sigma_v, \theta) = \frac{1}{2\pi\sigma_u\sigma_v}\exp\left\{-\frac{1}{2}\frac{x^2}{a_{11} - \frac{{a_{12}}^2}{a_{22}}}\right\} * \exp\left\{-\frac{1}{2}\frac{\left(y + \frac{a_{12}}{a_{22}}x\right)^2}{a_{22}}\right\} \quad . \tag{16}$$

The first factor represents a one-dimensional Gaussian in the x-direction at scale σ_x,

$$g_x(x; \sigma_x) = \frac{1}{\sqrt{2\pi\sigma_x}}\exp\left\{-\frac{1}{2}\frac{x^2}{\sigma_x^2}\right\} \tag{17}$$

where

$$\sigma_x = \sqrt{a_{11} - \frac{{a_{12}}^2}{a_{22}}} \quad . \tag{18}$$

The second factor represents a one-dimensional Gaussian along the line $t : y = x\tan\varphi$,

$$g_\varphi(r; \sigma_\varphi) = \frac{1}{\sqrt{2\pi\sigma_\varphi}}\exp\left\{-\frac{1}{2}\frac{r^2}{\sigma_\varphi^2}\right\} \tag{19}$$

where $r = \sqrt{x^2 + y^2}$ is the distance from the origin, and with direction tangent $\tan\varphi$ given by the total derivative of $y + \frac{a_{12}}{a_{22}}x$,

$$\tan\varphi = \frac{a_{22}}{a_{12}}$$

and standard deviation

$$\sigma_\varphi = \sqrt{a_{22}} \ . \tag{20}$$

Note that $\sqrt{2\pi\sigma_x}\sqrt{2\pi\sigma_\varphi} = 2\pi\sigma_u\sigma_v$, yielding the correct normalization (Eq. (16)). Rewriting Eq. (16) and substituting the quadratic coefficients Eq. (11) results in

$$g_\theta(x, y; \sigma_u, \sigma_v, \theta) = g_x(x; \sigma_x) * g_\varphi(\sqrt{x^2 + y^2}; \sigma_\varphi) \tag{21}$$

where

$$\begin{aligned} \tan\varphi &= \frac{\sigma_v^2 \cos^2\theta + \sigma_u^2 \sin^2\theta}{(\sigma_u^2 - \sigma_v^2)\cos\theta\sin\theta} \ , \\ \sigma_x &= \frac{\sigma_u\sigma_v}{\sqrt{\sigma_v^2 \cos^2\theta + \sigma_u^2 \sin^2\theta}} \ , \\ \sigma_\varphi &= \sqrt{\sigma_v^2 \cos^2\theta + \sigma_u^2 \sin^2\theta} \ . \end{aligned} \tag{22}$$

So we have achieved our goal namely that a Gauss filter at arbitrary orientation is decomposed into a one dimensional Gauss filter with standard deviation σ_x and another one dimensional Gauss filter at orientation φ and standard deviation σ_φ. For the anisotropic case $\sigma_u = \sigma_v = \sigma$, it is verified easily that $\sigma_x = \sigma$, $\sigma_\varphi = \sigma$, and $\tan\varphi = 0$. Further, for $\theta = 0$, trivially $\sigma_x = \sigma_u$, $\sigma_\varphi = \sigma_v$, and $\tan\varphi = 0$, and for $\theta = \frac{\pi}{2}$, $\sigma_x = \sigma_v$, $\sigma_\varphi = \sigma_u$, and $\tan\varphi = 0$.

An arbitrary example orientation of $\theta = \frac{\pi}{4}$ and $\sigma_v = \sigma, \sigma_u = 2\sigma$, results in $\sigma_x = \frac{2}{5}\sqrt{10}\sigma$, $\sigma_\varphi = \frac{1}{2}\sqrt{10}\sigma$, and $\tan\varphi = \frac{5}{3}$ ($\varphi \approx \frac{\pi}{3}$), see Fig. 1c.

3 Implementation

Implementation of Eq. (21) boils down to first applying a one dimensional Gaussian convolution in the x-direction. The resulting image is then convolved with a one-dimensional Gaussian in the φ-direction yielding the anisotropic smoothed image. The latter step implies interpolation, which can be achieved by linear interpolation between two neighboring x-pixels on the crossing between the image x-line of interest and the t-axis (see Fig. 1c). In this section, we consider two implementations of the anisotropic Gaussian, based on a common convolution operation, and based on a recursive filter [23], respectively.

Convolution Filter

Due to the filter symmetry, the x-filter Eq. (17) can be applied by adding pixel i left from the filter center with pixel i right from the filter center, and multiplying the summed pixels with filter weight i, or

$$g_x[x, y] = w_0 f[x, y] + \sum_{i=1}^{\lfloor N/2 \rfloor} w_i \left(f[x - i, y] + f[x + i, y] \right) \ . \tag{23}$$

Here, $f[x, y]$ is the input image, w_i is the filter kernel for half the sampled Gaussian from 0 to $\lfloor N/2 \rfloor$, and $g_x[x, y]$ is the filtered result image.

Filtering along the line $t : y = \mu x$, where $\mu = \tan \varphi$, is achieved by a sheared filter,

$$g_\theta[x, y] = w_0 g_x[x, y] + \sum_{i=1}^{\lfloor M/2 \rfloor} w_i \left(g_x[x - \mu i, y - i] + g_x[x + \mu i, y + i] \right) \quad . \tag{24}$$

Notice that the $y \pm i$ coordinate falls exactly on a line, whereas the $x \pm \mu i$ coordinate may fall between two pixels. Hence, the value of the source pixel may be obtained by interpolating between the pixels at the line of interest. To achieve our goal of fast anisotropic filtering, we consider linear interpolation between the neighboring pixels at $x \pm \mu i$ with interpolation coefficient a. The filter equation then becomes

$$\begin{aligned} g_\theta[x, y] = w_0 g_x[x, y] + \sum_{i=1}^{\lfloor M/2 \rfloor} w_i \{ & a \left(g_x[\lfloor x - \mu i \rfloor, y - i] + g_x[\lfloor x + \mu i \rfloor, y + i] \right) \\ & + (1 - a) \left(g_x[\lfloor x - \mu i \rfloor - 1, y - i] + g_x[\lfloor x + \mu i \rfloor + 1, y + i] \right) \} \quad . \end{aligned} \tag{25}$$

The multiplication of $w_i a$ and $w_i(1 - a)$ can be taken out of the loop to reduce the computational complexity of the filter.

Recursive Filter

Rather than applying convolution operators, Eq. (21) may be implemented by recursive filters. Van Vliet *et al.* [23, 24] define a scheme for one-dimensional Gaussian filtering with infinite support. The recursive filter requires only 7 multiplications per pixel, an improvement over [5]. The complexity is independent of the Gaussian standard deviation σ. In [24] it is shown that the recursive filter is faster than its normal counterpart for $\sigma > 1$. When using the recursive filter, filtering along the x-line is given by the forward and backward filter pair,

$$\begin{aligned} g_x^f[x, y] &= a_0 f[x, y] - a_1 g_x^f[x - 1, y] - a_2 g_x^f[x - 2, y] - a_3 g_x^f[x - 3, y] \\ g_x^b[x, y] &= a_0 g_x^f[x, y] - a_1 g_x^b[x + 1, y] - a_2 g_x^b[x + 2, y] - a_3 g_x^b[x + 3, y] \quad . \end{aligned} \tag{26}$$

Here, a_i represent the filter coefficients as given by [23, 24], and $g_x^b[x, y]$ is the x-filtered result image. The computational complexity of the recursive filter is 7 multiplications per pixel.

Filtering along the line $t : y = \mu x$, $\mu = \tan \varphi$, is achieved by a sheared recursive filter,

$$\begin{aligned} g_\theta^f[x, y] = g_\theta^f[t] &= a_0 g_x^b[x + \mu y, y] - a_1 g_\theta^f[t - 1] - a_2 g_\theta^f[t - 2] - a_3 g_\theta^f[t - 3] \\ g_\theta[x, y] = g_\theta^b[t] &= a_0 g_\theta^f[x + \mu y, y] - a_1 g_\theta^b[t - 1] - a_2 g_\theta^b[t - 2] - a_3 g_\theta^b[t - 3] \quad . \end{aligned} \tag{27}$$

Note that (x, y) are constraint to lie on the line t, hence may point to positions "between" pixels. Since interpolation of the recursive filter values is not possible, the filter history $g_\theta^f[t]$ and $g_\theta^b[t]$ has to be buffered, such that all t values are at the buffer "grid". The input values, $f[x, y]$ for the forward filter and $g_\theta^f[x, y]$ for the backward filter, are interpolated from the input data. The results $g_\theta^f[x, y]$ and $g_\theta[x, y]$ are interpolated to the output pixel grid by combining with the previous result. Since all pixels are at the exact line position, interpolation can be performed linearly between the current value and the previous value.

Computational complexity of the proposed implementations and a few common methods for Gaussian convolution is shown in Tab. 3. From the table it is expected that in the case of arbitrary θ, the xt-separated filter performs faster than the uv-separated filter with identical outcome.

Table 1. Complexity per pixel of various algorithms for Gaussian smoothing. Filter size is denoted by $N \times M$, depending on the Gaussian standard deviation σ.

Filter type	*Separability*	*Complexity*	
		Multiplications	*Additions*
convolution	xy[1]	$\lfloor N/2 \rfloor + \lfloor M/2 \rfloor + 2$	$N + M - 2$
	uv[2]	$2(N + M - 1)$	$2(N + M - 2)$
	xt[2]	$\lfloor N/2 \rfloor + M + 1$	$N + 2M - 3$
recursive	xy[1]	14	6
	uv[2]	44	36
	xt[2]	21	16
2D convolution	n.a.	NM	$NM - 1$
FFT convolution[3]	n.a.	$\log WH$	$\log WH$

[1]Restricted to Gaussian filters oriented along the x- and y-axis only, thus $\theta = 0°$ or $\theta = 90°$.
[2]Unrestricted θ.
[3]The complexity of a FFT based convolution depends on the image size $W \times H$. Note that the FFT based convolution was not fully optimized.

4 Results

Performance of the filter with respect to computation speed is shown in Tab. 4. The analysis was carried out on a normal PC (Intel Pentium III at 550 MHz) on a 512×512 image. The maximum calculation time for the proposed xt-separable recursive implementation was 65 msec. Small variations in the computation time for the xt-separable recursive implementation is due to the varying direction of the t-axis as function of σ_u, σ_v. The variation causes the processing of different pixels with respect to the filter origin, hence are influenced by the processor

cache performance. The use of recursive filters is already beneficial for $\sigma_u > 1$ or $\sigma_v > 1$. The xt-separable recursive implementation is 1.5 times slower than isotropic recursive filtering (44 msec, xy-separable implementation), but takes only 0.5 times the computational load of an uv-separable anisotropic filter. The computation time for the xt-separable filter is only 1.5 times slower than xy-aligned recursive filtering (44 msec), in which case orientation selection is only horizontal or vertical. The results correspond to the predictions in Tab. 3. For the xt-separable convolution filter, calculation is approximately 1.6 times slower than xy-aligned filtering (data not shown), and 1.3 up to 2 times faster than uv-separable filtering. Normal convolution filtering is advantageous when considering locally steered filtering, as in tracking applications, for example Fig. 2. The recursive filtering is, given its computation speed, more attractive when smoothing or differentiating the whole image array, as in feature detection, see Fig. 3. Note that in this case 108 filters were applied to the image, each filter having different parameter values. The result shown represents the per pixel maximum response over all 108 filters. Calculation time was within 10 seconds.

Table 2. Performance of various anisotropic Gaussian filter implementations. All timings in [msec], averaged over 100 trials. Image size 512×512 pixels. Filter direction $\theta = 45°$.

σ_u	σ_v	*2D convolution*[1]	*FFT convolution*	*1D convolution*[1]		*1D recursive*[2]	
				uv[3]	xt	uv[3]	xt
1.0	1.0	515	1402	100	67	128	63
1.5	1.0	516	1402	114	81	128	63
2.0	1.0	828	1402	129	100	128	63
3.0	1.0	1610	1402	166	109	128	64
5.0	1.0	4282	1402	238	140	128	65
7.0	2.0	8047	1402	341	181	128	65
7.0	4.0	10203	1402	410	205	128	63
10.0	3.0	16594	1402	483	231	128	65
10.0	5.0	18000	1402	553	256	128	64
10.0	7.0	21125	1402	622	294	128	63

[1]Filter sizes truncated at 3σ.
[2]Approximation to Gauss, see Tab. 4.
[3]Implemented by scanning along the u and the v line, respectively, and by applying bilinear interpolation between pixels.

The approximation of the two-dimensional Gaussian kernel of Eq. (4) by separable filters is not perfect due to interpolation of source values along the line $t = y + \tan\varphi\ x$. We evaluated the error for the xt-separable convolution filter in comparison to the full two-dimensional spatial convolution. The results are given in Tab. 4. Interpolation can be considered as a smoothing step with a small rectangular kernel. Hence, the effective filter is slightly larger than the theoretical size of the anisotropic Gaussian filter. As a result, the error is large

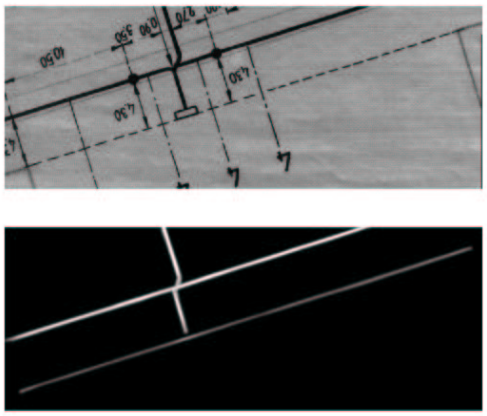


Fig. 2. Example of line detection by local anisotropic Gaussian filtering. Lines are tracked by steering the filter in the line direction. Hence, line evidence will be integrated by the large Gaussian standard deviation along the line, while maintaining spatial acuity perpendicular to the line. Original from an engineering drawing, courtesy of PNEM, The Netherlands.

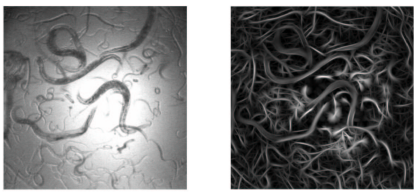


Fig. 3. Example of the detection of *C. Elegans* worms by applying recursive anisotropic Gauss filters. The original image is filtered at different orientations and scales, and the maximum response per pixel over all filters is accumulated. At each pixel, the local orientation and best fitting ellipse is available to be further processed for worm segmentation. Computation time was within 10 seconds for 5° angular resolution and 3 different aspect ratios (image size 512×512 pixels). Original courtesy of Janssen Pharmaceuticals, Beerse, Belgium.

for small σ_u, σ_v, as can be concluded from the table. For the convolution filters and $\sigma_u, \sigma_v \geq 3$, the interpolation error is of the same magnitude as the truncation error for a 3σ sized filter (last 4 rows in the table). The interpolation error is smaller for the xt-filter than for the uv-filter. For the latter, bilinear interpolation have to be performed, corresponding to a larger interpolation filter than the linear interpolation for the xt-separable filter. For the recursive filter, the interpolation error of the forward filter accumulates in the backward filter, causing a larger error. Especially the small filters are less accurate, as pointed out in [23, 24]. Note that the error due to interpolation is neglectable compared to the error made by the recursive approximation of the Gaussian filter. For the uv-separated recursive filter, the bilinear interpolation caused the error accumulation to have such a drastic effect that the result was far from Gaussian (data not shown). In conclusion, accuracy for the xt-separated convolution filter is better than bilinear interpolation combined with uv-separated filtering. For recursive filtering, error is larger due to the recursive approximation of the Gauss filter. For numerous applications the computation speed is of more importance than the precision of the result.

Table 3. Accuracy of various anisotropic Gaussian filter implementations. The maximum error over all filter orientations is shown. Error measured as root of the sum squared differences with the true Gaussian kernel.

σ_u	σ_v	*convolution uv*	*convolution xt*	*recursive xt*
1.0	1.0	0	0	0.0196
1.5	1.0	0.0195	0.0132	0.0608
2.0	1.0	0.0160	0.0131	0.0536
3.0	1.0	0.0126	0.0114	0.0324
5.0	2.0	0.0018	0.0017	0.0062
7.0	2.0	0.0015	0.0014	0.0050
7.0	4.0	0.0003	0.0003	0.0012
10.0	3.0	0.0005	0.0004	0.0017
10.0	5.0	0.0001	0.0001	0.0008
10.0	7.0	0.0001	0.0001	0.0007

5 Derivative Filters

For the uv-separable filtering approach, as for the full two-dimensional convolution, Gaussian derivative filtering can be achieved by taking the derivatives of the kernel function. For the proposed xt-separable approach, kernel differentiation is not applicable due to the misalignment of the filter directions (x, t) with respect to the direction of derivation (u, v) (see Fig. 1c). Like in [23], sample differences may be used as approximations to the true image derivatives. Hence, filtering

with a rotated version of the derivative kernel results in the image derivatives in the u, v direction, where the rotation kernel is given by Eq. (3).

The first order derivatives transform after rotation by

$$\begin{pmatrix} du \\ dv \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} \quad . \tag{28}$$

Hence, rotation of the sample differences $[1, 0, -1]$ yield

$$g_u^\theta = \frac{1}{2}\cos\theta\,(g_\theta[x+1,y] - g_\theta[x-1,y]) + \frac{1}{2}\sin\theta\,(g_\theta[x,y+1] - g_\theta[x,y-1]) \tag{29}$$

$$g_v^\theta = -\frac{1}{2}\sin\theta\,(g_\theta[x+1,y] - g_\theta[x-1,y]) + \frac{1}{2}\cos\theta\,(g_\theta[x,y+1] - g_\theta[x,y-1]) \quad . \tag{30}$$

The second order derivatives transform by

$$\begin{bmatrix} du^2 & dudv \\ dudv & dv^2 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} dx^2 & dxdy \\ dxdy & dy^2 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}^T \quad . \tag{31}$$

Transforming the second order sample differences yields

$$\begin{aligned} g_{uu}^\theta &= \cos^2\theta\,(g_\theta[x+1,y] - 2g_\theta[x,y] + g_\theta[x-1,y]) + 2\sin\theta\cos\theta \\ &\quad (g_\theta[x+1,y+1] + g_\theta[x-1,y-1] - g_\theta[x-1,y+1] - g_\theta[x+1,y-1]) \\ &\quad + \sin^2\theta\,(g_\theta[x,y+1] - 2g_\theta[x,y] + g_\theta[x,y-1]) \end{aligned} \tag{32}$$

$$\begin{aligned} g_{uv}^\theta &= \sin\theta\cos\theta\,\{(g_\theta[x,y+1] - 2g_\theta[x,y] + g_\theta[x,y-1]) - \\ &\quad (g_\theta[x+1,y] - 2g_\theta[x,y] + g_\theta[x-1,y])\} + \left(\cos^2\theta - \sin^2\theta\right) \\ &\quad (g_\theta[x+1,y+1] + g_\theta[x-1,y-1] - g_\theta[x-1,y+1] - g_\theta[x+1,y-1]) \end{aligned} \tag{33}$$

$$\begin{aligned} g_{vv}^\theta &= \sin^2\theta\,(g_\theta[x+1,y] - 2g_\theta[x,y] + g_\theta[x-1,y]) - 2\sin\theta\cos\theta \\ &\quad (g_\theta[x+1,y+1] + g_\theta[x-1,y-1] - g_\theta[x-1,y+1] - g_\theta[x+1,y-1]) \\ &\quad + \cos^2\theta\,(g_\theta[x,y+1] - 2g_\theta[x,y] + g_\theta[x,y-1]) \quad . \end{aligned} \tag{34}$$

These filters can be included into the xt-separable filtering (Eq. (23), Eq. (25), Eq. (26), Eq. (27)).

6 Conclusion

We derived the decomposition of the anisotropic Gaussian in a one dimensional Gauss filter in the x-direction followed by a one dimensional filter in a non-orthogonal direction φ. The decomposition is shown to be extremely efficient from a computing perspective. An implementation scheme for normal convolution and for recursive filtering is proposed. Also directed derivative filters are demonstrated.

We proposed a scheme for both anisotropic convolution filtering and anisotropic recursive filtering. Convolution filtering is advantageous when considering locally steered filtering, as is the case in tracking applications [11, 18, 19]. Recursive filtering is more attractive when smoothing or differentiating the whole image array, for example in feature detection [3, 15, 20, 21]. Error due to interpolation is neglectable compared to the error made by the recursive approximation of the Gaussian filter, and compared to the truncation error for convolution filters. The use of fast recursive filters [23, 24] result in an calculation time of 65 msec. for a 512×512 input image on a normal PC.

Differentiation opposite to or along the filter direction is achieved by convolution with a rotated sample difference filters. For practical applicability of orientation scale-space analysis, we believe the exact approximation of Gaussian derivatives is of less importance than the ability to compute results in limited time.

Although the decomposition of Eq. (4) is possible in higher dimensions, the method is less beneficial for three dimensional filtering applications. Only one of the axes can be chosen to be aligned with the organization of the pixels in memory. For the other directions, traversing in arbitrary directions through the pixel data is required. Hence, computational gain is only marginal for higher dimensional smoothing.

The proposed anisotropic Gaussian filtering method allows fast calculation of edge and ridge maps, with high spatial and angular accuracy. The anisotropic filters can be applied in cases where edge and ridge data is distorted. Invariant feature extraction from a 2 dimensional affine projection of a 3D scene can be achieved by tuning the anisotropic Gaussian filter, an important achievement for computer vision. When structures are inherently interrupted, as is the case for dashed line detection, anisotropic Gaussian filter may accumulate evidence along the line while maintaining spatial acuity perpendicular to the line [9]. Orientation scale-space analysis can best be based on anisotropic Gaussian filters [22]. The proposed filtering method enables the practical applicability of orientation scale-space analysis.

References

1. A. Almansa and T. Lindeberg. Fingerprint enhancement by shape adaptation of scale-space operators with automatic scale selection. *IEEE Image Processing*, 9:2027–2042, 2000.
2. J. Bigün, G. H. Granlund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optic flow. *IEEE Trans. Pattern Anal. Machine Intell.*, 13:775–790, 1991.
3. F. J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Machine Intell.*, 8(6):679–698, 1986.
4. R. Deriche. Separable recursive filtering for efficient multi-scale edge detection. In *Proceedings of the International Workshop on Machine Vision and Machine Intelligence*, pages 18–23, 1987.
5. R. Deriche. Fast algorithms for low-level vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 12:78–87, 1990.

6. L. M. J. Florack, B. M. ter Haar Romeny, J. J. Koenderink, and M. A. Viergever. Scale and the differential structure of images. *Image and Vision Comput.*, 10(6):376–388, 1992.
7. W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Anal. Machine Intell.*, 13:891–906, 1991.
8. J. Gårding and T. Lindeberg. Direct computation of shape cues using scale-adapted spatial derivative operators. *Int. J. Comput. Vision*, 17(2):163–191, 1996.
9. J. M. Geusebroek, A. W. M. Smeulders, and H. Geerts. A minimum cost approach for segmenting networks of lines. *Int. J. Comput. Vision*, 43(2):99–111, 2001.
10. L. D. Griffin. Critical point events in affine scale space. In *Scale-Space Theories in Computer Vision*, pages 165–180. Springer-Verlag, 1997.
11. A. Jonk, R. van den Boomgaard, and A. W. M. Smeulders. A line tracker. *submitted to Comput. Vision Image Understanding*.
12. S. Kalitzin, B. ter Haar Romeny, and M. Viergever. Invertible orientation bundles on 2d scalar images. In *Scale-Space Theories in Computer Vision*, pages 77–88. Springer-Verlag, 1997.
13. J. J. Koenderink. The structure of images. *Biol. Cybern.*, 50:363–370, 1984.
14. J. J. Koenderink and A. J. van Doorn. Receptive field families. *Biol. Cybern.*, 63:291–297, 1990.
15. T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, Boston, 1994.
16. T. Lindeberg. Edge detection and ridge detection with automatic scale selection. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 465–470. IEEE Computer Society, 1996.
17. P. Perona. Steerable-scalable kernels for edge detection and junction analysis. *Image Vision Comput.*, 10:663–672, 1992.
18. E.P. Simoncelli. *Distributed Representation and Analysis of Visual Motion*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, 1993.
19. E.P. Simoncelli, E.H. Adelson, and D.J. Heeger. Probability distributions of optical flow. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 310–315. IEEE Computer Society, 1991.
20. C. Steger. An unbiased detector of curvilinear structures. *IEEE Trans. Pattern Anal. Machine Intell.*, 20:113–125, 1998.
21. B. M. ter Haar Romeny, editor. *Geometry-Driven Diffusion in Computer Vision*. Kluwer Academic Publishers, Boston, 1994.
22. M. van Ginkel, P. W. Verbeek, and L. J. van Vliet. Improved orientation selectivity for orientation estimation. In M. Frydrych, J. Parkkinen, and A. Visa, editors, *Proceedings of the 10th Scandinavian Conference on Image Analysis*, pages 533–537, 1997.
23. L. J. van Vliet, I. T. Young, and P. W. Verbeek. Recursive Gaussian derivative filters. In *Proceedings ICPR '98*, pages 509–514. IEEE Computer Society Press, 1998.
24. I. T. Young and L. J. van Vliet. Recursive implementation of the Gaussian filter. *Signal Processing*, 44:139–151, 1995.