Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2011, Article ID 234090, 9 pages
doi:10.1155/2011/234090

Research Article

Boundary Treatment for Young-van Vliet Recursive Zero-Mean Gabor Filtering

Vladimír Ulman

Centre for Biomedical Image Analysis, Faculty of Informatics, Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic

Correspondence should be addressed to Vladimír Ulman, xulman@fi.muni.cz

Received 1 June 2010; Revised 23 November 2010; Accepted 8 February 2011

Academic Editor: Ricardo Merched

This paper deals with convolution setting at boundary regions for 1D convolution computed during recursive Gaussian and Gabor filtering as well as staged Gabor filtering computed more efficiently as modulation, recursive Gaussian, and demodulation. These are established fast approximations to their filters. Until recently, all the three applications of recursive filters suffered from distortions near boundary as a result of inappropriate boundary treatment. The extension of input data with constant border value is presumed. We review a recently suggested setting for recursive Gaussian and Gabor filtering. Then, a new convolution setting for the more efficient staged Gabor filtering is presented. We also offer a formula to compute the scale coefficient, using which a zero-mean Gabor filter can be obtained from either recursive or staged Gabor filter.

1. Introduction

Shortly after publication, the paper by Young et al. [1] (we will use the acronym *YvV* based on their paper [2] where the framework was first introduced) has become a standard reference for the implementation of Gaussian filtering in image processing although the paper was primarily focused on Gabor filtering. The shift towards Gaussian filtering may be in part because the authors originally developed filtering coefficients for Gaussian from which coefficients for Gabor filter were subsequently derived and in part because the Gabor filtering can be implemented with advantage as modulation, Gaussian filtering, and demodulation [3, 4]—we refer to this as to the *staged* Gabor filtering. Thus, the Gaussian filtering can become central even to Gabor filtering.

The original YvV's paper utilizes bidirectional recursive infinite-impulse-response (IIR) filtering where a "forward" pass processes input data in a causal direction and a "backward" pass processes results of the forward pass in an anticausal direction

$$u_t = i_t + \sum_{j=1}^{3} a_j u_{t-j}, \quad t = 1, \ldots, n, \tag{1}$$

$$v_t = u_t + \sum_{j=1}^{3} b_j v_{t+j}, \quad t = n, \ldots, 1. \tag{2}$$

The i_t represents input value at position t from 1D data of length n, for example, a scanning line of a digital image. Similarly, u_t and v_t hold results of forward and backward passes at position t, respectively. The a_j and b_j are the forward and backward pass coefficients, respectively. They are set based on Gaussian sigma parameter [1]. The resulting v_t is an overscaled filtering result because we, intentionally, adopted both equations and labeling from Triggs and Sdika [5]. To comply with YvV's notation introduced in [1], for Gaussian filtering set

$$a_j = b_j = -b_j^{\mathrm{YvV}}, \quad j = 1, 2, 3, \tag{3}$$

b_1^{YvV}, b_2^{YvV}, b_3^{YvV} are b_1, b_2, b_3 from (10) of [1]. To get properly scaled result, multiply v_t afterwards by B from (18) of [1]. As a matter of fact, the forward pass, (1), and the backward pass, (2), for Gaussian are the same except for the direction.

Even though the above describes only 1D convolution, there exist approaches to 2D, 3D, or nD Gaussian and Gabor filtering techniques utilizing cascades of several such 1D

convolutions and the separability of the filters; for example, isotropic Gaussian can be computed with convolutions computed along the main coordinate-system axes. If 1D convolution along general arbitrarily oriented real axis is available, a general anisotropic 2D [6] or nD [7] Gaussian filtering can be computed. A slightly more stable and a bit more computationally demanding arbitrary 2D [8] or nD [9] Gaussian filtering utilizes 1D convolutions only along integer axes (axes where orientation vector consists only of integer components). As a consequence, any arbitrarily oriented anisotropic Gabor filtering can be carried out then by using the staged filtering technique but, as we shall see in Section 4, a special care must be taken at boundaries.

It is very important to handle boundary initialization correctly in the YvV's recursive filter because the filter tends to reflect relatively large portion of its response history, see Figures 1 and 2. The span is estimated up to 3σ [5]. Hence, a care must be taken of what values of u_{-2}, u_{-1}, u_0 to initiate with the forward pass and what values of v_{n+1}, v_{n+2}, v_{n+3} to initiate with the backward pass (the *forward-backward* transition) in order to avoid cumulation of errors in the border regions of image due to the cascading of 1D convolutions. Depending on one's preference, the input data may be prefixed with a constant sequence of $i_t = i_1$, $t = -\infty, \ldots, 0$ and, similarly, postfixed with $i_t = i_n$, $t = n+1, \ldots, \infty$ to facilitate filtering. A padding with zeros or mirroring input data near border are also plausible choices.

In the two figures, we illustrate how different techniques for a Gaussian and Gabor filtering perform near boundary on some randomly generated data. The direct filtering, both Gaussian and Gabor, with YvV boundary initialization assumes that backward pass is initiated with a steady-state response of the filter. But recursive filters, typically, decay to the steady state on constant input only after several iterations. The constant sequence would have to begin already a way ahead the image boundary, which cannot be always guaranteed, so that the forward pass, if continued after the image boundary, would enter the boundary region already with constant steady-state responses. Then, the YvV forward-backward initialization would perform well. In the staged techniques, the use of Gaussian filtering with any of the established boundary treatment may be tempting, but it is incorrect as demonstrated in Figure 2. Again, the assumption of constant boundaries in these cases of staged Gabor techniques is violated because of the reasons to be discussed in detail in Section 4. Note that all of the currently available boundary treatment techniques for YvV recursive filtering are presented in both figures, except for those we are going to discuss in this paper.

In this paper, we deal with approaches to correct initialization of YvV filters in applications of direct Gaussian in Section 2, direct Gabor in Section 3, and staged Gabor filtering in Section 4. The first and the second of the three cases were dealt with in [5] explicitly and implicitly, respectively. We had only replayed their derivation for the second case, a complex direct Gabor filtering, to arrive to some elegant solution presented in this paper. The last case forms the main contribution of this paper. We always consider extending the input 1D data with i_1 and i_n in the anti-causal and causal directions, respectively, that is, extending input data line with constant value of the pixel at line's end. Finally, in Section 5, we offer a formula to compute the scale coefficient, using which a zero-mean Gabor filter can be obtained from either direct or staged Gabor filter.

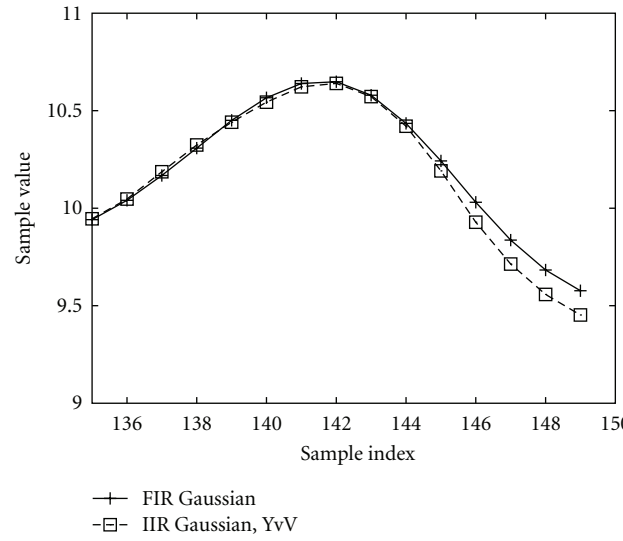

Figure 1: Example of an error after a direct Gaussian filtering on sample input data: $n = 149$, Sigma = 3, input extended constantly with the border value. A result of the incorrect forward-backward transition according to YvV is compared to the FIR implementation with constantly extended border as well. The FIR serves as a ground-truth result. The error spans 5 data positions.

This paper is supposed to summarize all necessary information for efficient and correct 1D recursive implementation of any of the three filtering applications. Such an implementation would only require reading the paper by Young et al. [1], in addition to this one, to learn how to find coefficients for the forward and backward passes.

2. Direct Gaussian Filtering

In this section, we will remind of the results of the recent paper by Triggs and Sdika [5] who derived closed-form equations for computing correct forward and backward initialization values based solely on filter parameters. Indeed, their solution does not require any prior knowledge of the filter, for example, Gaussian sigma, dimensionality, isotropy, the way filter was derived, and so forth. It only requires that the filtering should be realized by means of forward and consecutive backward passes, (1) and (2), even with arbitrary depth of recursion. Based on their assumption that linear filtering can be written using matrix multiplication, they managed to first describe the filtering itself and then, the most importantly, also the forward-backward transition.

Triggs and Sdika [5] confirmed that for 3rd order recursive filters according to (1) and (2), the correct initialization of the forward pass with the steady-state response to an infinite stream of i_1 is

$$u_{-2} = u_{-1} = u_0 = \frac{i_1}{1 - a_1 - a_2 - a_3}, \quad u_t \in \mathbb{R},\ a_j \in \mathbb{R}. \tag{4}$$

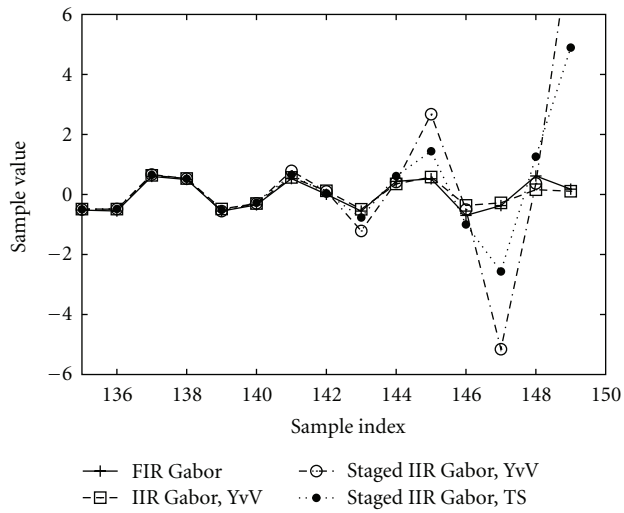

Figure 2: Example of errors after a direct (square marks) and staged (circle marks) Gabor filtering on sample input data: $n = 149$, Sigma = 3, period = 4 px, input extended constantly with the border value. For the staged cases, which is always a modulation, Gaussian filtering, and demodulation, we show that any forward-backward transition for direct Gaussian filtering (discussed in Section 2) shall not be used, despite that this transition may be correct for regular Gaussian filtering. The results on the forward-backward transition are compared to the FIR implementation with constantly extended border as well. The FIR serves as a ground-truth result. The error spans 7 data positions. Only real parts of the complex filtering results are shown here.

However, their main contribution was the derivation of an explicit formula for the forward-backward transition. We reproduce their result, contained in their (14), (15), and (5) from [5], for $k = l = 3$ and $i_+ = i_n$

$$\begin{pmatrix} v_n \\ v_{n+1} \\ v_{n+2} \end{pmatrix} = M \begin{pmatrix} u_n - u_+ \\ u_{n-1} - u_+ \\ u_{n-2} - u_+ \end{pmatrix} + \begin{pmatrix} v_+ \\ v_+ \\ v_+ \end{pmatrix}, \tag{5}$$

$$u_+ = \frac{i_n}{1 - a_1 - a_2 - a_3}, \qquad v_+ = \frac{u_+}{1 - b_1 - b_2 - b_3}, \tag{6}$$

where M is a 3×3 matrix with elements $[m_{i,j}]$, $i, j = 1, 2, 3$,

$$s = \frac{1}{(1 + a_1 - a_2 + a_3)(1 - a_1 - a_2 - a_3)(1 + a_2 + (a_1 - a_3)a_3)}, \tag{7}$$

$$m_{1,1} = s(-a_3a_1 + 1 - a_3a_3 - a_2), \tag{8}$$

$$m_{1,2} = s(a_3 + a_1)(a_2 + a_3a_1), \tag{9}$$

$$m_{1,3} = sa_3(a_1 + a_3a_2), \tag{10}$$

$$m_{2,1} = s(a_1 + a_3a_2), \tag{11}$$

$$m_{2,2} = -s(a_2 - 1)(a_2 + a_3a_1), \tag{12}$$

$$m_{2,3} = -sa_3(a_3a_1 + a_3a_3 + a_2 - 1), \tag{13}$$

$$m_{3,1} = s(a_3a_1 + a_2 + a_1a_1 - a_2a_2), \tag{14}$$

$$m_{3,2} = s(a_1a_2 + a_3a_2a_2 - a_1a_3a_3 - a_3a_3a_3 - a_3a_2 + a_3), \tag{15}$$

$$m_{3,3} = sa_3(a_1 + a_3a_2). \tag{16}$$

Note that for Gaussian, it holds $a_j = b_j$, $j = 1, 2, 3$ in (1) and (2).

Figure 3 confirms the desired effect of this forward-backward transition. This initialization uses the last input value and three last values from the forward pass to initiate the backward pass.

3. Direct Gabor Filtering

A 1D complex Gabor filter consists of a 1D Gaussian filter associated with some sigma and a complex exponential $e^{kx\omega}$ where x is position within the filter, ω is the filter frequency, and $k^2 = -1$ is the imaginary unit. YvV filter coefficients from (1) and (2) then change to complex

$$\begin{aligned} a_j &= -b_j^{\mathrm{YvV}} e^{kj\omega}, \\ b_j &= -b_j^{\mathrm{YvV}} e^{-kj\omega}, \quad j = 1, 2, 3, \end{aligned} \tag{17}$$

with b_j^{YvV} being b_j from (10) of [1]. Both forward and backward passes happen in the complex domain.

The correct initialization for the YvV's forward pass is (4) with complex coefficients a_j,

$$\begin{aligned} u_{-2} = u_{-1} = u_0 \\ &= \frac{i_1}{1 - a_1 - a_2 - a_3}, \quad u_t \in \mathbb{C},\ a_j \in \mathbb{C} \\ &= \frac{i_1}{1 + b_1^{\mathrm{YvV}} e^{k\omega} + b_2^{\mathrm{YvV}} e^{2k\omega} + b_3^{\mathrm{YvV}} e^{3k\omega}}, \quad b_j^{\mathrm{YvV}} \in \mathbb{R}. \end{aligned} \tag{18}$$

Similarly for the YvV's forward-backward transition, following the derivation from [5] right from the beginning ((3) of [5]) in the complex domain, we arrived to (5) and (6), and we found that correct initialization matrix M' is the Gaussian's matrix M element-wise multiplied with complex exponentials:

$$M' = \begin{bmatrix} m_{1,1} & m_{1,2} \cdot e^{k\omega} & m_{1,3} \cdot e^{2k\omega} \\ m_{2,1} \cdot e^{k\omega} & m_{2,2} \cdot e^{2k\omega} & m_{2,3} \cdot e^{3k\omega} \\ m_{3,1} \cdot e^{2k\omega} & m_{3,2} \cdot e^{3k\omega} & m_{3,3} \cdot e^{4k\omega} \end{bmatrix}. \tag{19}$$

Note that since in the case of Gabor filtering the forward and backward coefficients are not the same, $a_j \neq b_j$, the direct substitution into the Gaussian's matrix M does not lead to the valid result. The matrix for Gabor filter must be evaluated with different a_j and b_j right from the start leading into an overly complicated matrix from which, after realizing that a_j

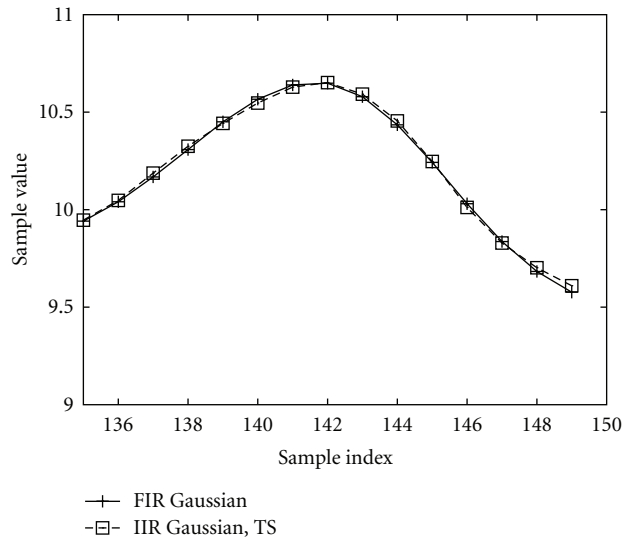

Figure 3: The forward-backward transition initiates correctly when compared to an FIR implementation. Compare it with Figure 1.

and b_j differ only in the complex exponential (17), the result of (19) is derived.

A more elegant and short derivation exists. During their derivation, Triggs and Sdika [5] arrived to the implicit definition of general M for YvV recursive filtering, (18) of [5],

$$M = I_1 + BMA, \tag{20}$$

in which I_1 is a matrix with a 1 in the top-left corner and zeros elsewhere, A and B "encodes" the forward and backward passes, respectively, for example, when $k = l = 3$:

$$A = \begin{bmatrix} a_1 & a_2 & a_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \tag{21}$$

For details, refer to their original paper. We now consider again the Gaussian filtering as defined in Section 2, that is, with $a_j = b_j$, $a_j \in \mathbb{R}$, so that we can define A' for Gabor filtering explicitly as $A' = e^{k\omega} D^{-1} A D$ and similarly $B' = e^{-k\omega} D B D^{-1}$ with

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & e^{k\omega} & 0 \\ 0 & 0 & e^{2k\omega} \end{bmatrix}. \tag{22}$$

Using (20) and generality of their solution, we have $M' = I_1 + B'M'A'$ also for a Gabor filter. After substitution to the right hand side of it,

$$M' = I_1 + e^{-k\omega} D B D^{-1} M' e^{k\omega} D^{-1} A D, \tag{23}$$

the terms $e^{-k\omega}$ and $e^{k\omega}$ cancel themselves, and we multiply appropriately from both sides,

$$D^{-1} M' D^{-1} = D^{-1} I_1 D^{-1} + B D^{-1} M' D^{-1} A. \tag{24}$$

Since the I_1 has zeros everywhere except for the top-left corner, when multiplying with any matrix, only the value at this corner is not zeroed in the result. Hence, $D^{-1} I_1 D^{-1} = I_1$. Finally, the forward-backward transition for Gabor filter is expressed by means of Gaussian filtering,

$$(D^{-1} M' D^{-1}) = I_1 + B(D^{-1} M' D^{-1}) A, \tag{25}$$

from which we realize that $M' = DMD$.

Figure 4 shows an application of the presented forward-backward transition.

4. Staged Gabor Filtering

The staged Gabor filtering makes use of the equality $e^{q+w} = e^q \cdot e^w$ in the 1D convolution expression

$$o_t = \sum_l i_l \cdot \text{gauss}(\sigma, t - l) \cdot e^{k(t-l)\omega}, \tag{26}$$

to yield the convolution

$$o_t = e^{kt\omega} \sum_l \left[i_l e^{-kl\omega} \right] \cdot \text{gauss}(\sigma, t - l). \tag{27}$$

The Gabor filtering is then split into three stages: Modulation of input data by $e^{-kl\omega}$, followed by a Gaussian convolution, and demodulation of the convolution result by $e^{kt\omega}$ in the end. This is a generally favored approach to a convolution with complex Gabor filter (Section 3) because it requires less real multiplications and additions per pixel [4] and is easier to implement compared to the direct Gabor filtering. Both are a consequence of replacing truly complex Gabor convolution with two real Gaussian convolutions.

A real constant input data $i_t = c$, $t = 1, \ldots, n$ is no longer constant after a modulation which gives *complex* $i_t^{\text{mod}} = ce^{-kt\omega}$. This is an important observation for the subsequent Gaussian filtering in the second stage since the modulated data i_t^{mod} is actually an input to it. In particular, the initialization for Gaussian filtering presented in Section 2 cannot be used because it expects that input data is extended with a real constant. In the following, we focus on the two real Gaussian convolutions using YvV's forward and backward passes as defined in (1)–(3), $a_j = b_j$, but with initialization based on the presumption that input data line is extended with complex periodic function of ω.

4.1. Application to Higher Dimensions. Before we get to an initialization of individual 1D convolution, we will first discuss its applicability in a higher dimensional space to ease the comprehension of the rest of this section. This is motivated by the fact that any Gaussian filtering can be realized as a cascade of several 1D Gaussian filtering [7, 9]. Note that the filtering happens after the modulation stage. Also note that an input to the next filtering in a cascade is actually an output of the preceding one. Consider, for example, a 2D Gabor filtering task with Gaussian envelope separable along the x and y axes and with frequency tuning of $|(dx, dy)|$ in the direction of vector (dx, dy). One would typically compute the product $i_{x,y} \cdot e^{-k(x \cdot dx + y \cdot dy)\omega}$ for a

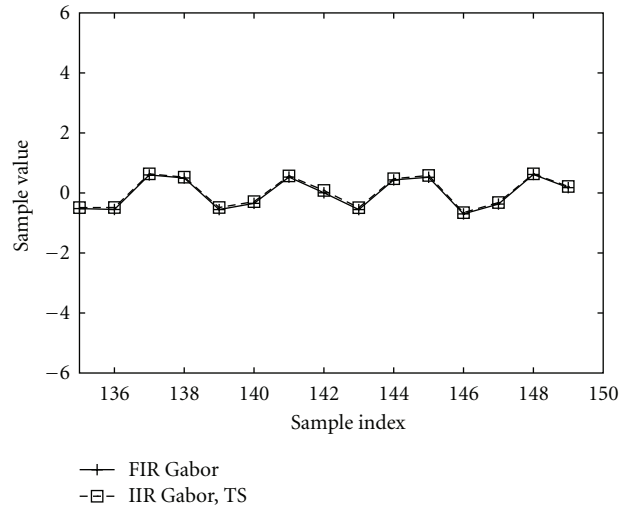

FIGURE 4: The forward-backward transition for the direct Gabor filtering initiates correctly when compared to an FIR implementation. Compare it with Figure 2. Only real parts of the complex filtering results are shown here.

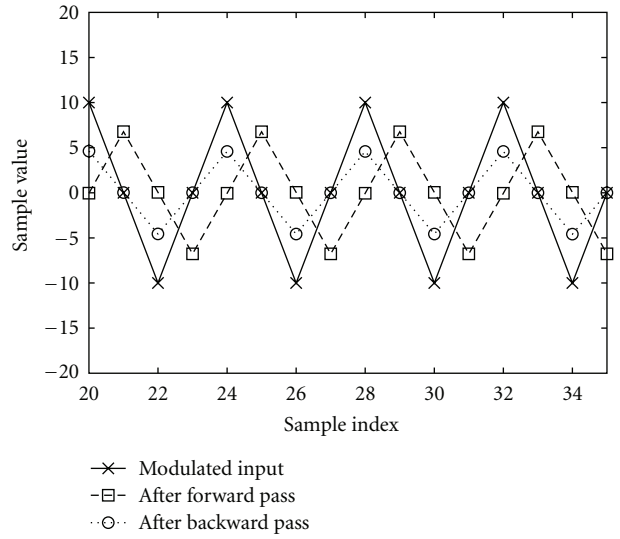

FIGURE 5: Illustration of a result of the forward pass u_t (square marks) during a Gaussian filtering on a modulated input i_t^{mod} (cross marks). Only the amplitude and phase of the signal are changed. After the backward pass (circle marks), the phase is synchronized again with the modulated input as the Gaussian filtering is fully computed now. The period of signal is kept unchanged after each of the two passes.

modulation of input data $i_{x,y}$ followed by 1D Gaussian convolutions along the x and y axes (in arbitrary order) and then demodulation. The staged convolution equation, (27), for 2D would be

$$o_{x,y} = e^{k(x\cdot dx+y\cdot dy)\omega}\left[\sum_n\left[\sum_m\left[i_{m,n}e^{-k(m\cdot dx+n\cdot dy)\omega}\right]\times\mathrm{gauss}_x(\sigma_x,x-m)\right]\mathrm{gauss}_y\left(\sigma_y,y-n\right)\right]. \tag{28}$$

But we may easily interleave the modulation with filtering

$$o_{x,y} = e^{k(x\cdot dx+y\cdot dy)\omega}\left[\sum_n\left[\sum_m\left[i_{m,n}e^{-k(m\cdot dx)\omega}\right]\mathrm{gauss}_x(\sigma_x,x-m)\right]\times e^{-k(n\cdot dy)\omega}\mathrm{gauss}_y\left(\sigma_y,y-n\right)\right]. \tag{29}$$

The point of this change was only to show that it is correct to split the 2D, or generally nD, Gabor filtering to a cascade of *pairs*, each pair consisting of "1D" modulation in the direction of the subsequent 1D convolution and the 1D convolution itself. Hence, in the following subsections, we can assume that the 1D Gaussian filtering with complex period boundary extension was always preceded with an appropriate modulation. This is correct for any 1D filtering in a cascade even when modulation is only conducted prior the nD Gaussian filtering, as in (28).

4.2. Forward Initialization. For the forward initialization, we start by computing what would be the response function of the forward pass applied on complex periodic data $i_t^{\mathrm{mod}} = i_1 e^{-kt\omega}$, $t = -\infty,\ldots,0$. Suppose that, based on observation depicted in Figure 5, the response u_t has the shape of $Re^{-kt\omega+k\varphi}$ where R is a complex gain and φ is some phase shift. Rewriting (1), we obtain

$$u_t = Re^{-kt\omega+k\varphi} = i_1 e^{-kt\omega} + \sum_{j=1}^{3} a_j R e^{-k(t-j)\omega+k\varphi}, \tag{30}$$

which yields (after division by $e^{-kt\omega+k\varphi}$)

$$R = i_1 e^{-k\varphi} + \sum_{j=1}^{3} a_j R e^{kj\omega}, \tag{31}$$

$$R = \frac{i_1 e^{-k\varphi}}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}}, \quad R\in\mathbb{C},\ a_j\in\mathbb{R}, \tag{32}$$

$$= \frac{i_1 e^{-k\varphi}}{1 + b_1^{\mathrm{YvV}} e^{k\omega} + b_2^{\mathrm{YvV}} e^{2k\omega} + b_3^{\mathrm{YvV}} e^{3k\omega}}, \quad b_j^{\mathrm{YvV}}\in\mathbb{R}.$$

Note that R depends on φ, but the value of u_t is independent of it.

The expression in (32) requires the knowledge of the original value (prior to the modulation) of input data i_1. Value of i_1 may be also computed by reverting the effect of modulation $i_1 = i_1^{\mathrm{mod}} e^{k\omega}$. Alternatively, we realize that the forward initialization values u_{-2}, u_{-1}, u_0 are adjacent in position to the first convolution position $t = 1$. In general, we seek initialization values u_{t-j}, $j = 1,2,3$, for some 1D

convolution starting at position t with a value i_t^{mod} (recall that $u_t = R \cdot e^{-kt\omega+\mathrm{k}\varphi}$):

$$u_{t-j} = \frac{i_t^{\mathrm{mod}}\, e^{kt\omega} e^{-k\varphi}}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}} \cdot e^{-k(t-j)\omega+k\varphi}. \quad (33)$$

We observe that (de)modulation terms including t and φ actually cancel out. The only term with t that remains is the i_t^{mod} which is the first value the 1D convolution receives on its input. The convolution then uses this value to compute the initialization values and starts the forward pass of the staged Gabor filtering with

$$\begin{aligned} u_{-2} &= \frac{i_1^{\mathrm{mod}}}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}} e^{3k\omega}, \\ u_{-1} &= \frac{i_1^{\mathrm{mod}}}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}} e^{2k\omega}, \\ u_0 &= \frac{i_1^{\mathrm{mod}}}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}} e^{k\omega}. \end{aligned} \quad (34)$$

Note that the right-hand side of (18) for the direct Gabor filtering and the right-hand side of (32) for R are the same except for the phase shift $e^{-k\varphi}$. This evocates impression that we actually perform Gaussian filtering with initialization for the direct Gabor filtering. The handling of forward-backward transition shall support this observation to some extent as well.

4.3. Backward Initialization. For the forward-backward transition, we first demodulate input data, so that it appears constant, and we may reuse previous result for the direct Gabor filtering from Section 3. We examine the original YvV filtering equations (1), (2) for Gaussian, that is, $a_j = b_j$, on a modulated input data $i_t^{\mathrm{mod}} = i_n e^{-kt\omega}$, $t = n+1, \ldots, \infty$, after both equations are multiplied with demodulation term $e^{kt\omega}$:

$$\begin{aligned} u_t \cdot e^{kt\omega} &= \left[i_t^{\mathrm{mod}} + \sum_{j=1}^{3} a_j u_{t-j} \right] \cdot e^{kt\omega}, \\ v_t \cdot e^{kt\omega} &= \left[u_t + \sum_{j=1}^{3} a_j v_{t+j} \right] \cdot e^{kt\omega}. \end{aligned} \quad (35)$$

Let $u'_t = u_t \cdot e^{kt\omega}$ and $v'_t = v_t \cdot e^{kt\omega}$,

$$\begin{aligned} u'_t &= i_n + \sum_{j=1}^{3} \left(a_j e^{kj\omega} \cdot u_{t-j} e^{kt\omega} e^{-kj\omega} \right), \\ v'_t &= u'_t + \sum_{j=1}^{3} \left(a_j e^{-kj\omega} \cdot v_{t+j} e^{kt\omega} e^{kj\omega} \right), \end{aligned} \quad (36)$$

from which

$$\begin{aligned} u'_t &= i_n + \sum_{j=1}^{3} \left(a_j e^{kj\omega} \cdot u'_{t-j} \right), \\ v'_t &= u'_t + \sum_{j=1}^{3} \left(a_j e^{-kj\omega} \cdot v'_{t+j} \right). \end{aligned} \quad (37)$$

This pair of equations is the YvV Gabor filtering (compare with the first paragraph of Section 3) on constant input data i_n. We may use the result for the direct Gabor forward-backward transition where matrix M' from (19) is used

$$\begin{pmatrix} v'_n \\ v'_{n+1} \\ v'_{n+2} \end{pmatrix} = M' \begin{pmatrix} u'_n - u'_+ \\ u'_{n-1} - u'_+ \\ u'_{n-2} - u'_+ \end{pmatrix} + \begin{pmatrix} v'_+ \\ v'_+ \\ v'_+ \end{pmatrix}, \quad (38)$$

$$\begin{aligned} u'_+ &= \frac{i_n}{1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}}, \\ v'_+ &= \frac{u'_+}{1 - a_1 e^{-k\omega} - a_2 e^{-2k\omega} - a_3 e^{-3k\omega}}. \end{aligned} \quad (39)$$

We aim to rewrite (38) back for the modulated input data. The following equations are useful:

$$\begin{aligned} \begin{pmatrix} v'_n \\ v'_{n+1} \\ v'_{n+2} \end{pmatrix} &= \begin{pmatrix} e^{kn\omega} & 0 & 0 \\ 0 & e^{k(n+1)\omega} & 0 \\ 0 & 0 & e^{k(n+2)\omega} \end{pmatrix} \begin{pmatrix} v_n \\ v_{n+1} \\ v_{n+2} \end{pmatrix}, \\ M' \begin{pmatrix} u'_+ \\ u'_+ \\ u'_+ \end{pmatrix} &= M' \begin{pmatrix} e^{kn\omega} & 0 & 0 \\ 0 & e^{kn\omega} & 0 \\ 0 & 0 & e^{kn\omega} \end{pmatrix} \begin{pmatrix} u_+ \\ u_+ \\ u_+ \end{pmatrix}. \end{aligned} \quad (40)$$

In a similar way, $(u'_n, u'_{n-1}, u'_{n-2})^T$ and $(v'_+, v'_+, v'_+)^T$ can be expressed. These equations enable us to express (38) only with u_n, u_+, v_n, and v_+. We multiply the whole equation with the matrix E from the left,

$$E = \begin{pmatrix} e^{-kn\omega} & 0 & 0 \\ 0 & e^{-k(n+1)\omega} & 0 \\ 0 & 0 & e^{-k(n+2)\omega} \end{pmatrix}. \quad (41)$$

This leaves the vector $(v_n, v_{n+1}, v_{n+2})^T$ alone in the left-hand side of the equation. The right-hand side of it can be regarded as a subtraction of the two terms $A - B$:

$$A = EM' \begin{pmatrix} e^{kn\omega} & 0 & 0 \\ 0 & e^{k(n-1)\omega} & 0 \\ 0 & 0 & e^{k(n-2)\omega} \end{pmatrix} \begin{pmatrix} u_n \\ u_{n-1} \\ u_{n-2} \end{pmatrix}, \quad (42)$$

$$\begin{aligned} B = E \Bigg[& M' \begin{pmatrix} e^{kn\omega} & 0 & 0 \\ 0 & e^{kn\omega} & 0 \\ 0 & 0 & e^{kn\omega} \end{pmatrix} \begin{pmatrix} u_+ \\ u_+ \\ u_+ \end{pmatrix} \\ & - \begin{pmatrix} e^{kn\omega} & 0 & 0 \\ 0 & e^{kn\omega} & 0 \\ 0 & 0 & e^{kn\omega} \end{pmatrix} \begin{pmatrix} \mathrm{v}_+ \\ v_+ \\ v_+ \end{pmatrix} \Bigg]. \end{aligned} \quad (43)$$

Regarding the term A, the multiplication of the three 3×3 matrices results in

$$A = M \begin{pmatrix} u_n \\ u_{n-1} \\ u_{n-2} \end{pmatrix}, \tag{44}$$

where M is the forward-backward transition matrix for *Gaussian*, that is, the real matrix from (5).

Regarding the term B, it is easy to show that M' multiplied from right by a diagonal matrix is equal to M' multiplied from left by the same diagonal matrix provided the diagonal is constant. Hence, we may swap the multiplication with M' in (43). Furthermore, we let $D = 1 - a_1 e^{k\omega} - a_2 e^{2k\omega} - a_3 e^{3k\omega}$ and $\overline{D}$ be its complex conjugate, so that

$$u_+ = \frac{i_n^{\mathrm{mod}}}{D}, \qquad v_+ = \frac{u_+}{\overline{D}}. \tag{45}$$

After some manipulations, we end up with

$$B = \left[\begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{-k\omega} & 0 \\ 0 & 0 & e^{-2k\omega} \end{pmatrix} \left[M' - \begin{pmatrix} \frac{1}{\overline{D}} & 0 & 0 \\ 0 & \frac{1}{\overline{D}} & 0 \\ 0 & 0 & \frac{1}{\overline{D}} \end{pmatrix} \right] \times \begin{pmatrix} \frac{1}{D} & 0 & 0 \\ 0 & \frac{1}{D} & 0 \\ 0 & 0 & \frac{1}{D} \end{pmatrix} \right] \begin{pmatrix} i_n^{\mathrm{mod}} \\ i_n^{\mathrm{mod}} \\ i_n^{\mathrm{mod}} \end{pmatrix}, \tag{46}$$

in which the term in outer square brackets can be precomputed prior to the computation of convolution and M' is the forward-backward transition matrix for *Gabor* filtering, that is, the complex matrix from (19). Moreover, since the vector $(i_n^{\mathrm{mod}}, i_n^{\mathrm{mod}}, i_n^{\mathrm{mod}})^T$ is constant, the precomputation can be simplified. For instance, the second row of B evaluates to

$$\left[e^{-k\omega} \cdot \left(m_{2,1} e^{k\omega} + m_{2,2} e^{2k\omega} + m_{2,3} e^{3k\omega} - \frac{1}{\overline{D}} \right) \cdot \frac{1}{D} \right] \cdot i_n^{\mathrm{mod}}. \tag{47}$$

Finally, to summarize the forward-backward transition

$$\begin{pmatrix} v_n \\ v_{n+1} \\ v_{n+2} \end{pmatrix} = M \begin{pmatrix} u_n \\ u_{n-1} \\ u_{n-2} \end{pmatrix} - \begin{pmatrix} B_1 & 0 & 0 \\ 0 & B_2 & 0 \\ 0 & 0 & B_3 \end{pmatrix} \begin{pmatrix} i_n^{\mathrm{mod}} \\ i_n^{\mathrm{mod}} \\ i_n^{\mathrm{mod}} \end{pmatrix},$$
$$B_l = \frac{e^{-(l-1)k\omega}}{D} \cdot \left(\sum_{j=1}^{3} m_{l,j} e^{(l+j-2)k\omega} - \frac{1}{\overline{D}} \right), \tag{48}$$

M and $m_{l,j}$ are defined in Section 2. Figure 6 confirms the desired effect of this forward-backward transition. Note that the initialization for the backward pass works with i_n^{mod}, which is an (already modulated) input to the filtering (to the second stage), instead of working with i_n, which is an (original, user's) input to the modulation (to the first stage). Thus, the original data i_t, after being processed in the first stage, is not required any more during the filtering or in the initialization of it.

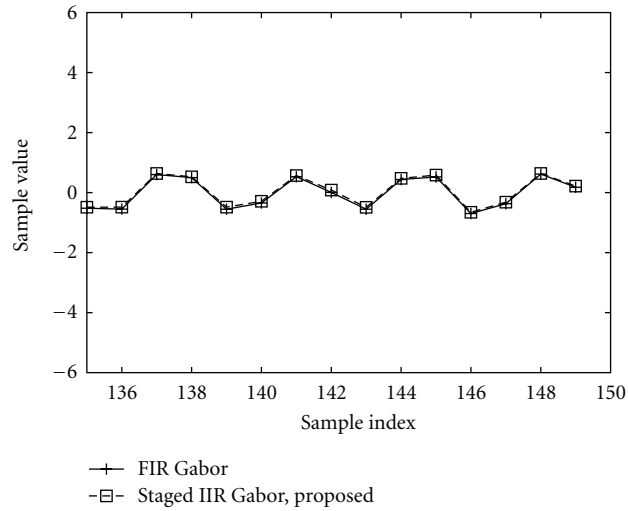

Figure 6: The forward-backward transition for the staged Gabor filtering initiates correctly when compared to an FIR implementation. Compare it with Figure 2. Only real parts of the complex filtering results are shown here.

4.4. Theoretical and Experimental Comparison. Let us make a comment on the solution by Bernardino and Santos-Victor [4], which was, to our best knowledge, so far the only published solution for the forward-backward transition during the *staged* Gabor filtering based on the YvV recursive filters. Though they approached it with the Z-transform, we found that our solution is equal to theirs both in terms of accuracy and speed (measured as number of multiply-add operations per pixel, ops/px, required to process the transition). The major difference, however, is that while we establish initial values v_n, v_{n+1}, and v_{n+2} for the backward pass *only* from the filter coefficients a_1, a_2, and a_3, they require poles of Z-transform of the filter *in addition* to its coefficients. Their formulae for the transition simply require more input parameters. This does not limit applicability of their approach. Only for filters for which we know only their coefficients, for example, the "latest" YvV filters from 2002 [1], the poles must be computed additionally. Note that the poles are roots of a 3rd order polynomial (the 3rd order refers to (1) and (2) in which we fixed the order of filter recursivity) that is found in denominator of the filter's transfer function [10]. "Our approach is closer to the spirit of Triggs and Sdika [5] since we are able to treat any general recursive filter that is according to the YvV scheme regardless of how the filter coefficients were derived and what its poles are." Note that Bernardino and Santos-Victor [4] used the "older" YvV filter from 1998 [11], where filter poles were explicitly computed to derive filter coefficients, so that all parameters for their forward-backward transition were easily available.

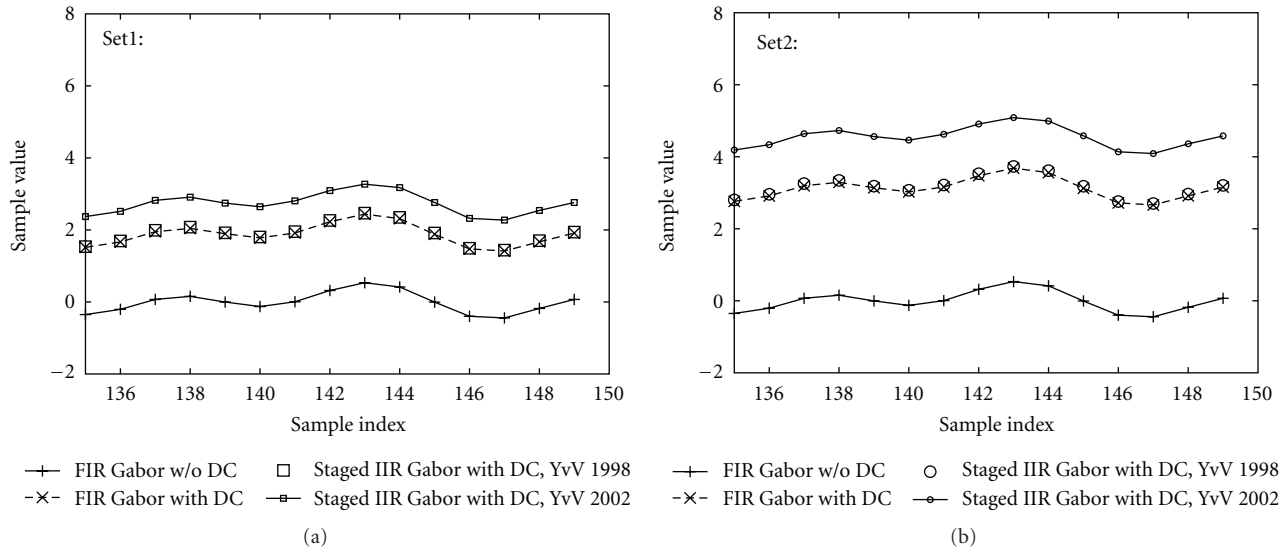

Figure 7: Comparison of results of the same Gabor filter computed with different filtering without the zero-mean correction, denoted as *with DC*. Two input images were used: One was randomly generated, denoted as *Set1*, while the second is a copy of the first one with a constant value added to every image position. We may also notice how differently respective filtering variants respond to the change of mean value of input data, which is a result of different approximations involved in the respective variants. Only real parts of the complex filtering results are shown here.

We have also consulted and tested their C++ implementation, which the authors advertise in their original paper. We tested it over several filters on some randomly generated data. In each case, we inserted filter coefficients a_i established by their implementation to our implementation (it cannot be done the other way round because we do not have and we don't even expect to have the poles at hand) to ensure the same filter is used and compared the results. For 1D tests and with respect to errors due to the floating-point arithmetics, both filtering implementations returned with the same values, not only within the border regions. Minor difference is that they actually compute values of v_{n+1}, v_{n+2}, and v_{n+3} needing, therefore, to conduct one iteration more of the backward pass, which is another 14 ops/px more for every transition yielding 94 ops/px in total in contrast to 54 ops/px required by our solution for every forward-backward transition.

We argue that our solution appears to be easier to use as it is independent of filter poles.

5. Towards Zero-Mean Gabor Filter

"When implementing a Gabor filter, one should not forget to consider one of the classical results on Gabor filtering, since the real part of a filtering result is biased, see Figure 7." A generally acknowledged solution to remove the bias is to subtract a scaled Gaussian filtered input image from the real part [4, 12, 13]. A Gaussian filter with exactly the same parameters as those of the Gaussian envelope of the Gabor filter is used.

In the staged filtering, enforcing the zero-mean property is nothing else but yet another exactly the same filtering with the same Gaussian this time on original nonmodulated input image [4]. Indeed, the staged 1D zero-mean complex Gabor filtering involves three 1D convolutions with the same Gaussian filters (the same coefficients a_j and b_j). Twice the (real) filtering operates on complex-modulated input with boundaries treated according to Section 4 and once the filtering operates on real input image with boundaries treated according to Section 2. The latter result is then scaled and subtracted from the real-part result of the first two filtering.

For the scale coefficient, one typically uses the value of Fourier transform of the Gaussian envelope at the Gabor filter's frequency [4, 13]. Actually, Fourier transform of the YvV recursive filter is used [4]. Note that since the filter is an approximation to Gaussian filter, the explicit formula for Fourier transform of Gaussian can't be used for the computation of the scale coefficient.

Because we deal with complex Gabor filtering, we may alternatively obtain the scale coefficient from Fourier transform of the Gabor filter at the "zero" (the DC) frequency [12]. We offer here a simple and fast to compute formula that is alternative and equal to the solution by Bernardino and Santos-Victor [4]. We make use of the fact that transfer function of a YvV filter, given with the system in (1) and (2), is

$$H(z) = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2} - a_3 z^{-3}} \cdot \frac{1}{1 - b_1 z^{1} - b_2 z^{2} - b_3 z^{3}}. \tag{49}$$

If the coefficients for some Gaussian are used, that is, $a_j = b_j$ given in (3), then the expression $BH(z)$ will become transfer function of the Gaussian filter. Note that it is a property of YvV Gaussian filters from 2002 [1] that $B = (1+a_1+a_2+a_3)^2$, $a_j \in \mathbb{R}$ and that $1+a_1+a_2+a_3$ equals to B, (10) from [2], and to α, (12) from [11]. Following Young et al. [1], we change

the transfer function in a fashion similar to changes of filter coefficients in (17)

$$H'(z) = \frac{1}{1 - \sum_{j=1}^{3} a_j e^{kj\omega} z^{-j}} \cdot \frac{1}{1 - \sum_{j=1}^{3} a_j e^{-kj\omega} z^{j}}. \quad (50)$$

The expression $BH'(z)$ will become transfer function of the Gabor filter with frequency ω and Gaussian envelope given above. Finally, the Fourier transform of the Gabor filter for the DC frequency can be obtained from $BH'(z)$ by letting $z = e^{k0} = 1$. Making use of complex conjugates in the denominators of (50), the scale coefficient is equal to

$$\frac{(1 + a_1 + a_2 + a_3)^2}{\left|1 - \sum_{j=1}^{3} a_j e^{kj\omega}\right|^2}, \quad a_j = -b_j^{\mathrm{YvV}},\ b_j^{\mathrm{YvV}} \in \mathbb{R}, \quad (51)$$

where $|z|^2$ is square of a magnitude of z.

6. Conclusion

We have presented convolution setting at boundary regions for 1D convolution with real Gaussian and complex Gabor filters for the case the filters are implemented as (de facto) the standard recursive filters as suggested by Young et al. [1]. These are efficient approximations to their filters. However, computing Gabor convolution as modulation, Gaussian filtering and demodulation, that is, the staged approach, yields even higher efficiency. On the other hand, Gaussian filtering preceded by a modulation requires a different boundary setting than that for regular direct Gaussian filtering. We presume the extension of input data with constant border value. We have reviewed the correct and easy-to-compute solution by Triggs and Sdika [5] for the regular Gaussian filtering and replayed their approach to show how to cope with the Gabor filtering. Based on these results, we have derived a new solution for the more efficient staged Gabor filtering. Finally, we offered a simple formula, a function of solely recursive filter coefficients, to compute the scale coefficient, using which a zero-mean Gabor filter can be obtained from either recursive or staged Gabor filter.

Sample C++ implementation of the three applications based on YvV filtering with correct initialization can be downloaded from the author's web page at http://cbia.fi.muni.cz/projects/sample-iir-filter-implementations.html .

Acknowledgments

The author would like to thank M. Kozubek for fruitful discussions and suggestions. The author would also like to thank the reviewers of this paper for very useful comments and also for pointing out the more elegant deduction of the formula of (19). This work has been supported by the Ministry of Education of the Czech Republic (Grants no. MSM0021622419, LC535, and 2B06052).

References

[1] I. T. Young, L. J. van Vliet, and M. van Ginkel, "Recursive Gabor filtering," *IEEE Transactions on Signal Processing*, vol. 50, no. 11, pp. 2798–2805, 2002.

[2] I. T. Young and L. J. van Vliet, "Recursive implementation of the Gaussian filter," *Signal Processing*, vol. 44, no. 2, pp. 139–151, 1995.

[3] M. Unser, "Fast Gabor-like windowed Fourier and continuous wavelet transforms," *IEEE Signal Processing Letters*, vol. 1, no. 5, pp. 76–79, 1994.

[4] A. Bernardino and J. Santos-Victor, "Fast IIR isotropic 2-D complex Gabor filters with boundary initialization," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3338–3348, 2006.

[5] B. Triggs and M. Sdika, "Boundary conditions for Young-van Vliet recursive filtering," *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 2365–2367, 2006.

[6] J.-M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer, "Fast anisotropic Gauss filtering," in *Proceedings of the 7th European Conference on Computer Vision (ECCV '02)*, pp. 99–112, Springer, 2002.

[7] C. H. Lampert and O. Wirjadi, "An optimal nonorthogonal separation of the anisotropic Gaussian convolution filter," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3501–3513, 2006.

[8] S. Y. M. Lam and B. E. Shi, "Recursive anisotropic 2-D Gaussian filtering based on a triple-axis decomposition," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1925–1930, 2007.

[9] V. Ulman, "Arbitrarily-oriented anisotropic 3D Gaussian filtering computed with 1D convolutions without interpolation," in *Proceedings of the 8th Conference on Signal Processing, Computational Geometry and Artificial Vision (ISCGAV '08)*, pp. 56–62, World Scientific and Engineering Academy and Society, Stevens Point, Wis, USA, 2008.

[10] S. W. Smith, *Digital Signal Processing: A Practical Guide for Engineers and Scientists*, Elsevier, Newnes, Australia, 2003.

[11] L. J. van Vliet, I. T. Young, and P. W. Verbeek, "Recursive Gaussian derivative filters," in *Proceedings of the 14th International Conference on Pattern Recognition (ICPR '98)*, vol. 1, p. 509, IEEE Computer Society, 1998.

[12] T. S. Lee, "Image representation using 2D Gabor wavelets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 959–971, 1996.

[13] O. Jun, B. Xiao-Bo, P. Yun, M. Liang, and L. Wei, "Automatic facial expression recognition using Gabor filter and expression analysis," in *Proceedings of the International Conference on Computer Modeling and Simulation (ICCMS '10)*, vol. 2, pp. 215–218, January 2010.