FILTERING WITH ANISOTROPIC 3D GABOR FILTER BANK EFFICIENTLY COMPUTED WITH 1D CONVOLUTIONS WITHOUT INTERPOLATION

Vladimír Ulman
Centre for Biomedical Image Analysis
Faculty of Informatics, Masaryk University
Botanická 68a, Brno, 602 00
Czech Republic
email: xulman@fi.muni.cz

ABSTRACT
We present an efficient spatial computational scheme designed for the convolution with a bank of 3D Gabor filters. The scheme covers most of the state-of-the-art principles. In particular, we start the paper with the staged approach to computing convolution with Gabor filter. It consists of three stages: modulation, IIR Gaussian filtering and demodulation. Two realizations of the IIR Gaussian filtering are considered and discussed: the method of Lam and Shi. Both methods allow for multiplications and additions per pixel, and the extension pert and Wirjadi, which is efficient in terms of number of operations per pixel is compensated for by applying efficient convolutions with up to four 3D Gabor filters simultaneously. The formal derivation and justification of new constraints as well as experimental results are presented.

KEY WORDS
Gabor filtering, bank filtering, filter design.

1 Introduction and Motivation
We describe a method for an efficient filtering with a bank of Gabor filters, by which we understand filtering certain input image with multiple regularly tuned Gabor filters in a way that requires less operations per pixel, multiplications or additions, than filtering it separately, filter by filter.

We focused on bank filtering in the optical flow estimation. Though, there are many other tasks in vision and image processing, such as texture analysis [1], image segmentation [2], edge detection [3], data compression or image enhancement [4] that also use oriented Gabor filters. Anisotropic filters may be required when processing images with different resolution in some axis such as microscopy images [5]. We are concerned with the filtering-based 3D (2D+t) methods of Heeger [6] and Fleet and Jepson [7]. These methods have had traditionally good accuracy but high computational demand [8]. Having fast and accurate method for convolving with densely populated oriented anisotopic filter bank at hand efficiently, the accuracy may increase while the computational load may decrease.

Motion detection imposes specific Gabor’s parameter setting [9, 10]. The 3D image is regarded as a stack of spatial 2D image planes stacked along the temporal z-axis. For the Gaussian envelope of a Gabor filter we consider the spatial orientation of the filter, which is given by an azimuth angle $\alpha_G$ in the xy-plane, and a temporal tilt $\beta_G$, which gives the deviation angle from the xy-plane, see Figure 1. Similarly, we define $\alpha_S$ and $\beta_S$ for a direction of the sine component of the filter. The typical detection setup is then $\alpha_G = \alpha_S$ and $\beta_G = \beta_S + \pi/2$. The $\alpha_G$ describes the spatial direction of a motion and $\beta_G$ corresponds to the velocity of it. Throughout this paper, we assume for every filter in the bank that $\alpha_G = \alpha_S$ while we leave $\beta_G$ and $\beta_S$ mutually unrelated but fixed. The bank consists of even number $2 + 2n$ filters with orientations $\alpha_G$ regularly spanning across the interval $[0, \pi]$ in steps of $\pi/(2+2n)$ radians, see Figure 1c). We assume that $\alpha_G = 0$ rad and $\alpha_G = \pi/2$ rad filters are always present in the bank. Such restrictions are probably common to many other fields as well. Especially, we don’t put constraint on the equality of Gaussian’s sigma allowing for anisotropy nor we limit spatial orientations to only along x,y and z axes allowing to use arbitrarily oriented anisotropic filters in a bank.

In the next section, we give definitions of filters, brief background and analysis. We continue with the section...
We define an oriented anisotropic complex 3D Gabor filter. Results and conclusion are given in the last two sections. Next section summarizes the whole convolution algorithm. Experimental results and conclusion are given in the fifth section. The fourth and most important section identifies common computing parts in a group of Gabor convolutions and shows how to benefit from it. Supplementary conditions to obtain an efficient filtering are given in the fifth section. Next section summarizes the whole convolution algorithm. Experimental results and conclusion are given in the last two sections.

## 2 Oriented Gabor Filtering

We define an oriented anisotropic complex 3D Gabor filter \(G(\vec{x}; \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \alpha_S, \beta_S, \omega) = \frac{1}{(2\pi)^{3/2} |C|^{1/2}} \cdot e^{-\frac{1}{2} \vec{x}^T C^{-1} \vec{x}} \cdot e^{jW\vec{x}}\) (1) where \(\vec{x} = (x, y, z)^T\) is a column vector giving position in the image, \(j\) is the complex unit, \(C\) is a symmetric 3 \(\times\) 3 Gaussian covariance matrix and \(W = [w_x, w_y, w_z]\) is a 1 \(\times\) 3 frequency tuning matrix. As we shall see later, it is important for the bank filtering to understand the content of both matrices. Unfortunately, the explicit definition of \(C\) would take too much space. We rather explain the way one may arrive to the matrix \(C\): \(C^{-1} = R^T \text{diag}(1/\sigma_x^2, 1/\sigma_y^2, 1/\sigma_z^2) R\) with \(R^{-1} = \begin{bmatrix} \cos(\alpha_G) & -\sin(\alpha_G) & 0 \\ \sin(\alpha_G) & \cos(\alpha_G) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\beta_G) & 0 & -\sin(\beta_G) \\ 0 & 1 & 0 \\ \sin(\beta_G) & 0 & \cos(\beta_G) \end{bmatrix}\) (2) \(R\) defines spatial rotation by \(\alpha_G\) and then temporal rotation by \(\beta_G\) along newly positioned y-axis. The matrix \(W\) amounts to \(\omega \cdot [\cos(\alpha_S) \cos(\beta_S), \sin(\alpha_S) \cos(\beta_S), \sin(\beta_S)]\) (3) The state-of-the-art method for computing a convolution with anisotropic Gabor filter, according to [11], is a method we will call staged as the process is split into three stages: modulation, Gaussian filtering, demodulation. The middle part is computed with Young et al. IIR filter [12], optionally equipped with improved boundary handling according to Triggs and Sdika [13]. However, the Gaussian filtering is still bound only to x, y and z axes.

For an arbitrarily oriented 3D Gaussian, two approaches have emerged quite recently. The one by Lampert and Wirjadi [14, 15] uses cascade of three oriented 1D IIR convolutions. Every convolution runs along certain direction vector. In this case, the first vector is always \((1, 0, 0), \) the second \((x_1, 1, 0)\) and the third \((x_2, y_2, 1)\) with \(x_1, x_2, y_2 \in R\). If some of \((x_1, x_2, y_2) \not\in Z\) (not an integer number), one or both convolutions require the use of interpolation because the direction vector gets off the image pixel grid, see Figure 2. The second approach is an extension of Lam and Shi [16, 17] in which cascade of six oriented 1D IIR convolutions is used. In this case, all vectors \((x_1, y_1, z_1), \ldots, (x_6, y_6, z_6)\) are with \(x_1, \ldots, 6; y_1, \ldots, 6, z_1, \ldots, 6 \in Z\). This forces convolutions to always stay on the image grid avoiding the need for interpolation.

The interpolation causes positional variability of responses of the former method. The variability manifests itself in the convolution results with the same filter on fixed-size blank images with differently positioned sample impulses. When the convolved images are translated such that original impulse positions become aligned, the convolution result images become registered in this way, we can compute a sum over every spatial coordinate of squared differences between pixel values and mean pixel value. The sum is then divided by the sum of squared means (total energy of the mean impulse) and \(10\log()\) is computed from it to express in dB (deciBell). The Lampert and Wirjadi method for 3D Gaussians produces mostly variability of -20dB while its accuracy is mostly better than -40dB [17]. Thus, the error is two orders of magnitude higher than its accuracy, the error is more dominant. The other method does not suffer from this.

Another criterion is the number of operations per pixel (or memory accesses if you will), ops/px, required for 3D filtering. Since Gabor and Gaussian filters are linear, only multiplications and additions with real numbers are considered. Both operations are regarded as equally demanding. In this perspective, the Young et al. IIR filter does 3, and 4, multiplications and 3, and 3, additions in the forward, and backward respectively, directions. Together it requires 13 ops/px. Modulation, because it turns input real image into complex one, requires 2 ops/px. Demodulation needs 6 ops/px. A complex IIR 1D Gabor filter requires 49 ops/px on real input and 50 ops/px on complex input image [12, 11].

Both approaches need to modulate, apply Gaussian filter both on real and imaginary parts and finally demodulate. Lampert and Wirjadi uses both linear (3 ops/px) and bilinear (7 ops/px) interpolations in their cascade. These convolutions are equal to shearing the image first, filtering it and shearing back [15]. Such oriented Gabor filtering needs \(2 + 2(13 + 3 + 3 + 7 + 13 + 7) + 6 = 126\) ops/px. If better interpolation technique is used, the number of ops/px will increase. Lam and Shi in 3D uses six 1D convolutions with no interpolation yielding \(2 + 2(6 \cdot 13) + 6 = 164\) ops/px. The latter method trades positional invariability for...
an increased number of operations per pixel.

In case when four 3D Gabor filters are to be computed the more efficient method of the two needs 504 ops/px. We will show in this paper that we can achieve this number as well when computing the four filters simultaneously and using the position invariant method, which otherwise uses 656 ops/px. The key of our approach is to share some 1D convolutions that are common to all four filtering cascades. This is possible because the method with interpolations uses less ops/px and is claimed to be optimal [15] implying some redundancy in the other method.

3 Separability of Gabor filter

We will drop the constant fraction in equation (1) in this section and rewrite $G(\vec{x}; \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \alpha_s, \beta_s, \omega) = e^{-\frac{1}{2} \vec{x}^T C^{-1} \vec{x}} e^{jW \vec{x}}$.

The expression for convolution becomes

$$O(\vec{y}) = \sum \bar{I}(\vec{x}) e^{-\frac{1}{2} (\vec{y} - \vec{x})^T C^{-1} (\vec{y} - \vec{x})} e^{jW (\vec{y} - \vec{x})}.$$  \hspace{1cm} (5)

where $I(\vec{x})$ is input image’s real pixel value and $O(\vec{y})$ is output image’s complex pixel value. The staged scheme splits the frequency component:

$$O(\vec{y}) = e^{jW \vec{y}} \left[ \sum \bar{I}(\vec{x}) e^{-jW \vec{x}} \cdot e^{-\frac{1}{2} (\vec{y} - \vec{x})^T C^{-1} (\vec{y} - \vec{x})} \right].$$ \hspace{1cm} (6)

Similarly to [17], we introduce new coordinate system given by six base vectors $\vec{b}_i = (a_{i1}, a_{i2}, a_{i3})^T, i = 1, \ldots, 6, a_{i,k} \in Z$. We can establish the following $3 \times 6$ transition matrix $A$ between $\vec{x}$ and $\vec{u} = (u_1, \ldots, u_6)^T$:

$$\vec{x} = A \cdot \vec{u} = [\vec{b}_1 \vec{b}_2 \vec{b}_3 \vec{b}_4 \vec{b}_5 \vec{b}_6] \cdot \vec{u}.$$ \hspace{1cm} (7)

Let $E = diag(\sigma_1^2, \ldots, \sigma_6^2)$ be the diagonal covariance matrix of some Gaussian kernel in the 3D separable along $\vec{b}_i$. Consider the development of the argument of the Gaussian’s separability constraint:

$$\vec{u}^T E^{-1} \vec{u} = (A^{-1} \vec{x})^T E^{-1} (A^{-1} \vec{x}) = \vec{x}^T (AEA^T)^{-1} \vec{x}.$$ \hspace{1cm} (8)

The right-hand-side of equation explains the shape of the $\vec{b}_i$-separable 3D Gaussian in the context of Cartesian coordinate system. We observe that $C = AEA^T$, which can be further developed, following [16], into the Gaussian separability constraint:

$$C = \sum_{i=1,\ldots,6} \sigma_i^2 \begin{bmatrix} \vec{b}_i^T \vec{b}_i \end{bmatrix}.$$ \hspace{1cm} (9)

Since $C$ is the Gaussian’s covariance matrix, by the definition, it is symmetric as such. It is easy to show that the right-hand-side of eq. (9) is symmetric as well. For the eq. (9) to hold, it is enough to compare diagonals and both upper triangles, which constitutes six linear equations with $\sigma_1^2, \ldots, \sigma_6^2$ unknowns because $C$ was given and $\vec{b}_1, \ldots, \vec{b}_6$ were fixed. The left-hand-side of the such system is $\bar{C}$ and $\bar{E}$, and the right-hand-side is $[c_{11}, c_{22}, c_{33}, c_{12}, c_{13}, c_{23}]^T$. By solving it, we obtain correct $\sigma_i^2$ for the filter. However, the solution is not always guaranteed (when $\sigma_i^2 < 0$). Hence, proper initial base vectors $\vec{b}_i$ must be used with respect to the $C$. We will return to the selection of base vectors later in the section 5.

The equation (6) changes in the new coordinate system into

$$O(A\vec{u}) = e^{jW A \vec{u}} \cdot \left[ \sum \bar{I}(A\vec{x}) e^{-jW \vec{x}} \cdot e^{-\frac{1}{2} (\vec{y} - \vec{x})^T C^{-1} (\vec{y} - \vec{x})} \right].$$ \hspace{1cm} (11)

Finally, after computing the $WA = [w_1, \ldots, w_6]$, a $1 \times 6$ matrix, we see the same Gabor filter now clearly separated along the base vectors $\vec{b}_i$.

The oriented anisotropic 3D Gabor filter can be computed either using the staged scheme given in eq. (11) or using a cascade of six oriented 1D Gabor filters, each $i$-th filter is given by the triple $(\vec{b}_i, \sigma_i, w_i)$ with Gaussian’s sigma $\sigma_i$ and frequency $w_i$ both applied along a direction $\vec{b}_i$. A mixture is also possible: separate 1D Gabor filters are each computed with the staged scheme. Note the order of convolutions in the cascade is always arbitrary. Also
note that stages in equations (6) and (11) are computationally aligned. Hence, the first and the last stage in eq. (11) can be easily computed in the original Cartesian coordinates with \([w_x, w_y, w_z]\), just like in eq. (6), instead of with \([w_1, \ldots, w_6]\) along \(\hat{b}_1, \ldots, \hat{b}_6\). From the numbers of operations per pixel in the section 2 we see that the staged scheme is more efficient.

Nevertheless, we will set \(\vec{v} = (v_1, \ldots, v_6)^T\) and rewrite the convolution expression of the equation (11) as follows:

\[
O(A\vec{u}) = e^{j[w_1v_1 + \ldots + w_6v_6]} \left[ \sum_{u_6} \cdots \sum_{u_1} \left[ I(A\vec{u}) \cdot e^{-j[w_1\ldots, w_6][u_1, \ldots, u_6]^T} \right] \right. \\
\left. \cdot e^{-j[u_1u_2 + \ldots + u_6u_6]diag(1, \ldots, 1/6)[v_1 - u_1, \ldots, v_6 - u_6]^T} \right] \\
= e^{j[w_1v_1 + \ldots + w_6v_6]} \\
\cdot e^{-j[w_1u_1 + \ldots + w_6u_6] diag(1, \ldots, 1/6)} \\
\cdot e^{-j(w_1u_1^2 + \ldots + w_6u_6^2)} \\
\cdot e^{-j(w_1u_1 + \ldots + w_6u_6)} \\
\cdot e^{-j(w_2u_2 + w_3u_3)} \\
\cdot e^{-j(w_4u_4 + w_5u_5 + w_6u_6)} e^{-\frac{(w_1 - u_1)^2}{\sigma_1^2}} \\
\cdot e^{-\frac{(w_2 - u_2)^2}{\sigma_2^2}} \\
\cdot e^{-\frac{(w_3 - u_3)^2}{\sigma_3^2}} \\
\cdot e^{-\frac{(w_4 - u_4)^2}{\sigma_4^2}} \\
\cdot e^{-\frac{(w_5 - u_5)^2}{\sigma_5^2}} \\
\cdot e^{-\frac{(w_6 - u_6)^2}{\sigma_6^2}} \\
(12)
\]

Staged convolution is further parted into three sections. Basically, the modulation stage is split and each part of it is postponed just at the beginning of respective section.

The computation of such cascade is illustrated in the flow chart in Figure 3. We aim to share the first section, \(\sum_{u_1}\), between four carefully selected Gabor filters from the filtering bank. The computation is then branched. The second section, \(\sum_{u_2}\), is shared within both pairs, each pair uses its own parameters \((\hat{b}_2, \sigma_2, w_2), (\hat{b}_3, \sigma_3, w_3)\). Branches of both pairs are branched again and so the last section is computed for each filter separately. We will continue the paper with justifying constraints that allow for this shared cascade scheme.

### 4 Similarity between Gabor Filters

We noted in the introduction that the filtering bank contains regularly oriented filters and gained an insight in Figure 1 that the spatial orientations, besides the two fixed filters \(\alpha_G = 0\) rad and \(\alpha_G = \pi/2\) rad, shall be symmetric around the \(x = 0\) plane.

Consider the four filters from Figure 1 given with \(\alpha_G = \alpha_1, \alpha_2, \alpha_4\) and \(\alpha_5\). Note that we assume \(\alpha_S = \alpha_G\), the remaining six parameters of a Gabor filter are constant within the four (and usually within the entire bank). The regularity of spatial orientations in the bank establishes some filter symmetries defined through the equalities in the left column of Table 1 (the last two rows in the table are in fact identical). We are especially interested in the relation between the “\(y\)-symmetric” pairs \(\alpha_1, \alpha_5\) and \(\alpha_2, \alpha_4\) and the “\(x\)-symmetric” pair \(\alpha_1, \alpha_2\).

We had analyzed explicit expressions of the Gaussian covariance matrix

\[
C = \begin{bmatrix}
\alpha_{1} & c_{12} & c_{13} \\
\alpha_{2} & c_{22} & c_{23} \\
\alpha_{3} & c_{32} & c_{33}
\end{bmatrix},
(13)
\]
to which we arrived following the description in the section 2, and of the frequency matrix \(W = [w_x, w_y, w_z]\), which is given explicitly in equation (3). Note that \(C\) and \(W\) fully describes the 3D Gabor filter. During the analysis, always a pair of symmetric filters, in the sense of Table 1, was considered. We substituted the relevant equalities for sine and cosine and computed the symmetry constraints.
cosine from Table 1 into the covariance and frequency matrices of the first filter in the pair and compared them to those of the second filter in the pair. The second column from the left in Table 2 gives \( C \) and \( W \) of the first filter after substitutions. The difference in covariance matrices suggests how one would have to change all base vectors of the second filter’s Gaussian so that it would actually realize the first filter’s Gaussian. We examined the frequency matrices similarly. It resulted in the filter modification rules which modify base vectors, the matrix \( A \) in eq. (7), and the frequency matrix \( W \). They are summarized in the third column of the table.

The modification rules are crucial for the proposed method as they show us how to convert between Gabor filters under certain symmetry. In other words, once \( A, \sigma_i \) and \( W \) of one Gabor filter in the symmetry pair are known, parameters for the other filter in the pair can be obtained by applying the rules on \( A \) and \( W \). \( \sigma_i \) are left unchanged. Utilizing the selected symmetries, we only need to establish parameters of the filter \( \alpha_1 \). The parameters for the other three filters can be obtained just by applying the rules.

For symmetric filters, we seek any base vector that leads to the same triple \((\vec{b}_i, \sigma_i, w_i)\) before and after application of the modification rules. Such base vectors yield exactly the same convolution triples in cascades of the symmetric filters. If these convolutionals are shifted to the beginning of all cascades, we may compute them only once and share the intermediate results between the cascades. Note that base vectors (eq. (7)) determine sigmas (eq. (10)) and frequencies (below eq. (11)).

\[
\begin{align*}
\vec{b}_i &= \{w_x, w_y, w_z, \vec{b}_i\}, \\
W &= \begin{bmatrix}
\cos(\alpha_2) \\
\sin(\alpha_2)
\end{bmatrix} \\
C &= \begin{bmatrix}
\sigma_1^2 & 0 \\
0 & \sigma_2^2
\end{bmatrix}
\end{align*}
\]

In the most right column in Table 2 we introduced base vector masks for every modification rule. Sufficing any of the masks given for the respective modification rule is mandatory for the base vector not to be changed by it. Sufficing any of the masks from the \( C \)-related and \( W \)-related given for respective filter symmetry is sufficient condition. The intersection of masks for given filter symmetry suggests sort of base vectors optimal for the shared cascade scheme.

### 4.1 Example of an analysis

We will give an example of the reasoning we have just described. Let’s suppose we are investigating the “\( \alpha \)-symmetry” between \( \alpha_1 \) and \( \alpha_2 \) filters.

Consider the covariance matrix \( C \) in eq. (13) for the filter \( \alpha_1 \). We focus on system in equation (10) in which every row corresponds to eq. (9), for example the last but one row comes from

\[
c_{13} = \sigma_1^2 a_{1,1}^2 + \ldots + \sigma_6^2 a_{6,1} a_{6,3}.
\]

We assume the system has some solution \( \sigma_i^2, i = 1, \ldots, 6 \). The rows were established in some certain fixed order. The matrix \( C \) in the first row of Table 2 shows that \( C \) for \( \alpha_2 \) consists of the same values. Hence, the right-hand-sides of systems (10) for \( \alpha_1 \) and \( \alpha_2 \) differ only in the order of rows. Since there is always the same combination of base vectors’ elements in the same row of the \( 6 \times 6 \) matrix, changing the order of elements in all base vectors \( \vec{b}_i, i = 1, \ldots, 6 \), will produce the same rows but in different order. In this case, swapping \( a_{i,1} \) and \( a_{i,2} \), in the base vectors, what is the trick of the modification rule, re-arranges rows of the \( 6 \times 6 \) matrix to the same order. We arrived to the linear system which differs from the original one only in the order of rows. From linear algebra we know that the solution was kept and was re-ordered as well. Thus, the same \( \sigma_i \) is bound with original \( \vec{b}_i \) as well as with its “\( \alpha \)-symmetric” sibling. The “\( \alpha \)-symmetric” Gaussians differ in the row order in the linear system (10) what is a consequence of swapped \( x \) and \( y \) elements of base vectors between the two Gaussians, sigmas are kept unchanged.

Understanding the change in frequencies is simpler. Assume

\[
\begin{align*}
W_{\alpha_2} &= \omega \cdot [\cos(\alpha_2) \cos(\beta_3), \sin(\alpha_2) \cos(\beta_3), \sin(\beta_3)].
\end{align*}
\]

We seek a description of \( W_2 \) from \( W_1 \). By substituting from Table 1,

\[
\begin{align*}
W_{\alpha_2} &= \omega \cdot [\sin(\alpha_1) \cos(\beta_3), \cos(\alpha_1) \cos(\beta_3), \sin(\beta_3)].
\end{align*}
\]

We see that \( W_{\alpha_2} \) turns into \( W_{\alpha_1} \) by swapping \( w_x \) with \( w_y \).

We continue with the example and explain how the masks were established for the “\( \alpha \)-symmetry”. We seek to find such a vector \( \vec{b}_i^{\alpha_1} \) that, after applying the modification rule, will change into \( \vec{b}_i^{\alpha_2} = \vec{b}_i^{\alpha_1} \), for some \( i = 1, \ldots, 6 \).

Since the rule, in this particular case, swaps the \( x \) and \( y \) elements of a base vector, we are looking for some \( a_{i,1}, a_{i,2} \in \mathbb{Z} : a_{i,1} = a_{i,2} \). Among the infinite number of possible solutions we have chosen only \( a_{i,1} = a_{i,2} = 0 \) or \( a_{i,1} = a_{i,2} = 1 \). Regarding the value of \( w_i^{\alpha_2} = w_i^{\alpha_1} \), associated with some \( i = 1, 0, 6 \), both frequencies are an inner product of \( w_i^{\alpha_1} = (w_x, w_y, w_z, a_{i,1}, a_{i,2}) \) for the \( \alpha_1 \) filter and \( w_i^{\alpha_2} = (w_y, w_x, w_z, a_{i,1}, a_{i,2}) \) for the second filter \( \alpha_2 \). After putting together it evaluates to

\[
\begin{align*}
w_x(a_{i,1}^2 - a_{i,2}^2) + w_y(a_{i,1}^2 - a_{i,1}^2) + w_z(a_{i,3}^2 - a_{i,3}^2) = 0.
\end{align*}
\]

Since we don’t want to put constraints on frequencies, the only solution is to force the three parentheses to approach zero. This necessitates that \( \vec{b}_i^{\alpha_1} \) and \( \vec{b}_i^{\alpha_2} \) must be the same except for swapped \( x \) and \( y \) elements. This results in the same mask as above. Note that in general both constraints are only mandatory when considered separately. The intersection of the masks must be evaluated.

### 4.2 Practical considerations

An outcome of the filter masks is that the only base vector common to the all four filters is \( \vec{b}_1 = (0, 0, 1) \). Hence, \( w_1 = w_z \) and the initial modulation can be computed in the Cartesian coordinate system. Similarly, we observe that “\( y \)-symmetric” filter pairs, e.g. \( \alpha_1 \) and \( \alpha_3 \), may share vectors \( \vec{b}_2 = (0, k, l) \) and \( \vec{b}_3 = (0, m, n) \) for some \( k, l, m, n \in \mathbb{Z} \). This forces both \( w_2 \) and \( w_3 \) to be only a linear combination of \( w_y \) and \( w_z \). Provided \( w_1 = w_z \) was used in the
initial modulation, we can replace the second modulation \(b_{2,3}w_{2,3}\), from Figure 3, with the modulation with \(w_y\) in the Cartesian coordinates — we additionally modulate to achieve a contemporary modulation of an input image with both frequencies \(w_y\) and \(w_z\). The last modulation in each filter’s own branch then completes the process by modulating with \(w_z\). Another resume is that it is better to group \(y\)-symmetric Gabor rather than \(\alpha\)-symmetric ones. This is due to the fact that the \(y\)-symmetry has less restrictive masks on base vectors, see Table 2. This may allow for more shared convolutions be available enabling greater reduction in the overall number of required operations per pixel.

We have initially required that \(\alpha_G = \alpha_S\) and \(\beta_G, \beta_S\) be fixed in the filtering bank. Disabling one or both constraints may considerably hamper the generality of the vector masks. This, in turn, may severely limit the number of base vectors that comply with new masks. Consider we require \(\alpha_G = \alpha_S + \gamma, \gamma \neq 0\) rad. The frequency \(W\)-related rules will probably differ from the rules for base vectors, the \(C\)-related, even if we consider different filter pair symmetries. More modification rules tend to give rise to more different base vector masks. Clearly, the more masks there are, and consequently the less base vectors comply with them, the less 1D convolutions can be shared.

5 Base Vectors for Gaussian

It is sad but true that we don’t know any constructive algorithm that would lead us directly to certain six-tuple of base vectors, not even for a plain oriented 3D Gaussian filter. However, there is such when the position variant method of Lampert and Wirjadi [15] is used. We have shown previously for the oriented 3D Gaussian that even a “try some base vectors and see” algorithm with some heuristics can achieve nice computational times [17]. Hence, we used such algorithm again. It is still to be added that when establishing the six-tuple of base vectors we can focus only on the Gaussian. This is so because the oriented frequencies \([w_1, \ldots, w_5]\) are matrix product \(W A\) — what ever different base vectors we use, the frequencies will always be defined and correct.

Basically, we need to find six base vectors, compactly stored in the matrix \(A\), such that:

a. \(a_{i,k} \in \{-2, -1, 0, 1, 2\}, i = 1, \ldots, 6 \land k = 1, 2, 3\),

b. equation (9) has solution such that \(\sigma_i^2 \geq 0, i = 1, \ldots, 6\),

c. gives highest accuracy possible,

d. has enough vectors that satisfy masks for the shared cascade scheme.

Comment on a. In order to keep the proposed method position invariant we need to keep any convolution always on the image pixel grid. The simplest way to achieve it is to allow only integer elements in directional vectors. The length of a base vector \(\tilde{b}_i = (a_{i,1}, a_{i,2}, a_{i,3})\) also defines a unit length along its direction as \(|\tilde{b}_i| = \sqrt{a_{i,1}^2 + a_{i,2}^2 + a_{i,3}^2}\), see Figure 4. When two or three vector elements are not zero, the unit length will never get smaller than \(\sqrt{2}\). Filtering with Gaussian along the direction \(\tilde{b}_i\) compensates increased unit length by proceeding with \(\sigma_i/|\tilde{b}_i|\). This equals filtering with Gaussian with sigma \(\sigma_i\) along \((1, 0, 0)\) and rotating afterwards. As a consequence, the filtering uses always smaller sigma compared to the nominal \(\sigma_i\). We use the widely adopted IIR filters which are known to provide better accuracy with higher \(\sigma\) [12]. Hence, it is desirable to keep unit lengths as small as possible. This is why we have chosen only zeros and ones in the example of analysis in the previous section.

Comment on b. Even though we have limited the range of base vector’s elements to only five values, there still exists vast amount of their possible combinations. Given some base vector six-tuple we first pass it to the Gaussian separability constraint given in equation (9). This equation is expanded into the equation (10) and handled to the LAPACK software to find solution. However, many six-tuples provide some solution in which \(\sigma_i^2 \geq 0, i = 1, \ldots, 6\), see Figure 6. These are therefore handed over to a small convolution test to determine the one with lowest error.

Comment on c. We mentioned the use of fast IIR filter. It is, however, an approximation to exact Gaussian. Thus, different combinations of base vectors, which give good solution to equation (9), achieve different levels of accuracy due to, we believe, different solutions, i.e. different \(\sigma_i\). To further distinguish between base vector six-tuples, small convolution test in which a sample impulse response is computed in reasonably small image and compared to an ideal computed impulse response. We choose vector six-tuple that gives the lowest NSE error below \(-40dB\).

We opted for the normalized squared error (NSE), essentially the error-to-signal ratio as in [11, 16], which gives the average squared magnitude of errors in the examined image \(T\) with respect to the total energy of the reference.
image $R$: 

$$\text{NSE}(T, R) = 10 \log_{10} \frac{\sum \hat{T}(\hat{x}) - R(\hat{x})}{\sum R(\hat{x})^2}. \quad (18)$$

The NSE, unlike the RMSE for instance, readily gives the magnitude of the error regardless the average intensity in the filtered image. The lower the number is, the smaller error it measured. Others, for reference, used absolute relative error [18] or maximum absolute error or RMSE [19, 12, 14, 15].

Owing to the orientations used in the filtering bank, we can expect all input 3D Gabor filters, besides the 0 rad and $\pi/2$ rad, to satisfy $0 \text{rad} < \alpha_G < \pi/4$ rad. The other three symmetric filters of $\pi/2 - \alpha_G, \pi/2 + \alpha_G, \pi - \alpha_G$ can be computed using the modification rules from parameters $(\tilde{b}, \sigma_1, m_i)$ derived from the one $\alpha_G$ filter. We have used some heuristics on the filters from this region to find out which base vectors appears more often in the good six-tuples. This helped to narrow the number of combinations.

Comment on d. Following the ideas of b, we may try to further reduce the number of vector six-tuples by preferring those complying with masks. For instance, set to $\tilde{b}_1 = (0, 0, 1)$ and combine pairs for $\tilde{b}_2$ and $\tilde{b}_3$ from these: 

- $(0, 1, 0)$, $(0, 1, 1)$, $(0, 2, 1)$, $(0, 1, 2)$, $(0, -1, 1)$, $(0, 1, 2)$, $(0, 1, -1)$, $(0, 2, -1)$, $(0, 1, -2)$.

6 Gabor Bank Filtering

Before we give the algorithm for 3D Gabor bank filtering we will first review the property vital to the presented approach.

The four symmetric Gabor filters, from Figure 1 and Table 1, can be efficiently computed based on equation (12) using the shared scheme as is illustrated in Figure 3. Assume we have already obtained the new coordinate system of $\tilde{b}$ encoded in the matrix $A$ accompanied with $\sigma_1$ and the basic frequency decomposition $W = [w_x, w_y, w_z]$ for the given filter $G(x; \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \omega)$. The filtering with $G(x; \pi/2 - \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \omega)$ can be done by using $W = [w_y, w_z, w_x]$ and $b_i = (b_{i1}, b_{i2}, b_{i3})$, $i = 1, \ldots, 6$. The filtering with $G(x; \pi/2 + \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \omega)$ uses $W = [w_y, w_z, w_x]$ and $b_i = (-b_{i1}, b_{i2}, b_{i3})$, $i = 1, \ldots, 6$. Finally, the filtering with $G(x; \pi - \alpha_G, \beta_G, \sigma_x, \sigma_y, \sigma_z, \omega)$ can be done using $W = [w_x, w_y, w_z]$ and $b_i = (-b_{i1}, b_{i2}, b_{i3})$, $i = 1, \ldots, 6$. The shared cascade scheme starts with the same convolution for the all four filters. The computation is then branched. In each branch it is continued with filtering with a pair of filters. We had an option to either pair the “$y$-symmetric” or “$\alpha$-symmetric” filters at this moment. We opted to group filters according to the “$y$-symmetry” because it poses less restrictive mask on common base vectors. Thus, we expect more common base vectors to be found what should improve the overall performance of the proposed shared scheme.

The algorithm for efficiently computing bank of $2 + 2n$ oriented anisotropic 3D Gabor filters:

1. filter with $G(x; 0, \beta_G, \sigma_x, \sigma_y, \sigma_z, 0, \beta_S, \omega)$
2. filter with $G(x; \pi/2, \beta_G, \sigma_x, \sigma_y, \sigma_z, \pi/2, \beta_S, \omega)$
3. set $i = 1$ and $s = \frac{\pi}{2}(n + 1)$
4. while $n \geq 2$ do
5. find A, $\sigma_1$, W for $G(x; i, \beta_G, \sigma_x, \sigma_y, \sigma_z, is, \beta_S, \omega)$
6. re-arrange the cascade triples so that it starts with $\tilde{b}_1 = (0, 0, 1)$
7. modulate input image with $\exp(-jw_1x)$
8. filter it with 1D oriented Gaussian with sigma $\sigma_1$ along the direction $(0, 0, 1)$
9. branch1: make a copy of an intermediate result
10. re-arrange the cascade so that it continues with $\tilde{b}_2 = (0, k, l)$, $\tilde{b}_3 = (m, n, 0)$ for $k, l, m, n \in Z$
11. modulate with $\exp(-jw_2y)$
12. apply Gaussians with sigma $\sigma_2, \sigma_3$ along directions $(0, k, l)$, $(0, m, n)$
13. branch2: make a copy of an intermediate result
14. modulate with $\exp(-jw_3x)$
15. apply Gaussians with sigma $\sigma_4, \sigma_5, \sigma_6$ along directions $\tilde{b}_1, \tilde{b}_3, \tilde{b}_6$
16. demodulate with $\exp(w_x'y + w_y'y + w_z'y)$
17. revoke the copy from 13
18. modulate with $\exp(+)w_2x$
19. apply Gaussians along directions $(-a_{4,1}, a_{4,2}, a_{4,3})$, $(-a_{5,1}, a_{5,2}, a_{5,3})$, $(-a_{6,1}, a_{6,2}, a_{6,3})$ with sigma $\sigma_4, \sigma_5, \sigma_6$
20. demodulate with $\exp(-w_x'y + w_y'y + w_z'y)$
21. revoke the copy from 9
22. re-arrange the cascade so that it continues with $\tilde{b}_2 = (k, 0, l)$, $\tilde{b}_3 = (m, 0, n)$ for $k, l, m, n \in Z$
23. modulate with $\exp(-jw_4y)$
24. apply Gaussians with sigma $\sigma_2, \sigma_3$ along directions $(0, k, l)$, $(0, m, n)$
25. branch3: make a copy of an intermediate result
26. modulate with $\exp(-jw_5x)$
27. apply Gaussians along directions $(a_{4,2}, a_{4,1}, a_{4,3})$, $(a_{5,2}, a_{5,1}, a_{5,3})$, $(a_{6,2}, a_{6,1}, a_{6,3})$ with sigma $\sigma_4, \sigma_5, \sigma_6$
28. demodulate with $\exp(w_x'x + w_y'y + w_z'z)$
29. revoke the copy from 25
30. modulate with $\exp(+)w_2x$
31. apply Gaussians along directions $(-a_{4,2}, a_{4,1}, a_{4,3})$, $(-a_{5,2}, a_{5,1}, a_{5,3})$, $(-a_{6,2}, a_{6,1}, a_{6,3})$ with sigma $\sigma_4, \sigma_5, \sigma_6$
32. demodulate with $\exp(-w_x'y + w_y'y + w_z'y)$
33. increase $\ i = i + 1$ and $n = n + 2$
34. endwhile.
Note that the only approximations are in the IIR implementations of 1D Gaussians. The shared scheme is truly equivalent, in terms of accuracy, to convolving separately with filter after filter. There is also no interpolation involved allowing for the position invariance.

The performance in terms of operations per pixel depends on the established parameters in the step 5 of the algorithm. The step 6 requires that vector $(0,0,1)$ is present. The step 10 needs two vectors with zero in the x-element. The step 22 further requires presence of two vectors with zero in the y-element. This would be an optimal setup since under such circumstances the overall number of operations per pixel is $2+2\times13+2(6+4\times13+2(6+6\times13+6)) = 504$. However, it often happens that either in step 10 or 22 only one required vector is present. The overall number of ops/px then increases to 530 ops/px (only 5% increase) as a result of lacking one shared convolution.

In case the initial vector $(0,0,1)$ is not present in the $A$ and/or $n$ is odd, the algorithm can be modified to use only one level of branching and to apply only the less restrictive “y-symmetry” on two Gabor filters. The number of operations per pixel depends on the number of base vectors present in the $A$ and matching the mask $(0,*,*)$. Sharing one, two or three convolutions between only two filters makes the shared scheme to require 312, 286 or 260 ops/px, respectively, in order to convolve with two filters simultaneously. The position variant method would require 252 ops/px.

7 Experimental Results

We tested 720 different filtering banks: bank size was 4, 6, 8, . . . , 22 filters sampling the interval $(0^\circ,180^\circ)$ not denser than 8.18°. Gaussian tilting was from 10° to 80° in steps of 10° and 9 combinations of sigmas were used with $\sigma_x=3,4,5$ and $\sigma_y=\sigma_z=1,\ldots,\sigma_x-1$.

The crucial step of the proposed scheme is selecting a new coordinate system through the base vectors $\vec{b}_i$. Clearly, they directly affect availability (eq. (9)), i.e. whether all filtering in a bank can be conducted, and also sharing efficiency (eq. (19)) of the scheme for particular filtering task. We will show that they also determine accuracy (eq. (18)).

As we have noted earlier, base vectors directly influence all Gabor parameters: (de)modulation frequencies $w_i$ and Gaussian’s smoothing $\sigma_i$. The smoothing parameters are obtained from a solution of eq. (9). This may fail when solution cannot be found or the solution contains some $\sigma_i < 0$. Since the left-hand-side of eq. (9) encodes parameters of an input Gabor filter, changing some of base vectors is the only way to find valid solution, if it is possible. Figure 5 gives overview of filtering availability. Moreover, different combinations of base vectors that lead to valid solutions allow for different accuracies of a Gabor filter, Figure 6. The NSE error measure was used eq. (18) with theoretical impulse response of respective Gabor filter as a reference image $R$. In the following, we always use such base vectors that produce the lowest error for given Gabor filter.

The frequencies $w_i$ are product of matrix multiplication $WA$ and as such can be always computed. Both modulation and demodulation are pixel-wise complex exponential multiplications which can also be always computed. Thus, they don’t limit availability. The frequency-related portion of error is in the order of magnitude of CPU’s precision and does not influence overall error of convolutions significantly, see again Figure 6 and notice that the errors of Gaussian and Gabor filter are not differing significantly, what is also acknowledged in [12]. The magnitude of frequency doesn’t seem to change the accuracy as well. The availability and accuracy then seems to be solely driven by the selection of base vectors.

Figure 7 shows NSE error rates for different banks, computed as average error over all filters in a bank, for the proposed scheme and for the optimal approach. We consider convolution with filter by filter, each filter conducted with the staged scheme (eq. 6) with Gaussians according
Figure 7. The NSE error distribution for the proposed scheme (left) and for the optimal staged approach (right). Only the imaginary parts, the sine-component of complex Gabor filter, are compared in this figure. Darker grey level indicates smaller error.

Figure 8. The distribution of filtering efficiency in dependence on density of filters in a bank and their tilt.

Figure 9. The histogram of filtering efficiency.

To Lampard and Wirjadi, as the optimal approach. The proposed scheme is position invariant, relatively available for most filtering tasks and has slightly better results. We will now focus on the number of required operations per pixel.

We have shown that proper selection of base vectors permits convolution with certain Gabor filter and determines its accuracy. On top of it, the number of base vectors complying with symmetry masks implies the efficiency of the proposed scheme. The efficiency of a convolution with certain filter bank was computed as

\[
\frac{\text{total number of ops/px}}{\text{total number of ops/px optimally required}} \cdot 100\% 
\]

where, again, the filter by filter staged approach was considered to set the optimal requirement. The efficiency of the scheme is summarized in Figure 8 and 9. Roughly 75% from the 587 applicable filtering banks requires only up to 5% more operations per pixel than the optimal solution, 95% banks needs only up to 10% more. Note that the more filters in a bank are, the better efficiency is achieved. On the other hand, an increased efficiency in situations with small number of filters in a bank results in small overhead either since the total number of ops/px is small itself. For instance, a bank \( n = 1 \), \( \beta_G = 20^\circ = \beta_S + 90^\circ \) required 484 ops/px while optimally it would be 448 ops/px (difference 36 ops/px, efficiency 108%) whereas a bank \( n = 10 \), \( \beta_G = 70^\circ \) required 2744 ops/px and optimally 2716 ops/px (difference 28 ops/px, efficiency 101%).

Returning to the motion detection problem from the introduction, we demonstrate the convenience of having a bank with anisotropic filters. Suppose a spot of 4x4 pixels moving in direction (1,1) with velocity 1.41px/frame as in the left in Figure 10. We used three filtering banks: 8 filters, \( \beta_G = 35^\circ, \beta_S = -55^\circ, \omega = 2\pi/4, \sigma_x = A, \sigma_y = \sigma_z = B \) with sigma combinations A–B set as 2–2,2–1 and 1–1. Intensity profiles from the convolution results, pictured in the right in Figure 10, clearly show the difference how filters in the banks responded. Responses drop as the filter tuning is further away from the original motion but they are much better separated in the anisotropic filters.

8 Conclusion

We have presented the scheme that is able to convolve a bank of 3D anisotropic regularly tuned complex Gabor filters. We started by separating Gabor filter into six directions such that each filtering becomes position invariant. This allowed for some redundancy when separation of two filters was examined. We pointed out constraints to utilize this redundancy in order to maintain good computational demand. Finally, we arrived to a description of a scheme that does a convolution with Gabor bank efficiently according to these constraints.

We conclude that the proposed scheme is nearly optimal with increased accuracy as compared to the staged approach with Gaussian filtering according to Lampard and Wirjadi. In addition, we gained position invariant filtering. We demonstrated that overhead of up to 5% in required multiplications and additions per pixel is present in 3/4 of tested banks, mostly in banks with more than 15 filters where the computational demand is not negligible. On the other hand, the scheme was not able to convolve with 19% of tested banks due to inability to separate some filters in a bank. We tested banks with stepping up to 10° in spatial and temporal direction tunings.


Acknowledgment

I would like to thank my colleagues Michal Kozubek and, especially, David Svozil. This work has been supported by the Ministry of Education of the Czech Republic (Grants No. MSM0021622419, LC535 and 2B06052).
Figure 10. Left: Maximum intensity projection of 2D+t image shows a moving spot on a random background. Four 2D frames from this sequence are shown as well. Right: Intensity profiles drawn along the dashed line from filtering results of the three banks 2–2, 2–1 and 1–1 on ideal data (top) and data with noise (bottom). The magnitude of complex response is depicted.

References


