Narrow directional steerable filters in motion estimation

Dimitrios S. Alexiadis, George D. Sergiadis *

Aristotle University of Thessaloniki, Department of Electrical and Computer Engineering, Thessaloniki GR 54124, Greece

Received 2 March 2007; accepted 8 July 2007
Available online 11 August 2007

Abstract

We extend the mathematical analysis of previous work [M.T. Andersson, Controllable multi-dimensional filters and models in low-level computer vision, Ph.D. Thesis, Department of Electrical Engineering, Linköping University, Sweden, 1992] and we give rigorous, general, mathematical formulas for the construction of 3-D steerable directional cosine filters of arbitrary higher order. Furthermore, we present the mathematical analysis for the construction of arbitrary narrow, steerable directional quadrature pairs. Incorporating the “Donut Mechanism” of Simoncelli [E.P. Simoncelli, Distributed representation and analysis of visual motion, Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1993] and extending it for quadrature pairs, we present a unified theory and a simple algorithm for using the constructed filters to estimate the motion in image sequences. Based on simple theoretical analysis, we explain the advantages of using higher order filters. Experimental results on synthetic, realistic, and natural sequences verify the effectiveness of the main algorithm and our arguments.

© 2007 Elsevier Inc. All rights reserved.

Keywords: Computer vision; Fourier transform; Motion estimation; Steerable filters

1. Introduction

Estimation of optical flow in video sequences is an important, active research topic. It consists the base for many challenging tasks, ranging from passive scene interpretation to video coding, object tracking, visual surveillance systems, content-based video retrieval, and dynamic 3-D scene analysis [1,2]. Over the last 30 years, many motion estimation approaches have been proposed, while new ones appear. Most of them can be categorized into: (a) feature or correlation matching techniques [1,3], (b) differential (or gradient constraint) techniques [4–7], and (c) frequency or filter-based techniques [8–14].

Matching techniques typically use two or three consecutive frames, searching for the best match of small regions or features from one frame to the next. Frequency domain or filter-based approaches exploit the information in longer image sequences and are mainly based on the notion that motion is manifested as energy concentration along planes in the 3-D spatiotemporal spectrum. Neurophysiological evidence suggests that the human visual system probably works in a similar manner and many motion perception models are based on frequency domain considerations [7, 8, 12, 15, 16]. Moreover, experimental work [17] suggests that “spatiotemporal” approaches perform generally better than two-frame approaches.

Differential techniques are typically based on the assumption that image-intensity is conserved over time, which constraints the temporal derivative to zero. The number of frames needed depends on the size of the discrete 3-D filters, used to approximate the continuous derivatives [7]. Given an appropriate choice of derivative filters, Simoncelli [7] formalized a concrete relation between frequency domain and gradient-based approaches.

The use of spatiotemporal velocity-tuned Gabor filters for the extraction of motion was early proposed by [8–10]. Adelson and Bergen [8] performed local measurements of the “motion energy” from the sum of the squares of quadrature pairs, highlighting their importance for the elimination of symmetry and contrast dependencies of filter
responses. For the reduction of computational effort, the application of “steerable” filters [11,18–20,22] was proposed. Freeman and Adelson [18,19] elucidated an elegant theory for derivatives of Gaussians (DOG filters), that provide the interpolation schemes used to synthesize the response of a filter at an arbitrary orientation from the responses at some fixed orientations. Andersson [20] constructed 3-D directional cosine filters and also presented the mechanisms for the construction of quadrature pairs. He constructed filters of order $N \leq 3$, while Simoncelli [7] highlighted the importance of higher order directional filters for the computation of multiple motions, without however paying close attention to their steerability.

In the current work, we generalize the theory towards the construction of steerable directional cosine filters and quadrature pairs, of arbitrary higher order. Simultaneously, the “Donut Mechanism” of Simoncelli [7] is incorporated and extended to cover also the constructed quadrature pairs. The advantages of higher order filters are described and details of a simple filter-based motion estimation algorithm are given. The performance of the algorithm and the validity of our arguments are examined on both synthetic and real sequences.

2. Theoretical analysis

2.1. Steerable 3-D directional cosine filters

In this section, we generalize Andersson’s theory [20], constructing steerable directional cosine filters of higher order. Denote as $\omega = (\omega_x, \omega_y, \omega_t)$ a spatiotemporal frequency triplet and as $\bar{\omega} = \omega / |\omega|$ the corresponding unit-normalized frequency vector. A 3-D directional filter of order $N$, oriented along the unit vector $v$, in the frequency domain is given by

$$B_N^v(\omega) = G(|\omega|)(\bar{\omega} \cdot v)^N,$$

(1)

where $(\cdot)$ denotes vector inner product and $G(|\omega|)$ is an arbitrary radial function, which will be considered as equal to unity, since it does not affect the following analysis. Also, for the sake of simplicity, we omit $N$, wherever it is implied. The shape of such a filter in the frequency domain for $N = 0, 1, 2, 3$ is depicted in Fig. 1. It will be shown that such a filter can be steered (interpolated) at an arbitrary orientation $v$ from at least:

$$L(N) = \frac{(N+1)(N+2)}{2} \leq I(N)$$

(2)

![Fig. 1. Top-left diagram: The values of the coefficients $a_M[N]$ for $M = 1, \ldots, 8$. Each curve gives the coefficients $a_M[N]$ with respect to $N$, for the corresponding value of $M$. Remaining diagrams: The construction of the quadrature pair $Q_3^{even}(\omega)$, $Q_3^{odd}(\omega)$ from the cosine filters of orders $N \leq M = 3$ (see Eq. (21)). In this figure, the subscripts correspond to $M$ and the superscripts to $N$. For this illustration, we have used as $G(|\omega|)$ (see Eq. (1)) a 3-D Gaussian.](image-url)
linearly independent basis filters \( B_i(\omega) \) at orientations \( \mathbf{n}_i \), \( i = 1, 2, \ldots, I(N) \). Moreover, we derive the corresponding interpolation scheme. The values \( L(N) \) with respect to \( N \) are given in Table 1.

For notational simplicity, we have to introduce the vector:

\[
\mathbf{w}_N(\omega) = |\omega|^{-N} \left[ C_{pqr} \alpha^p \beta^q \gamma^r \right]_{p+q+r=N}^{T} \left[ \begin{array}{c} \alpha_1 \beta_1 \gamma_1 \\ \vdots \\ \alpha_N \beta_N \gamma_N \end{array} \right] \]

which is of length equal to \( L(N) \) and \( C_{pqr} = \frac{N!}{p!q!r!} \) represents the coefficients of the polynomial expansion:

\[
C_N \frac{x}{C_0} C_N \frac{x}{C_1} \cdots C_N \frac{x}{C_N} = \sum_{p+q+r=N} C_{pqr} \left( \frac{x}{C_0} \right)^p \left( \frac{x}{C_1} \right)^q \left( \frac{x}{C_N} \right)^r.
\]

**Example:** The vector \( \mathbf{w}_N(\omega) \) for \( N = 2 \) has length \( L(2) = 6 \) and it is defined as follows:

\[
\mathbf{w}_2(\omega) = |\omega|^{-2} \left[ \begin{array}{ccc} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \end{array} \right] = \sum_{p+q+r=2} C_{pqr} \alpha^p \beta^q \gamma^r.
\]

With \( \mathbf{n}_i = (n_{ix}, n_{iy}, n_{iz})^{T} \) denoting the orientation unit vector for the \( i \)-th basis filter, for each basis filter we obtain:

\[
B_i(\omega) = (\hat{\omega} \cdot \mathbf{n}_i)^N = |\omega|^{-N} \left( \omega_{ix} + n_{ix} \right)^N \left( \omega_{iy} + n_{iy} \right)^N \left( \omega_{iz} + n_{iz} \right)^N
\]

where \( k_{i,pqr} = n_{ix}^p n_{iy}^q n_{iz}^r \) and \( C_{pqr} = \frac{N!}{p!q!r!} \). The summation in the last term can be written as a vector inner product, using \( \mathbf{w}_N(\omega) \). For easier understanding, in equations where vector and matrix operations are involved, we also give vector and matrix dimensions:

\[
B_i(\omega) = \mathbf{k}_i \cdot \mathbf{w}_N(\omega),
\]

\[
1 \times 1 = (L \times L) \cdot (L \times 1),
\]

where \( \mathbf{k}_i = [k_{i,1}, k_{i,2}, \ldots, k_{i,N}] \) and has a length equal to \( L(N) \).

**Example:** The vector \( \mathbf{k}_i \) for \( N = 2 \) is

\[
\mathbf{k}_i = [n_{ix}^2, n_{ix} n_{iy}, n_{ix} n_{iz}, n_{iy}^2, n_{iy} n_{iz}, n_{iz}^2].
\]

Considering Eq. (6) for each \( i \leq I(N) \), we obtain the linear system:

\[
\mathbf{B}(\omega) = K \cdot \mathbf{w}_N(\omega),
\]

\[
I \times 1 = (L \times L) \cdot (L \times 1),
\]

where \( \mathbf{B}(\omega) = [B_1(\omega), B_2(\omega), \ldots, B_L(\omega)]^{T} \) is the vector containing all the basis filters \( B_i(\omega) \), while the \( i \)-th row of the matrix \( K \) is the vector \( \mathbf{k}_i \), namely: \( K = [\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_L]^{T} \). Solving the system of Eq. (8) for \( \mathbf{w}_N(\omega) \), we obtain:

\[
\mathbf{w}_N(\omega) = K^{-1} \cdot \mathbf{B}(\omega),
\]

\[
L \times 1 = (L \times I) \cdot (I \times 1),
\]

where \( K^{-1} \) is the pseudo-inverse of \( K \), since \( K \) is not necessarily square. The pseudo-inverse \( K^{-1} \) is the \( L \times I \) matrix, such that its product from left (or from right) with \( K \) gives the \( L \times L \) (or \( I \times I \)) identity matrix. In our implementation, it is calculated using the Matlab function \( \text{pinv} \).

Now consider the directional filter \( B_i(\omega) = (\hat{\omega} \cdot \mathbf{v})^N \) along the unit orientation \( \mathbf{v} = (v_v, v_v, v_v)^{T} \). Using the same arguments as previously, we obtain an equation similar to (6):

\[
B_i(\omega) = |\omega|^{-N} \sum_{p+q+r=N} C_{pqr} \alpha^p \beta^q \gamma^r = \mathbf{d}(\mathbf{v}) \cdot \mathbf{w}_N(\omega),
\]

\[
1 \times 1 = (1 \times L) \cdot (L \times 1),
\]

where \( d_{pqr}(v) = v_v^p v_v^q v_v^r \) and \( \mathbf{d}(\mathbf{v}) = [d_{pqr}(v)]_{p+q+r=N}^{T} \). Namely, \( d_{pqr}(v) \) and \( \mathbf{d}(\mathbf{v}) \) are described similarly as \( k_{i,pqr} \) and \( \mathbf{k}_i \), respectively.

From Eqs. (10) and (9), we conclude to:

\[
B_i(\omega) = \mathbf{d}(\mathbf{v}) \cdot K^{-1} \cdot \mathbf{B}(\omega) = \mathbf{t}(\mathbf{v}) \cdot \mathbf{B}(\omega) = \sum_{i=1}^{I(N)} t_i(\mathbf{v}) B_i(\omega),
\]

\[
1 \times 1 = (1 \times L) \cdot (L \times I) \cdot (I \times 1) = (1 \times I) \cdot (I \times 1),
\]

where \( \mathbf{t}(\mathbf{v}) = \mathbf{d}(\mathbf{v}) \cdot K^{-1} \) the interpolation vector, of length \( I(N) \). Eq. (11) states that a filter of order \( N \) at an arbitrary orientation \( \mathbf{v} \) can be linearly interpolated from the basis filters \( B_i(\omega) \), with weights \( t_i(\mathbf{v}) \).

The rank of the matrix \( K \), namely the number of its linearly independent rows, should be at least equal to \( L(N) \), in order to have a (pseudo-)inverse. This requires at least \( L(N) \) linearly independent basis filters, as expressed in Eq. (2).

One can notice that no constraints have been imposed on the directions \( \mathbf{n}_i = (n_{ix}, n_{iy}, n_{iz})^{T} \) of the basis filters. However, for symmetry reasons and in order to reduce the effects of noise in practical applications, we follow the approach of [20] and [21] by uniformly distributing the basis filters on the unit sphere. In Section 3.5, we present
a thorough analysis on this decision. Andersson, having studied polyhedra-theory in detail, describes the appropriate regular (Platonic) polyhedra for filters of order \( N \leq 3 \). Their vertices on the half-sphere correspond to the appropriate base-directions \( \mathbf{n} \). For \( N > 3 \), there exist no regular polyhedron, and therefore we are forced to use other types of polyhedra, such as the Archimedean solids, which present high (although not perfect) symmetry, with each of their vertices being equidistant from its neighbors. These solids are given in Table 1. Also, for \( N \geq 7 \) one can exploit the “geodesic dome” algorithm, which divides iteratively each face of a polyhedron into isosceles triangles.

### 2.1.1. Steerability of the filter responses

Let \( f(x) \) be the 3-D spatiotemporal image-sequence intensity (\( x \) represents both spatial and temporal location) and \( B_\omega(\omega) \) the corresponding 3-D spatiotemporal Fourier Transform. Also, let \( b_\omega(x) \) be the filter in the spatiotemporal domain, namely the 3-D inverse FT of \( B_\omega(\omega) \). Let finally \( y_\omega(x) = (b_\omega * f)(x) \) denote the response of the filter at orientation \( \mathbf{v} \), or equivalently \( Y_\omega(\omega) = B_\omega(\omega)F(\omega) \). The operator (*) denotes convolution. Based on the linearity of the Fourier Transform and the convolution operation, the interpolation scheme of Eq. (11) holds also for the response \( Y_\omega(\omega) \) or \( y_\omega(x) \). Namely:

\[
Y_\omega(\omega) = \sum_{i=1}^{j} t_i(\mathbf{v})Y_i(\omega),
\]

where \( Y_i(\omega) \) the response to the \( i \)th basis filter and \( t_i(\mathbf{v}) \) as defined in Eq. (11). Hereinafter, whenever we refer to the steerability of the filters, similar arguments hold for the responses and vice-versa.

### 2.1.2. Energy normalization

One may notice that the energy of a directional filter \( B_{\mathbf{v}}(\omega) \) is independent of its orientation \( \mathbf{v} \). Consider the filter of order \( N \), with direction along the temporal axis (\( z \)-direction). Using spherical coordinates, with \( \phi \) being the colatitude angle (angle from the \( z \)-axis) and \( \theta \) the longitude angle in the \( xy \)-plane, the energy of the filter is

\[
E(N) = \int_{0}^{2\pi} \int_{0}^{\pi} \cos^{2N} \phi \sin \phi \, d\phi \, d\theta = 2\pi \int_{0}^{\pi} \cos^{2N} \phi \sin \phi \, d\phi = \frac{4\pi}{2N + 1}.
\]

As the order \( N \) increases, the filter’s energy decreases. For energy-normalized filters, one has to divide with \( E(N) \).

### 2.1.3. A remark on the computational effort

The calculation of the interpolation vector \( t(\mathbf{v}) = \mathbf{d}(\mathbf{v}) \cdot K^{-1} \) for each \( \mathbf{v} \) requires significant computational effort. However, this step could be completely omitted by calculating off-line and creating a Look-Up-Table (LUT) of the interpolation vectors. The length of the interpolation vector \( l(\mathbf{v}) \) and consequently the computational effort is \( O((2N+1)(N+2)) \) (see Eq. (2)). For example, the required interpolation time for a filter of order \( N = 6 \), is increased by a factor of \( L(6)/L(3) = 28/10 \), compared to the time for a filter of order \( N = 3 \) (see Table 1).

### 2.2. The Donut Mechanism

In this section, we present the necessary theory for exploiting the introduced steerable filters in a motion estimation problem.

The 3-D power spectrum of a translating image with velocity \( \mathbf{u} = (u_x, u_y) \), lies on a plane \([7,8,11,13,14]\) perpendicular to the 3-D unit vector:

\[
s(\mathbf{u}) = \frac{(u_x, u_y, 1)}{\sqrt{|\mathbf{u}|^2 + 1}}.
\]

We search for two unit vectors \( s_n(\mathbf{u}) \) and \( s_0(\mathbf{u}) \), which are perpendicular to \( s(\mathbf{u}) \) and to each other. With \( \mathbf{e} \neq s(\mathbf{u}) \) being an arbitrary unit-vector, we obtain two such vectors from:

\[
s_n(\mathbf{u}) = s(\mathbf{u}) \times \mathbf{e}, \quad s_0(\mathbf{u}) = s(\mathbf{u}) \times s_n(\mathbf{u}).
\]

The vectors \( s_n(\mathbf{u}) \), \( s_0(\mathbf{u}) \) lie on and determine the motion plane. One can find \( N + 1 \) unit-vectors, which lie on the motion plane and are equally distributed (in angle), starting from \( s_n(\mathbf{u}) \), as follows:

\[
s_n(\mathbf{u}) = \cos \left( \frac{\pi n}{N + 1} \right) s_0(\mathbf{u}) + \sin \left( \frac{\pi n}{N + 1} \right) s_0(\mathbf{u}),
\]

\( n = 0, 1, \ldots, N \).

The previous equations are pictured in Fig. 2, where the motion plane and the vectors \( s(\mathbf{u}) \), \( s_n(\mathbf{u}) \), \( s_0(\mathbf{u}) \), and \( s_0(\mathbf{u}) \) are depicted.

One may notice now, that a directional filter is of the form \( \cos^N(t) \), where \( t \) is the 3-D angle from the filter’s direction \( \mathbf{v} \). It can be easily shown that it holds:

\[
\sum_{n=0}^{N} \left( \cos^N \left( t - \frac{n\pi}{N + 1} \right) \right)^2 = \text{const}(N).
\]

This is the leading-equation of the Donut Mechanism, illustrated in Fig. 3 (for the Quadrature pair filters, see text below). According to the Donut Mechanism, one can
search for the actual motion-plane by maximizing the following “Max-Steering” objective function:

\[ P_N(u) = \sum_{n=0}^{N} \left| B_{s_n(u)}(\omega) \cdot F(\omega) \right|^2 = \sum_{x} \sum_{n=0}^{N} \left| (b_{s_n(u)} * f)(x) \right|^2 \]

\[ = \sum_{x} \sum_{n=0}^{N} \left| y_{s_n}(x) \right|^2, \tag{18} \]

where \( B_{s_n}(\omega) = (s_n(u) \cdot \omega)^N \), \( b_{s_n}(x) \) its 3-D inverse FT and \( y_{s_n}(x) = (b_{s_n}(u) * f)(x) \). Parseval’s equation has been used for the transition from the second to the third term. The operator (*) represents convolution and \( f(x) \) stands for the image sequence, namely for the 3-D inverse FT of \( F(\omega) \).

According to Eq. (18), the Max-Steering distribution for a candidate velocity \( u \) is calculated as follows: Find the equally spaced vectors \( s_n(u) \), \( n = 0, 1, \ldots, N \), which lie on the candidate motion plane, using Eqs. (14)–(16). Calculate the responses \( y_{s_n}(x) \) of \( f(x) \) to the filters \( b_{s_n}(x) \). These filters form a “Donut” in the frequency domain, along a plane (see Fig. 3). Add the spatiotemporal energies of the responses for all directions \( s_n(u) \).

2.2.1. Steerability of the “Max-Steering” distribution

One does not have to compute and store the values of \( P_N(u) \) for each \( u \). It is easy to show that \( P_N(u) \) can be interpolated at any \( u \) from a fixed set of quadratic measurements. Notice that the filter responses (in space–time domain) are either pure real or pure imaginary, for even or odd filter order, respectively. Therefore, the squared absolute value operation \( |.|^2 \) can be easily handled in summations. Using the steering equation, (18) can be rewritten as follows:

\[ P_N(u) = \sum_{n=0}^{N} \sum_{x} \left| y_{s_n}(x) \right|^2 = \sum_{n=0}^{N} \sum_{x} \left( \sum_{j=1}^{I} t_i(s_n(u)) \cdot y_j(x) \right)^2 \]

\[ = \sum_{n=0}^{N} \sum_{x} \left( \sum_{j=1}^{I} t_i(s_n(u)) \cdot y_j(x) \right)^2, \tag{19} \]

where \( y_j(x) \) stands for the response to the \( j \)th basis filter and \( t_i \) is the corresponding interpolation weight. Expanding the square of the sum in the last term and interchanging summations, we conclude to:

\[ P_N(u) = \left| \sum_{n=0}^{N} \sum_{i=1}^{I} t_i(s_n(u)) \cdot y_i(x) \right|^2. \tag{20} \]

This states that \( P_N(u) \) can be calculated for arbitrary \( u \) from the \( \sigma' = \frac{\sigma + 1}{2} \) quadratic terms: \( \sum_{i,j} y_i(x) \cdot y_j(x) \).

2.3. 3-D steerable quadrature pairs

We consider a directional quadrature pair \( Q_{v}^{\text{even}}(\omega), Q_{v}^{\text{odd}}(\omega) \), in the frequency domain. The filters \( Q_{v}^{\text{even}}(\omega) \) and \( Q_{v}^{\text{odd}}(\omega) \) are directional along \( v \) and they are even- and odd-symmetric, respectively, such that the filter \( Q_{v}(\omega) = Q_{v}^{\text{even}}(\omega) + Q_{v}^{\text{odd}}(\omega) \) is zero for \( \omega \cdot v \leq 0 \), as shown in the last row of Fig. 1.

Andresson [20] has constructed steerable quadrature pairs of order \( M \):

\[ Q_{v}^{M,\text{even}}(\omega) = \sum_{N=0}^{M} a_N[N] B_{v}^{N}(\omega), \]

\[ Q_{v}^{M,\text{odd}}(\omega) = \sum_{N=-2}^{M} a_N[N] B_{v}^{N}(\omega), \tag{21} \]

by linearly combining the steerable filters \( B_{v}^{N}(\omega) \) of order \( N \leq M \), with the use of the appropriate weights \( a_N[N] \). By “\( N+ = 2 \)” in the summations, it is indicated that \( N \) increases by two. In other words, the even (odd) quadrature filter is constructed only from the directional filters of even (odd) order \( N \). The analysis of Andersson is restricted to filters of low order \( M \leq 3 \). We generalize the analysis to quadrature pairs of arbitrary higher order. The appropriate coefficients \( a_N[N] \) for \( M \leq 8 \) are plotted in Fig. 1, where also the construction of a quadrature pair of order \( M = 3 \) is illustrated.

In this section and the next Section 2.3.1, we give the mathematical details for the calculation of \( a_M[N] \) for arbitrary \( M \). Without loss of generality, we restrict the analysis for filters aligned along the \( \omega_{\phi}\text{-axis} \) (\( z\)-direction). With \( \phi \) the colatitude angle (angle from the \( z \)-axis), we construct the quadrature filter:

\[ Q_{0,0,1}^{M}(\omega) = Q_{0}^{M}(\phi) = \sum_{N=0}^{M} a_M[N] \cos^{N} \phi. \tag{22} \]
Since the filter has to be zero in the rear Fourier half-space \( \phi \in [\pi/2, \pi] \), we search for the appropriate coefficients \( \mathbf{a}_M = [a_M[0], a_M[1], \ldots, a_M[M]]^T \), such that the energy \( E_1 \) of the filter in the rear Fourier half-space

\[
E_1 = 2\pi \int_{\pi/2}^{\pi} \left( \sum_{N=0}^{M} a_M[N] \cos^2 \phi \right) \sin \phi \, d\phi \tag{23}
\]

is minimized, given that its total energy \( E_0 \) is constant:

\[
E_0 = 2\pi \int_{0}^{\pi} \left( \sum_{N=0}^{M} a_M[N] \cos^2 \phi \right) \sin \phi \, d\phi = 1. \tag{24}
\]

In the next Section 2.3.1, we show that \( E_0 \) and \( E_1 \) can be written in matrix notation as

\[
E_0 = \mathbf{x}_M^T \cdot \mathbf{M}_0 \cdot \mathbf{x}_M, \tag{25a}
\]

\[
E_1 = \mathbf{x}_M^T \cdot \mathbf{M}_2 \cdot \mathbf{x}_M, \tag{25b}
\]

\( \mathbf{M}_0 \) and \( \mathbf{M}_1 \) are positive definite matrices of size \((M + 1) \times (M + 1)\), where \( \mathbf{M}_0 \) is decomposed as \( \mathbf{M}_0 = \mathbf{W} \cdot \mathbf{D} \cdot \mathbf{W}^T \), with \( \mathbf{W} \) and \( \mathbf{D} \) being the square-root matrix and the diagonal matrix containing the eigenvalues of \( \mathbf{M}_0 \), respectively.

The positive definite matrix \( \mathbf{M}_0 \) is decomposed as \( \mathbf{M}_0 = \mathbf{W} \cdot \mathbf{D} \cdot \mathbf{W}^T \). Since \( \mathbf{D} \) is diagonal, its square-root matrix \( \mathbf{W} \) is obtained by taking the square root of its individual elements.

Now, we want to minimize the energy \( E_1 \), while we have assumed that the energy \( E_0 \) is unity. From Eq. (26), the restriction \( E_0 = 1 \) holds if the vector \( \mathbf{a}_M \) is of the form \( \mathbf{a}_M = \mathbf{L}^{-1} \cdot \mathbf{b} \), with \( \mathbf{b} \) being an arbitrary unit vector. Substituting this into Eq. (25b), we conclude to:

\[
E_1 = \mathbf{b}^T \cdot (\mathbf{L}^{-1})^T \cdot \mathbf{M}_1 \cdot \mathbf{L}^{-1} \cdot \mathbf{b} = \mathbf{b}^T \cdot \mathbf{M}_b \cdot \mathbf{b}, \tag{27}
\]

where \( \mathbf{M}_b = (\mathbf{L}^{-1})^T \cdot \mathbf{M}_1 \cdot \mathbf{L}^{-1} \).

From Eq. (27), the minimum value of \( E_1 \) is achieved for \( \mathbf{b} = \mathbf{b}_1 \), with \( \mathbf{b}_1 \) being the eigenvector of \( \mathbf{M}_b \) that corresponds to its minimum eigenvalue. Therefore, the desired vector \( \mathbf{a}_M \) is obtained using \( \mathbf{a}_M = \mathbf{L}^{-1} \cdot \mathbf{b}_1 \).

The vector \( \mathbf{d}_M = [d_M[0], d_M[1], \ldots, d_M[M]]^T \) can be written in a more compact form as

\[
\mathbf{d}_M = \mathbf{S}_M \cdot \mathbf{a}_M, \tag{30}
\]

where \( \mathbf{S}_M \) is of the form

\[
\mathbf{S}_M = \begin{bmatrix} k_N \frac{\pi}{2} \end{bmatrix}, \quad \text{if } N-n = 2l, \quad l = 0, 1, \ldots, \tag{31}
\]

\[
0, \quad \text{otherwise}
\]

\( \mathbf{a}_M = [a_M[0], a_M[1], \ldots, a_M[M]]^T \) with a square \((M + 1) \times (M + 1)\) matrix \( \mathbf{S}_M \) with elements:

\[
\mathbf{S}_M[n,N] = \begin{cases} k_N \frac{\pi}{2}, & \text{if } N-n = 2l, \quad l = 0, 1, \ldots, \tag{31} \\ 0, & \text{otherwise} \end{cases}
\]

\( n \) refers to zero-based row-number and \( N \) to zero-based column-number. With this notation, the energy of the filter in the rear Fourier half-space (Eq. (23)) can be written as

\[
E_1 = 2\pi \int_{\pi/2}^{\pi} \left( \sum_{n=0}^{M} d_M[n] \cos(n\phi) \right)^2 \sin \phi \, d\phi, \tag{32}
\]

which is expanded as follows, using some simple trigonometric identities:

\[
E_1 = \pi \int_{\pi/2}^{\pi} \sum_{n=0}^{M} d_M[n] \sum_{l=0}^{M} d_M[l](\cos((k+l)\phi) + \cos((k-l)\phi)) \sin \phi \, d\phi \tag{33}
\]

and interchanging summations and integration as

\[
E_1 = \frac{\pi}{2} \sum_{k=0}^{M} d_M[k] \sum_{l=0}^{M} d_M[l](R_0(k, l) + R_1(k, l) + R_2(k, l) + R_3(k, l)), \tag{34}
\]

where

\[
R_q(k,l) = (-1)^q \int_{\pi/2}^{\pi} \sin((k + (-1)^q/2)l + (-1)^q\phi) \, d\phi. \tag{35}
\]

Define the \((M + 1) \times (M + 1)\) matrix \( \mathbf{R} = \sum_{q=0}^{3} R_q \). Then, Eq. (34) is written in matrix notation as

\[
E_1 = \frac{\pi}{2} \mathbf{d}_M^T \cdot \mathbf{R} \cdot \mathbf{d}_M = \frac{\pi}{2} \mathbf{d}_M^T \cdot \mathbf{S}_M \cdot \mathbf{R} \cdot \mathbf{S}_M \cdot \mathbf{a}_M = \mathbf{a}_M^T \cdot M_1 \cdot \mathbf{a}_M, \tag{36}
\]

where \( M_1 = \mathbf{S}_M^T \cdot \mathbf{R} \cdot \mathbf{S}_M \) and \( \mathbf{S}_M \) is the matrix given in Eq. (31).

Using similar arguments, we obtain an equivalent equation for the total energy of the filter:

\[
E_0 = \mathbf{a}_M^T \cdot M_0 \cdot \mathbf{a}_M, \tag{37}
\]
with
\[ M_0 = M_1 + P \cdot M_1 \cdot P, \]  
where \( P \) an \((M + 1) \times (M + 1)\) diagonal matrix, defined by (zero-based matrix indexing):
\[ P(i,j) = \begin{cases} (-1)^i, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}. \]

\[ (17) \]

2.4. Donut Mechanism for quadrature pairs

In Appendix B, the following equations are proved:
\[ \sum_{n=0}^{M} \left( Q_{m,\text{even}}^{M} \left( t - \frac{n\pi}{M+1} \right) \right)^2 = \text{const}_e(M), \]  
\[ \sum_{n=0}^{M} \left( Q_{m,\text{odd}}^{M} \left( t - \frac{n\pi}{M+1} \right) \right)^2 = \text{const}_o(M). \]

We also calculate the constants \( \text{const}_e(M) \) and \( \text{const}_o(M) \). It can be shown that \( \text{const}_o(M) \approx \text{const}_e(M) \), as is verified in the diagram of Fig. 4. Since Eq. (40a) is similar to Eq. (17), one can define a Quadrature “Max-Steering” distribution, similar to the one in Eq. (18). Like in Eq. (16), for each candidate velocity, one should find \( M + 1 \) vectors \( s_m(u), m = 0, 1, \ldots, M \), equally distributed in angle. Then, the Quadrature “Max-Steering” distributions are
\[ P_{even}^M(u) = \sum_{\omega} \sum_{m=0}^{M} |Q_{m,\text{even}}^{M}(\omega) \cdot F(\omega)|^2, \]
\[ P_{odd}^M(u) = \sum_{\omega} \sum_{m=0}^{M} |Q_{m,\text{odd}}^{M}(\omega) \cdot F(\omega)|^2, \]
\[ P_M(u) = P_{even}^M(u) + P_{odd}^M(u). \]

2.4.1. Steerability of the Quadrature “Max-Steering” distributions

An interpolation scheme for the Quadrature “Max-Steering” distributions can be proposed, using similar arguments as in Section 2.2.1. From Eq. (21), the even quadrature filter of order \( M \) is constructed from \([M/2]\) even cosine filters \( B_{\text{even}}^N(\omega) \), which are interpolated respectively from \( I(\omega) \) basis filters. Therefore, the even quadrature filter is interpolated from \( J_0 = \sum_{N=0}^{M} I(N) \) even basis filters \( B_{\text{even}}^N(\omega) \). Consequently, the distribution \( P_{even}^M(u) \) can be calculated for each \( u \), using \( \sigma_{\text{even}} = \frac{J_0(2N+1)}{2} \) quadratic measurements. Similarly, \( P_{odd}^M(u) \) can be obtained from \( \sigma_{\text{odd}} = \frac{J_0(2N+1)}{2} \) quadratic measurements, with \( J_0 = \sum_{N=0}^{M} I(N) \).

3. Additional issues

3.1. Motion signatures—normal motion component

For a translating white noise pattern with velocity \( u = (u_x, u_y) \), the energy in the 3-D frequency domain is spread homogeneously over the motion plane: \( \omega_r = -u_x \omega_x - u_y \omega_y \). With \( \theta \) and \( \phi \) the longitude and colatitude angles, respectively, the motion-plane in spherical coordinates is given by
\[ \cos \phi = \sin \phi (-u_x \cos \theta - u_y \sin \theta), \]  
which is referred to as the “motion signature” in the \((\theta, \phi)\)-space [23]. Define the distribution:
\[ R_N(\theta, \phi) = \sum_x |(b_{n,\text{even}}^N(\theta, \phi) \ast f)(x)|^2, \]
where \( b_{n,\text{even}}^N(\theta, \phi) \) stands for the directional filter along the \((\theta, \phi)\)-direction, in spherical coordinates. Calculating \( R_N(\theta, \phi) \), we expect the “motion signature” to appear on the \( R_N(\theta, \phi) \) image of Eq. (42), which is verified in Fig. 5(a) and (b). Notice that the higher the filter order, the more clear and compact becomes the motion signature. Therefore, in the case of multiple motions, the multiple “motion signatures” can be easier separated. To extract the motion, a search for the present “motion signature” on the \( R_N(\theta, \phi) \) image [23] is needed. The integration (summation) over all possible “motion signature”-patterns on the \( R_N(\theta, \phi) \) image, in a Hough-like transform framework, is a suitable approach. However, the “Donut Mechanism” presented in Section 2.2 performs indirectly such a summation and therefore is used to reveal the motion.

Alternatively, consider a vertical strip translating along the \( x \)-direction, where the “aperture problem” takes place. The 3-D power spectrum is now concentrated along a line.

![Fig. 4. The values of \( \text{const}_e(M) \) and \( \text{const}_o(M) \) with respect to \( M \).](image-url)
\[ \omega_i = -u_i \omega_x, \] which defines infinite possible motion planes. We refer to this line as the “dominant 3-D orientation” line. Now, the motion estimation problem becomes under-constrained. Based on the above observations, one can extract the normal direction (2-D dominant orientation) and only the normal component of the motion by finding the peak of the \( R_N(\theta, \phi) \) image, namely finding the “3-D dominant orientation” (see Fig. 5(c)). The extraction of normal velocity is of great importance for many video processing tasks [17].

3.2. Capability in separating multiple motions—estimation bias

Multiple motions in image sequences often occur, near occlusion boundaries or due to transparency phenomena. It is obvious that narrow filters are more efficient in separating multiple motions, which will be manifested as multiple peaks in the “Max-Steering” distribution (18). Certainly, near-by velocities are resolved with more difficulty. Additionally, the motion of objects with low contrast (less spatial energy) is less probable to be captured.

The capability of the filters in resolving multiple velocities depends on the angular width of their lobe. The diagram of the -3 dB angular width with respect to the filter order, is presented in Fig. 6(a). We study the performance of the filters in capturing multiple motions with the following simple, but enlightening analysis. Consider a superposition of a still, vertically constant pattern (for example a sinusoid \( f_1(x,y,t) = \sin(k_1 x) \)) and a moving one with velocity \( u \) along the \( x \)-direction (for example \( f_2(x,y,t) = p \cdot \sin(k_2(x - ut)) \), with \( p \) the sinusoid’s amplitude). Also, consider that the spatiotemporal energy of the second pattern is \( p^2 \) (\( \leq 1 \)) times the energy of the first one. The same results hold if generally both patterns move with velocities \( u_1 \) and \( u_2 \), such that \( \tan^{-1} u_1 - \tan^{-1} u_2 = \tan^{-1} u \). In our case, the 3-D spatiotemporal spectrum is concentrated along two lines on the \((\omega_x, \omega_y)\)-plane, with latitude angles equal to zero and \( \phi_0 = \tan^{-1} u \), respectively. We do not have to study in detail the behavior of the “Max-Steering” distribution. It is adequate to study the behavior of the following distribution over the latitude angle (\( \delta(\phi) \) stands for the dirac function):

\[
A(\phi) = \int \left( \delta(\phi_1) + p^2 \delta(\phi_1 - \phi_0) \right) \cos^{2N}(\phi_1 - \phi) \, d\phi_1 \]

\[ = \cos^{2N}(\phi) + p^2 \cos^{2N}(\phi - \phi_0). \]

An ideal distribution should present two distinct peaks at \( \phi_1 = 0 \) and \( \phi_2 = \phi_0 \). However, the above function either presents two maxima at \( \phi_1 = b_1 \) and \( \phi_2 = \phi_0 - b_2 \), where \( b_1, b_2 > 0 \) are biases, or it is unimodal at \( 0 < \phi_1 \leq \phi_0/2 \), with the last inequality occurring when \( p^2 = 1 \). Namely, there is a low-limit value of \( \phi_0 \), under which the velocities cannot be resolved. This limit value is plotted with respect to \( p^2 \) and the filter’s order \( N \) in Fig. 6(b). Also, as already stated, even when the function presents two distinct peaks, these peaks are biased from their ideal positions towards each other. Fortunately, the bias decreases as the order \( N \) of the filters increases, as shown also in Fig. 6(c), where we plot the bias \( b_1 \) with respect to \( \phi_0 \) and \( N \), for \( p^2 = 0.5 \). The above facts are also verified experimentally.

3.3. Performance to noise and distortions

Consider a simple sequence with a moving 2-D cosine pattern: \( f(x) = \cos(\omega_1 x) \) (\( x \) represents both spatial and temporal location), with 3-D spatiotemporal Fourier transform:

\[ F(\omega) = \frac{1}{2} \delta(\omega - \omega_1) + \frac{1}{2} \delta(\omega + \omega_1) \quad (44) \]

and its noise-affected version: \( f_n(x) = f(x) + n(x) \), where \( n(x) \) is white spatiotemporal noise. Consider also the directional filter \( B^N(\omega) \) of order \( N \), aligned along the unit frequency vector \( \hat{\omega}_2 \). In the ideal noise-free case, the response of the filter is maximized when it is aligned with the dominant orientation \( \hat{\omega}_1 = \omega_1/|\omega_1| \). However, this may not hold when noise is present, depending on the SNR ratio. The situation becomes better as the filters become narrower. Assume that the filters used are not energy-normalized (see Section 2.1.2). For the perfectly aligned filter \( \hat{\omega}_2 = \hat{\omega}_1 \), the power of the signal component is independent of the filter order \( N \), while the power of the filtered noise component reduces as \( N \) increases. With \( S_{\text{snr}}(\omega) = q \) standing for the noise power spectrum, mathematically [24] we have:
\[ \sigma_n^2 = \int_{\omega} S_n(\omega) \left| B_{n,2}(\omega) \right|^2 \, d\omega = q \cdot E(N), \quad (45) \]

where \( E(n) \) is given in Eq. (13).

In the case of spatiotemporally correlated noise, similar conclusions can be drawn, since the interference due to the colored noise power spectrum decreases as the filter becomes narrower. As an extreme example, consider the previous section with two transparent motions, where we are interested only on the estimation of the dominant motion. The “weak” motion component can be considered as noise component. We have seen that the estimation error decreases as the filter order increases. Also, Beauchemin and Barron [25] study the frequency structure of the occlusion–distortion terms, showing that their orientations are essentially parallel to the constraint plane of the occluding signal. Finally, the wrap-around of the motion plane due to spatiotemporal aliasing [7,13] can be seen as distortion in the frequency domain. In all these cases, the narrower the filters, the better the performance, since the interference due to the distortion spectrum decreases. The above results can also be understood in the space–time domain. A narrower filter in the frequency domain, is wider in the space–time domain, exploiting therefore the information in a larger space–time region. This acts advantageously when the motion can be approximated as constant in a spatiotemporal neighborhood.

3.4. Why quadrature pairs?

Consider the noise affected sequence of the previous section and the quadrature pair \( q_{\omega_1}^{\text{even}}(x), q_{\omega_2}^{\text{odd}}(x) \), aligned in the Fourier domain along the unit vector \( \omega_2 \). With \( \omega_1 \) representing the dominant orientation as in the previous example, we have:

\[ O_{\omega_2}^{\text{even}}(\pm \omega_1) = \pm \omega(\omega_2), \quad O_{\omega_2}^{\text{odd}}(\pm \omega_1) = \pm \omega(\omega_2), \quad (46) \]

where \( \omega(\omega_2) \leq 1 \). The value of \( \omega(\omega_2) \) increases as the angle between the filter orientation \( \omega_2 \) and the dominant orientation \( \omega_1 \) decreases. The equality holds for \( \omega_2 = \omega_1 \), namely when the filters are aligned with \( \omega_1 \). Calculating the response of the even and odd filters separately, we have:

\[ y_{\omega_2}^{\text{even}}(x) = (q_{\omega_2}^{\text{even}} * f_n)(x) = \omega(\omega_2) \cos(\omega_1 x + \phi_1) + n_{\omega_2}^{\text{even}}(x), \]

\[ y_{\omega_2}^{\text{odd}}(x) = (q_{\omega_2}^{\text{odd}} * f_n)(x) = j \omega(\omega_2) \sin(\omega_1 x + \phi_1) + j n_{\omega_2}^{\text{odd}}(x), \quad (47) \]

where \( n_{\omega_2}^{\text{even}}(x) \) and \( n_{\omega_2}^{\text{odd}}(x) \) are the responses of the filters to the noise. Notice that the variance (power) of \( n_{\omega_2}^{\text{odd}} \) is equal to the variance of \( n_{\omega_2}^{\text{even}} \) and let it be \( \sigma^2 = \sigma_n^2 \). The power of the filtered noise is phase(location)-independent, while the power of the signal responses is phase(location)-dependent. In the noise-free case, the responses of the filters are maximized when \( \omega_2 = \omega_1 \), which is the desired goal. However, this may not hold when noise is present, and especially at spatiotemporal locations where the local SNR is low (given for the even filter):

\[ \text{SNR}(x) = 20 \log \left( \frac{\omega(\omega_2)}{\sigma_{\omega_2}} \right) + 20 \log \frac{\cos(\omega_1 x + \phi_1)}{\sigma_{\omega_1}^2}. \quad (48) \]

Additionally, consider the combination of the responses:

\[ y_{\omega_2} = y_{\omega_2}^{\text{even}} + y_{\omega_2}^{\text{odd}} = \omega(\omega_2) \cos(\omega_1 x + \phi_1) + j \omega(\omega_2) \sin(\omega_1 x + \phi_1) + n_{\omega_2}^{\text{even}} + j n_{\omega_2}^{\text{odd}}. \quad (49) \]

The power of the signal component and consequently the SNR is now phase(location)-independent: \( \text{SNR}(x) = 20 \log \left( \frac{\omega(\omega_2)}{2 \sigma_{\omega_2}} \right) \). Therefore, the results are less probable to be affected.

3.5. Why uniform distribution of the basis filters?

Intuitively, it seems reasonable to distribute the basis filters (their unit directions \( \mathbf{n} \)) uniformly, since it is reasonable to cover uniformly the 3-D space and moreover to have a rotation-invariant system. Knutsson [21] represents 3-D local directions using tensors. He gives the necessary properties for an appropriate tensor and describes a filter realization using quadrature filters. In order to meet the tensor requirements (such as rotation-invariance), he distributes the quadrature filters uniformly in the 3-D space.

In our case and handling the problem totally theoretically, one concludes that the initial distribution of the basis filters is the most appropriate.
filters does not affect the final results. However, the theoretical analysis cannot take into account some important practical issues. Consider the 3-D signal \( f(x) \) of Section 3.3, which is directional along the unit vector \( \omega_1 = \omega_1/|\omega_1| \), for example, a moving sinusoid \( f(x) = \cos(\omega_1^T x) \). Also, consider its noise affected version: \( f_n(x) = f(x) + n(x) \), where \( n(x) \) is white noise with power spectrum \( S_{nn}(\omega) = q \). Convolving \( f_n(x) \) with each basis filter \( b_1^N(x) \), we have:

\[
y_{n,i}(x) = y_i(x) + n_i(x),
\]

where

\[
y_{n,i}(x) = (b_1^N * f_n)(x),
\]

\[
y_i(x) = (b_1^N * f)(x) = (n_i \cdot \omega_1)^N \cos(\omega_1^T x),
\]

\[
n_i(x) = (b_1^N * n)(x).
\]

Notice that the amplitude of the filtered signal has changed with the cosine of the angle between the \( i \)th basis filter and the dominant orientation \( \omega_1 \), namely by \( (n_i \cdot \omega_1)^N \). Therefore, the mean SNR for each individual response is

\[
\text{SNR}_i = 20 \log \left( \frac{|n_i \cdot \omega_1|^N}{\sqrt{2\sigma_n}} \right).
\]

where \( \sigma_n \) corresponds to the filtered noise and it is given by Eq. (45). We want to interpolate the response \( y_{n,i}(x) \) of the filter \( b_1^N(x) \) (which is perfectly aligned with the dominant orientation \( \omega_1 \)), given the responses \( y_{n,i}(x) \) of the basis filters \( b_1^N(x) \). Multiplying Eq. (50) with \( t_i(\omega_1) \) and with summation for all \( i \leq I(N) \), we have:

\[
\sum_{i=1}^{I} t_i(\omega_1) y_{n,i}(x) = \sum_{i=1}^{I} t_i(\omega_1) y_i(x) + \sum_{i=1}^{I} t_i(\omega_1) n_i(x).
\]

By the definition of the steerable filters (see Eq. (12)) this means:

\[
y_{n,\omega_1}(x) = y_{\omega_1}(x) + n_{\omega_1}(x),
\]

where

\[
y_{n,\omega_1}(x) = \left( b_1^N \ast f_n \right)(x),
\]

\[
y_{\omega_1}(x) = \left( b_1^N \ast f \right)(x) = \cos(\omega_1^T x),
\]

\[
n_{\omega_1}(x) = \left( b_1^N \ast n \right)(x).
\]

The final mean SNR is therefore:

\[
\text{SNR} = 20 \log_{10} \left( \frac{1}{\sqrt{2\sigma_n}} \right),
\]

which is independent of the directions \( n_i \) of the basis filters.

However, the above theoretical result does not take account an important practical issue. The individual ratios SNR, in (52) depend on the directions of the basis filters, namely to \( n_i \). Now consider, that \( n_i \) are not regularly arranged in the 3-D angle (for example they are all concentrated near the z-axis), so that even the nearest in angle vector \( n_i \) is still far from the given direction \( \omega_1 \). For strong noise (or equivalently for weak signal) all the individual SNR, will be too small, so that all the responses \( y_i(x) \) may be “useless”, in the sense that their power may be under the minimum level of detection in real systems. Although their linear combination (with weights \( t_i \) theoretically produces correct results, in practical situations such as with hardware implementation, one has to take also into consideration that the dynamic range of electronics is limited, otherwise introducing non-linearities. With a computer implementation, the problem is less important but still exists, if for example we consider non-linearities due to quantization. For these reasons and in order to have a rotation-invariant system, which may be a requirement for some applications (for example [21]), we recommend the uniform distribution of the basis filters in the 3-D angle.

4. Algorithm for motion estimation and experiments

The introduced steerable filters, except from the motion estimation problem, can be exploited in many tasks that require the determination of 3-D local orientations, for example in 3-D MRI volumetric data. However, in this paper we deal only with the motion estimation problem. Therefore, the proposed algorithm, presented in the next subsection, shows how to use the steerable filters in motion estimation. Many modifications or extensions of the algorithm are applicable, as described later, that may improve its performance. However, since the main goal of this paper is to introduce higher order filters and to compare their performance with the lower order ones, we prefer to use a straightforward version of the algorithm, for simplicity.

4.1. Algorithm and implementation details

Based on the theoretical analysis, a simple algorithm for the estimation of the optical flow using the steerable filters, can be summarized in the following steps. We mention that all necessary convolutions, described below are performed in the frequency domain (as products). With this, the algorithm speeds up significantly, thanks to the existence of very fast FFT-IFFT implementations [26]. Notice that product in the frequency domain, corresponds to circular 3-D convolution in the spatiotemporal domain.

(1) Repeat the boundary pixels: Considering a finite-size sequence \( f(x,y,t) \), defined on the discrete grid \( (x,y,t) \in [0,N_x-1] \times [0,N_y-1] \times [0,T-1] \), repeat the boundary pixels according to:

\[
f(-x,y,t) = f(0,y,t),
\]

\[
f(N_x+x,y,t) = f(N_x-1,y,t)
\]

and similar equations for \( y \) and \( t \). Reflection of the boundary pixels is also possible. However, with the circular convolution, the above boundary-handling scheme generally produced better results than a simple zero-padding, or a boundary-reflection scheme, according to our experiments.
The parameter $\sigma_1$ can be considered as the standard deviation of the window along $x$, $y$, and $t$, normalized relatively to the sizes $D_x$, $D_y$, and $D_t$, respectively. Appropriate values are $\sigma_1 \sim 0.5$. Larger values, such as $\sigma_1 \sim 1$, may oversmooth the estimated motion field, although they may provide robustness against noise and help capturing multiple motions.

If quadrature pairs are used, calculate the “Max-Steering” distribution for Quadrature pairs (see Eq. (41) and Section 2.4.1) in a similar manner.

(5) Estimate the local velocity by finding the maximum of the “Max-Steering” distribution. The validity of the estimates could be characterized by the confidence measure described below, in Eq. (64). Also, if multiple motions are expected, search for multiple peaks in the distribution $P_N(\mathbf{u}, \mathbf{D})$. For this task we have used the following pseudo-code:

**Searching for multiple peaks:** We drop $\mathbf{D}$ from $P_N(\mathbf{u}, \mathbf{D})$ for notational simplicity. Notice that the distribution $P_N(\mathbf{u})$ is a 2-D image, calculated in practice over a discrete 2-D grid in the velocity space $\mathbf{u} = (u_x, u_y)$. Suppose that we search for two peaks. The generalization for arbitrary number of peaks is straightforward.

(a) Find the maximum of $P_N(\mathbf{u})$ and let it be $[\mathbf{u}_{\text{max}}, P_N(\mathbf{u}_{\text{max}})]$. Keep this as the first peak.

(b) Define a threshold for the minimum accepted second peak’s height. We have chosen the value $T_h = P_N(\mathbf{u}_{\text{max}})/3$.

(c) Set $P_N(\mathbf{u}_{\text{max}}) \leftarrow 0$ and find again the maximum of $P_N(\mathbf{u})$. Let it be $[\mathbf{u}_{\text{max}}, P_N(\mathbf{u}_{\text{max}})]$.

(d) If the euclidian distance of the new max-position $\mathbf{u}_{\text{max}}$ from the nearest zero-valued position in $P_N(\mathbf{u})$ is less than a predefined value $d$, then this does not correspond to a peak, therefore go to step c. Otherwise go to step e. We have used the value $d = 0.1$.

(e) The maximum $[\mathbf{u}_{\text{max}}, P_N(\mathbf{u}_{\text{max}})]$ is the wanted second peak, unless $P_N(\mathbf{u}_{\text{max}}) < T_h$, which means that we do not have a second peak.

Extensions of the described motion estimation algorithm are possible. For example, the window $w^2_{\mathbf{D}}(\mathbf{x})$ in Eq. (61) (namely the weights used to combine the pixel-wise motion constraints) is fixed for the whole image. On the other hand, as in [7] one could use weights that depend on the pixel-wise spatial gradient, which acts as a confi-
dence measure against “blank wall” (absence of texture) and “aperture” problems. Also, the size of the region $D$ (see Eq. (61)) is fixed for all image neighborhoods. However, in a textured image neighborhood, a small region $D$ might produce adequately confident local estimates, while for a textureless one, a larger region $D$ would be necessary. Therefore, an adaptive version of the algorithm is possible, that increases the size of $D$ until a confidence measure (see for example Eq. (64)) becomes greater than a threshold.

Finally, a coarse-to-fine strategy, which is used in many motion estimation techniques [7,17,27] in order to handle temporal aliasing, would improve the estimation results.

4.2. Performance measures

Following [10,17], as performance measures we use the angular errors:

$$\psi_E = \arccos(s(u) \cdot s(u)), \quad \psi_{E1} = |\arcsin(s_h \cdot s(u))|,$$

(63)

where $s(u)$ is the vector, normal to the correct motion plane (see Section 2.2), $s(u)_{\hat{s}}$ the corresponding estimated one and $s_h$ is the estimated “3-D dominant orientation” unit-vector, which should be perpendicular to $s(u)$. The error measure $\psi_{E1}$ is used for the measurements of the normal velocity component (see Section 3.1).

4.3. A confidence measure

Similarly to all optical-flow estimation techniques, the proposed algorithm produces local velocity estimates whose accuracy varies with the local characteristics of the underlying 3-D signal. The following simple, but effective “confidence” measure is used to distinguish the reliable local estimates from the unreliable ones:

$$C(D) = \frac{\max_{u} (P_N(u, D))}{\text{mean}_{u} (P_N(u, D))},$$

(64)

since the accuracy is directly connected with the sharpness of the peak in the “Max-Steering” distribution. Alternatively, we have also tried to use the eigenvalues (the minimum value or their product) of the Covariance Matrix of the bivariate Gaussian distribution that best-fits the “Max-Steering” distribution, similarly to ideas presented in [7,17]. However, this was not so efficient as the simple measure in Eq. (64).

4.4. Experiments and results

4.4.1. Comparison of the algorithm with other techniques

Although the main purpose of the paper is to introduce narrow directional filter and compare them with the lower order ones, in this subsection we present comparison results of the used algorithm with other optical flow techniques [17,27]. Although the proposed algorithm is not the core of this work and therefore not optimized, it performs satisfactory, compared to many other motion estimation techniques.

To make comparisons, we use the three synthetic sequences “Translating/Diverging tree” and “Yosemite fly”, without any noise addition. For these experiments we used the quadrature pairs of order $M = 4$. The algorithm of Section 4.1 has been applied with rectangular regions $D$ of size $D_x \times D_y \times D_z = 20 \times 20 \times 5$ (see step 4).

Also, the normalized standard deviation for the window $w(x)$ (see Eq. (62)) has been set equal to $\sigma_1 = 0.45$.

(1) Translating tree: We use the well-known “Translating tree” sequence, of length $T = 15$ frames. The moving image is shown in Fig. 7(a) and the actual motion field in Fig. 7(b). In the pre-filtering stage, the parameter of the spherical Gaussian pre-filter $G(\omega)$ (Eq. (60) in step 2) is set equal to $\sigma_\omega = 0.8\pi$. The estimated motion-field for the 7th frame, is given in Fig. 7(c). The mean angular errors for full density and 75% density, calculated using the confidence measure of Eq. (64), are tabulated in Tables 2 and 3 and compared with the results of other optical-flow techniques [17]. One may notice that the mean error is not reduced for 75% density, because the estimates are similarly confident for this experiment.

(2) Diverging tree: The “Diverging tree” sequence, of length $T = 15$ frames, is used. The moving image is shown in Fig. 7(a) and the actual motion field in Fig. 8(a). The same pre-filter is used as in the previous experiment. The estimated motion-field for the 7th frame and for full density is depicted in Fig. 8(b). The 60% most confident
estimates, based on the confidence measure of Eq. (64), are shown in Fig. 8(c). The mean angular errors are given in Tables 4 and 5, together with the results of other techniques [17].

(3) Yosemite fly, with clouds: The realistic synthetic "Yosemite fly" sequence is used, with the clouds. For the Gaussian pre-filter $G(\omega)$ (Eq. (60) in step 2) we used for $\sigma_{x}$ the value $\frac{\pi}{5}$. The relatively strong low-pass here, improves the results because it reduces the temporal aliasing effects due to high velocity components at the bottom-left image region. The 7th frame of the sequence (for which the results are given) and the actual motion field are depicted in Fig. 9(a) and (b). Also the estimated motion field for full density and 75% density (using the confidence measure of Eq. (64)) are given in Fig. 9(c) and (d), respectively. As it was expected, the less confident estimates correspond mainly to the fractal cloudy sky and the bottom-left region of the sequence, where temporal aliasing (due to high velocity components) and boundary effects corrupt the estimates. The mean angular error for 100%, 75%, and 50% density is reported in Tables 6–8, together with the results of other optical-flow techniques [17,27].

### Table 2
Translating tree—100% density

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn and Schunck (original)</td>
<td>38.72</td>
</tr>
<tr>
<td>Anandan</td>
<td>4.54</td>
</tr>
<tr>
<td>Nagel</td>
<td>2.44</td>
</tr>
<tr>
<td>Horn and Schunck (modified)</td>
<td>2.02</td>
</tr>
<tr>
<td>Singh (step 1, $n = 2, w = 2$)</td>
<td>1.64</td>
</tr>
<tr>
<td>Quadrature Steerable ($M = 4$, $\sigma_{\omega} = 0.8\pi$, $\sigma_{1} = 0.45$)</td>
<td>1.43</td>
</tr>
<tr>
<td>Singh (step 2, $n = 2, w = 2$)</td>
<td>1.25</td>
</tr>
<tr>
<td>Uras et al. (unthresholded)</td>
<td>0.62</td>
</tr>
</tbody>
</table>

### Table 4
Diverging tree—100% density

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Singh (step 1, $n = 2, w = 2, N = 4$)</td>
<td>17.66</td>
</tr>
<tr>
<td>Horn and Schunck (original)</td>
<td>12.02</td>
</tr>
<tr>
<td>Singh (step 2, $n = 2, w = 2, N = 4$)</td>
<td>8.60</td>
</tr>
<tr>
<td>Anandan (frames 19 and 21)</td>
<td>7.64</td>
</tr>
<tr>
<td>Quadrature Steerable ($M = 4$, $\sigma_{\omega} = 0.8\pi$, $\sigma_{1} = 0.45$)</td>
<td>5.83</td>
</tr>
<tr>
<td>Uras et al. (unthresholded)</td>
<td>4.64</td>
</tr>
<tr>
<td>Nagel</td>
<td>2.94</td>
</tr>
<tr>
<td>Horn and Schunck (modified)</td>
<td>2.55</td>
</tr>
</tbody>
</table>

### Table 3
Translating tree—75% density and less

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
<th>Density (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn and Schunck (original), $|\mathbf{V}(l)| &gt; 5.0$</td>
<td>32.66</td>
<td>55.9</td>
</tr>
<tr>
<td>Heeger (level 1)</td>
<td>4.53</td>
<td>57.8</td>
</tr>
<tr>
<td>Nagel, $|\mathbf{V}(l)| &gt; 5.0$</td>
<td>2.24</td>
<td>53.2</td>
</tr>
<tr>
<td>Horn and Schunck (modified), $|\mathbf{V}(l)| &gt; 5.0$</td>
<td>1.89</td>
<td>53.2</td>
</tr>
<tr>
<td>Quadrature Steerable ($M = 4$, $\sigma_{\omega} = 0.8\pi$, $\sigma_{1} = 0.45$)</td>
<td>1.53</td>
<td>75</td>
</tr>
<tr>
<td>Fleet and Jepson ($\tau = 2.5$)</td>
<td>0.32</td>
<td>74.5</td>
</tr>
</tbody>
</table>

### Table 5
Diverging tree—about 60% density and less

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
<th>Density (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn and Schunck (original), $|\mathbf{V}(l)| &gt; 5.0$</td>
<td>8.93</td>
<td>54.8</td>
</tr>
<tr>
<td>Heeger</td>
<td>4.49</td>
<td>64.2</td>
</tr>
<tr>
<td>Uras et al. (det$(H)$) $\geq 1.0$</td>
<td>3.83</td>
<td>60.2</td>
</tr>
<tr>
<td>Quadrature Steerable ($M = 4$, $\sigma_{\omega} = 0.8\pi$, $\sigma_{1} = 0.45$)</td>
<td>3.75</td>
<td>60</td>
</tr>
<tr>
<td>Fleet and Jepson ($\tau = 2.5$)</td>
<td>0.99</td>
<td>61</td>
</tr>
</tbody>
</table>

4.4.2. Comparisons of the introduced narrow filters with the lower order ones
In this subsection we present experimental results which verify the advantages of the introduced narrow directional filters compared to lower order ones, mainly on the robustness against noise and the capability in separating multiple motions.

(1) Translating sinusoid: With this experiment we demonstrate the facts described in Sections 3.1 and 3.4. The image sequence is a vertically constant sinusoid pattern of size $N_x \times N_y = 65 \times 65$ pixels, which translates...
Table 6
Yosemite fly—100% density

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn and Schunck (original)</td>
<td>31.69</td>
</tr>
<tr>
<td>Singh (step 1, n = 2, w = 2)</td>
<td>15.28</td>
</tr>
<tr>
<td>Anandan</td>
<td>13.36</td>
</tr>
<tr>
<td>Singh (step 2, n = 2, w = 2)</td>
<td>10.44</td>
</tr>
<tr>
<td>Nagel</td>
<td>10.22</td>
</tr>
<tr>
<td>Horn and Schunck (modified)</td>
<td>9.78</td>
</tr>
<tr>
<td>Uras et al. (unthresholded)</td>
<td>8.94</td>
</tr>
<tr>
<td>Quadrature Steerable (M = 4)</td>
<td>7.34</td>
</tr>
<tr>
<td>Papenberg et al. (3-D) [27]</td>
<td>1.78</td>
</tr>
</tbody>
</table>

Table 7
Yosemite fly—about 75% density and less

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
<th>Density (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn and Schunck (original),</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Heeger (level 0)</td>
<td>22.82</td>
<td>64.2</td>
</tr>
<tr>
<td>Quadrature Steerable (M = 4)</td>
<td>5.44</td>
<td>75.0</td>
</tr>
<tr>
<td>σ₁ = 0.45</td>
<td>19.1</td>
<td></td>
</tr>
</tbody>
</table>

Table 8
Yosemite fly—about 50% density and less

<table>
<thead>
<tr>
<th>Technique</th>
<th>Mean error (°)</th>
<th>Density (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heeger (combined)</td>
<td>15.93</td>
<td>44.8</td>
</tr>
<tr>
<td>Nagel,</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Horn and Schunck (modified),</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Fleet and Jepson (τ = 2.5)</td>
<td>4.63</td>
<td>34.1</td>
</tr>
<tr>
<td>Lucas and Kanade</td>
<td>4.28</td>
<td>35.1</td>
</tr>
<tr>
<td>Quadrature Steerable (M = 4)</td>
<td>3.51</td>
<td>50.0</td>
</tr>
<tr>
<td>σ₁ = 0.45</td>
<td>19.1</td>
<td></td>
</tr>
</tbody>
</table>

along the x-direction with velocity 0.4 pixels/frame: $f(x, y, t) = \cos(\frac{2\pi}{N}(x + 0.4t))$. It is corrupted by spatiotemporal zero-mean Additive White Gaussian Noise (AWGN) $N(0, \sigma = 0.6)$. Its length $T = 33$ frames. The 17th frame is depicted in Fig. 10(a). Due to the “aperture problem”, there is ambiguity about the velocity along the y-direction. For each pixel $(x_c, y_c)$, we look only for the normal velocity component by searching the “dominant 3-D orientation”, namely the peak in the $(0, \phi)$-distribution:

$$R_\phi(x_c, y_c, t, 0, \phi) = \sum_{(x,y)\in D(x_c, y_c)} \left[ (q^M_{(0,\phi)} * f)(x, y, t) \right]^2,$$

like in Eq. (43), where $q^M_{(0,\phi)}(x, y, t) = q^M_{(0,\phi)}^{even}(x, y, t) + q^M_{(0,\phi)}^{odd}(x, y, t)$ is the quadrature filter of order $M = 3$. The summation is performed in a rectangular spatial region $D(x_c, y_c)$ of size $w \times w$, combining the pixel-wise constraints. We experimented also in estimating the normal velocity by using only the even or the odd parts of the quadrature pairs. As performance measure, we use the measure $\psi_{E_1}$ of Eq. (63). In Fig. 10, we present the angular error results for the 17th frame, using $w = 1$ (pixel-wise constraints) and $w = 4$. Additionally, the mean angular error is plotted with respect to $w$. As a conclusion, the estimates are better with the use of the quadrature filters, than using only their even or odd part, especially at regions of low local SNR.

(2) Translating tree: We demonstrate the improved performance of higher order filters in the case of noise presence, using the “Translating tree”. We use the algorithm of Section 4.1 with the directional cosine filters. In the pre-filtering stage, the parameter of the spherical Gaussian low-pass filter $G(\omega)$ (see algorithm’s step 2) $\sigma_\omega = \frac{\pi}{4}$. The motion-field is estimated for the 7th frame, using overlapping blocks $D$ of size $D_1 \times D_2 \times D_3 = 20 \times 20 \times 7$ (see algorithm’s step 4) and $\sigma_1 = 1$. After performing Monte–Carlo simulations (ten experiments per noise-level) with spatiotemporal zero-mean AWGN $N(0, \sigma)$, we present in Fig. 11 the mean angular error, with respect to the noise power. In accordance with the discussion of Section 3.3, higher order filters show better performance as the noise power increases.

(3) Diverging tree: In a similar context as in the previous experiment, we present the results for the “Diverging tree” sequence. For this experiment, we apply the algorithm with the quadrature pairs. The estimation is performed for the 7th frame, using overlapping blocks of size $D_x \times D_y \times D_z = 20 \times 20 \times 7$ and $\sigma_1 = 1$, as previously. In Fig. 12, the presented results were obtained using two different low-pass filters in the pre-filtering stage. The first one is relatively weak, with $\sigma_\omega = \frac{\pi}{4}$, while the second one is stronger with $\sigma_\omega = 0.3\pi$. As expected, the pre-filter affects the performance. For low noise power, the low-pass filtering acts disadvantageously, since it removes mainly valuable information instead of noise. The opposite holds for higher noise power. As previously, the main conclusion is that higher order filters improve the performance as the noise power increases.
Yosemite fly: The results refer to the 7th frame of the sequence, applying the algorithm with overlapping blocks of size $D_x \times D_y \times D_t = 20 \times 20 \times 7$ and with $\sigma_1 = 1$, similarly to the previous two experiments. We study the behavior of the directional cosine filters for increasing order $N$, applying the algorithm to the sequence with the clouds. In Fig. 13(a), we present the mean angular error with respect to $N$. These results were obtained using the low-pass Gaussian pre-filter (step 2) with $\sigma_\omega = 0.3\pi$. As one can verify, the estimates are improved as the filter order increases. The performance is improved mainly for the clouds region and the bottom-right region, where the image velocity is high.

In order to study the behavior of the filters with noise, we add zero-mean AWGN $N(0, \sigma)$ of increasing power. It should be noticed however, that the cloud pattern of the sequence is fractal and therefore the brightness constancy assumption does not hold exactly for this region. In order to have a more objective comparison between the filters of different order, we report the mean errors only for the rigid region of the scene. In Fig. 13, we present the mean angular error of the estimates for increasing noise power. The results refer to the “Yosemite fly—without clouds” sequence, not including the black sky. We use two different low-pass Gaussian filters. For the stronger low-pass ($\sigma_\omega = 0.25\pi$), one can notice relatively small improvement of the results, for increasing order $N$. For the weaker low-pass ($\sigma_\omega = 0.3\pi$), the improvement is greater, as expected.

(5) Transparent Translating tree with Lenna: We demonstrate the effectiveness of higher order filters in separating and estimating multiple motions. The “Translating tree” sequence is combined with the static “Lenna” image, in order to produce the synthetic transparent sequence of Fig. 14(a). A $256 \times 256$ version of the “Lenna” image is used, in gray scale. More specifically, we use the region $x \in [30, 179]$ and $y \in [70, 219]$ of this image, which is a $150 \times 150$ region, similar to the tree sequence. The “Lenna” image remains still, namely its velocity $u_1 = 0$, while the...
velocity of the tree is about $u_2 = 1.8$ pixels/frame. The two 8-bits/pixel images are combined simply in an additive manner.

For the low-pass filter $G(\omega)$ in the pre-filtering stage of the algorithm (step 2), we use $\sigma_{\omega} = 0.8\pi$. Notice that the separation of multiple motions becomes harder for stronger low-pass filters. The algorithm is applied with blocks

![Graph](image-url)

**Table 9**

<table>
<thead>
<tr>
<th>$N$</th>
<th>Velocities resolved (two peaks found) (%)</th>
<th>Error Lenna ($^\circ$)</th>
<th>Error Tree ($^\circ$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>35</td>
<td>30.9</td>
<td>14.8</td>
</tr>
<tr>
<td>5</td>
<td>64</td>
<td>8.1</td>
<td>6.2</td>
</tr>
<tr>
<td>6</td>
<td>48</td>
<td>16.5</td>
<td>14.5</td>
</tr>
</tbody>
</table>

![Images](image-url)

**Fig. 13.** Yosemite fly: (a) The mean angular error with respect to $N$, for the sequence with clouds. (b) Without clouds. The two pre-filters, using the sequence without clouds, and ignoring the black sky.

**Fig. 14.** Transparent Translating tree with Lenna: (a) A frame of the transparent sequence. (b–d) The “Max-Steering” distribution for the low-left corner shown in (a), for $N = 4, 5, 6$, respectively. Notice that the odd filters ($N = 5$) are more appropriate for this region. (e–h) Estimated motion fields, for $N = 4, 5, 6$. The circles indicate that a second velocity (peak in the Max-Steering distribution) could not be found.
of size $D_x \times D_y \times D_t = 30 \times 30 \times 7$ and with $\sigma_1 = 1$ for the weighting window of Eq. (62).

We search for two distinct peaks in the “Max-Steering distribution”. For $u_1 = 0$ (Lenna) and $u_2 = 1.8$ pixels/frame, the angle between the motion planes is $\phi_0 = \tan^{-1}(u_2) \approx 60^\circ$. According to the arguments in Section 3.2 and the diagrams in Fig. 6(a) and (b), one can see that even in the ideal case, the two motions could be separated only for $N \geq 3$. In this experiment, three cases were studied, with $N = 4$, 5, and 6 (Table 9). In each case, two peaks were found with 35%, 64%, and 48% participation, respectively. Whenever two peaks are found, the estimated velocities are assigned to the “Lenna” and the “Tree” images in such a way that the mean angular error is minimized. The corresponding motion fields are given in Fig. 14(e)–(h) for $N = 4$, 5 and $N = 6$. One may notice that specifically for this experiment, the odd filters ($N = 5$) perform better than the even ones ($N = 4, 6$). This verifies our arguments in Section 3.4. Finally, it is verified that the estimates are improved as the order increases ($N = 6$, compared to $N = 4$).
The used block size is zero-velocity of the static background. Therefore, with poorly defined. Its velocity is very small, very close to the identity is introduced in the estimation of the vehicle velocities, which however decreases using narrower filters. The latter were calculated by manual feature-point tracking. The diagrams reveal that a bias towards the zero-velocity is introduced in the estimation of the vehicle velocities, as shown in Fig. 15. We study the performance of the filters in separating the velocity of the vehicles from the zero-background velocity, in the three regions (1,2,3) shown in Fig. 15, of size $D_x \times D_y = 30 \times 30$. The size of the used 3-D blocks along $t$ is $D_t = 10$ frames. The given results refer to the 11th frame. The “Max-Steering” distribution for $N = 4$ and $N = 8$ is depicted in Fig. 16(a)–(f), for each region. It is shown that the separation of the vehicle velocities is possible or easier using narrower filters. In the diagrams of Fig. 16(i) and (j), we plot the estimated horizontal velocity of the bottom-left car and the taxi, for $N = 2, 4, 6, 8$, together with the true velocities. The latter were calculated by manual feature-point tracking. The diagrams reveal that a bias towards the zero-velocity is introduced in the estimation of the vehicle velocities, which however decreases using narrower filters.

Finally, one may also notice that a pedestrian moves at the top left of the sequence. This object is very small and poorly defined. Its velocity is very small, very close to the zero-velocity of the static background. Therefore, with the used block size $D_x \times D_y = 30 \times 30$, it is obvious that the peak corresponding to the pedestrian in the Max-Steering distribution will be hidden by the strong background’s peak. Therefore, the pedestrian’s velocity cannot be resolved in a multiple-motions estimation framework. It can be revealed in a single-motion estimation framework, using a smaller block (e.g., $D_x \times D_y = 15 \times 15$), as verified in Fig. 16(g) and (h). The estimated velocity is biased towards zero, due to the presence of the static background. However, the bias is reduced as the filters order increases. This is verified in Fig. 16(k).

(6) Hamburg Taxi: The well-known “Hamburg Taxi” sequence is used, where the three moving vehicles occlude the static background, as shown in Fig. 15. We study the performance of the filters in separating the velocity of the vehicles from the zero-background velocity, in the three regions (1,2,3) shown in Fig. 15, of size $D_x \times D_y = 30 \times 30$. The size of the used 3-D blocks along $t$ is $D_t = 10$ frames. The given results refer to the 11th frame. The “Max-Steering” distribution for $N = 4$ and $N = 8$ is depicted in Fig. 16(a)–(f), for each region. It is shown that the separation of the vehicle velocities is possible or easier using narrower filters. In the diagrams of Fig. 16(i) and (j), we plot the estimated horizontal velocity of the bottom-left car and the taxi, for $N = 2, 4, 6, 8$, together with the true velocities. The latter were calculated by manual feature-point tracking. The diagrams reveal that a bias towards the zero-velocity is introduced in the estimation of the vehicle velocities, which however decreases using narrower filters.

Finally, one may also notice that a pedestrian moves at the top left of the sequence. This object is very small and poorly defined. Its velocity is very small, very close to the zero-velocity of the static background. Therefore, with the used block size $D_x \times D_y = 30 \times 30$, it is obvious that the peak corresponding to the pedestrian in the Max-Steering distribution will be hidden by the strong background’s peak. Therefore, the pedestrian’s velocity cannot be resolved in a multiple-motions estimation framework. It can be revealed in a single-motion estimation framework, using a smaller block (e.g., $D_x \times D_y = 15 \times 15$), as verified in Fig. 16(g) and (h). The estimated velocity is biased towards zero, due to the presence of the static background. However, the bias is reduced as the filters order increases. This is verified in Fig. 16(k).

(7) Calendar: We use the frames 180–212 of the original “Calendar” sequence. In order to demonstrate the effectiveness of the methodology in estimating multiple motions, we apply it to the lower part of the sequence, with size $350 \times 50$, as shown in Fig. 17(a), where a toy moves leftwards–upwards, while occluding the background. The results in 17(b)–(e), refer to the frame 197. The algorithm was applied with quadrature filters of order $M = 5$, the low-pass pre-filter with $\sigma_a = \frac{\pi}{5}$ in the pre-filtering stage, 3-D blocks of size $D_x \times D_y \times D_t = 25 \times 25 \times 7$ and $\sigma_t = 1$. As verified from Fig. 17, the algorithm for $M = 5$ can effectively resolve the two motions at most of the regions with multiple-motions boundaries.

5. Conclusion

Extending and generalizing the mathematical analysis of previous work [20], we presented a rigorous mathematical framework together with the appropriate methodology to construct 3-D steerable directional cosine filters and quadrature pairs, of arbitrary narrow bandwidth and arbitrary higher order. Incorporating the “Donut Mechanism” [7] in the constructed quadrature pairs, we offer additional tools to advance the motion estimation problem.

We presented in detail a simple algorithm to apply the directional cosine filters or quadrature pairs in the (multiple) motions estimation problem in video sequences, exploiting the processing power of the filters’ steerability. Also, using simple mathematical analysis, we demonstrated the robustness of higher order filters in estimating noisy or multiple motions, at the cost of a relatively increased computational effort. After giving some implementation details, we presented typical experimental results on synthetic and real sequences, which demonstrate the effectiveness of the algorithm and support our arguments.

Appendix A. Nth power of cosine

Use the complex exponential form of cosine:
where \( C_n[n] = \frac{N}{N+n} \). Let \( N \) be an even number (the analysis for \( N \) odd is almost identical). Split the summation into three parts (two for \( N \) odd):

\[
\cos^n \phi = \frac{1}{2^n} \left( \sum_{n=0}^{N/2-1} C_n[n] e^{i(2\pi n)\phi} + \sum_{n=N/2}^{N/2-1} C_n[n] e^{-i(2\pi n)\phi} + C_N[N/2] \right)
\]

\[
= \frac{1}{2^n} \left( \sum_{n=0}^{N/2-1} C_n[n] e^{i(2\pi n)\phi} \right)
= \frac{1}{2^n} \left( \sum_{n=0}^{N/2-1} C_n[n] e^{i(2\pi n)\phi} + \sum_{n=0}^{N/2-1} C_n[n] e^{-i(2\pi n)\phi} \right)
+ C_N[N/2] \cos((N-2n)\phi).
\]

The analysis for \( N \) odd is identical. With \( C_n[n] = \frac{N}{(N+n)} \) and setting:

\[
k_n[n] = \begin{cases} 
\frac{1}{2^n} C_N[N/2], & \text{for } n = N/2 \text{ and if } N \text{ is even} \\
\frac{1}{2^n} C_N[n], & \forall n \text{ otherwise}
\end{cases}
\]

the expansion is generally:

\[
\cos^n \phi = \sum_{n=0}^{N/2} k_n[n] \cos((N-2n)\phi)
= \sum_{n=N-2N/2}^{N/2-1} k_n\left[\frac{N-n}{2}\right] \cos(n\phi).
\]

Appendix B. Deriving Donut Mechanism for quadrature pairs

Setting:

\[
x_M[n, L] = \cos^{2L} \left( t - \frac{n\pi}{M+1} \right),
\]

we prove that:

\[
\sum_{n=0}^{M} x_M[n, L] = \frac{1}{2^n} C_{2L}[L](M+1), \quad \text{where}
C_{2L}[l] = \frac{(2L)!}{l!(2L-l)!}. \tag{B.2}
\]

**Proof**

\[
\sum_{n=0}^{M} x_M[n, L] = \sum_{n=0}^{M} \cos^{2L} \left( t - \frac{n\pi}{M+1} \right) = \frac{1}{2^n} \sum_{n=0}^{M} \left( e^{i(\frac{n\pi}{M+1})} + e^{-i(\frac{n\pi}{M+1})} \right)^{2L}
= \frac{1}{2^n} \sum_{n=0}^{M} \sum_{l=0}^{2L} \binom{2L}{l} e^{i(l\frac{n\pi}{M+1})} e^{-i(l\frac{n\pi}{M+1})}
= \frac{1}{2^n} \left( C_{2L}[l] \sum_{n=0}^{M} e^{i(l\frac{n\pi}{M+1})} + \sum_{l=0}^{2L} C_{2L}[l] \sum_{n=0}^{M} e^{-i(l\frac{n\pi}{M+1})} \right)
= \frac{1}{2^n} \left( C_{2L}[l] \sum_{n=0}^{M} e^{i(l\frac{n\pi}{M+1})} + \sum_{l=0}^{2L} C_{2L}[l] \sum_{n=0}^{M} e^{-i(l\frac{n\pi}{M+1})} \right)
= \frac{1}{2^n} C_{2L}[L](M+1). \quad \square \tag{B.3}
\]

We used the fact that \( \sum_{n=0}^{M} e^{i(\frac{n\pi}{M+1})} = 0 \), which can be easily proved based on the properties of the \((M+1)\)th roots of unity. Now, we prove Eq. (40b) for \( Q_{M,\text{even}} \).

**Proof**

\[
\sum_{n=0}^{M} Q_{M,\text{even}} \left( t - \frac{n\pi}{M+1} \right)^2 = \sum_{n=0}^{M} \sum_{N=0}^{M/2} a_M[N] \cos^{2N} \left( t - \frac{n\pi}{M+1} \right) \tag{B.4}
= \sum_{n=0}^{M/2} \sum_{N=0}^{M/2} a_M[N] \cos^{2N} \left( t - \frac{n\pi}{M+1} \right) \tag{B.4}
= \sum_{n=0}^{M/2} \sum_{N=0}^{M/2} a_M[N] \sum_{N=1}^{M/2} a_M[N] x_M[n, N] \tag{B.4}
= \sum_{N=1}^{M/2} \sum_{N=1}^{M/2} a_M[N] \sum_{N=1}^{M/2} a_M[N] x_M[n, N, N + N]. \tag{B.4}
\]

We used the definition in Eq. (B.1) and the fact that \( x_M[n, N_1] x_M[n, N_2] = x_M[n, N_1 + N_2] \). Using Eq. (B.2) for \( L = N_1 + N_2 \), we conclude to:

\[
\sum_{n=0}^{M} \left( Q_{M,\text{even}} \left( t - \frac{n\pi}{M+1} \right) \right)^2 = (M+1) \sum_{N_1=0}^{M/2} \sum_{N_2=0}^{M/2} a_M[N_1] a_M[N_2] x_M[n, N_1] x_M[n, N_2] \tag{B.5}
\]

\[
\times \frac{a_M[N_1] a_M[N_2]}{2^{2L}} C_{2L}[L]. \quad \square
\]
The proof for $Q^{M,\text{odd}}$ uses the same arguments. Using $L = N_1 + N_2$, it can be similarly shown that:

$$
\sum_{n=0}^{M} \left( Q^{M,\text{odd}} \left( t - \frac{nM}{M+1} \right) \right)^2
= (M + 1) \sum_{N_1=1}^{[M/2]} \sum_{N_2=1}^{[M/2]} \frac{a_M[2N_1 - 1]a_M[2N_2 - 1]}{2^{2(L-1)}} C_{2(L-1)[L - 1]}. 
\tag{B.6}
$$

References