Differentiation of Discrete Multidimensional Signals

Hany Farid and Eero P. Simoncelli, *Senior Member, IEEE*

***Abstract*—We describe the design of finite-size linear-phase separable kernels for differentiation of discrete multidimensional signals. The problem is formulated as an optimization of the rotation-invariance of the gradient operator, which results in a simultaneous constraint on a set of one-dimensional low-pass prefilter and differentiator filters up to the desired order. We also develop extensions of this formulation to both higher dimensions and higher order directional derivatives. We develop a numerical procedure for optimizing the constraint, and demonstrate its use in constructing a set of example filters. The resulting filters are significantly more accurate than those commonly used in the image and multidimensional signal processing literature.**

***Index Terms*—Derivative, digital filter design, discrete differentiation, gradient, steerability.**

I. Introduction

ONE OF THE most common operations performed on signals is that of differentiation. This is especially true for multidimensional signals, where gradient computations form the basis for most problems in numerical analysis and simulation of physical systems. In image processing and computer vision, gradient operators are widely used as a substrate for the detection of edges and estimation of their local orientation. In processing of video sequences, they may be used for local motion estimation. In medical imaging, they are commonly used to estimate the direction of surface normals when processing volumetric data.

Newton's calculus provides a definition for differentiation of continuous signals. Application to discretized signals requires a new definition, or at the very least a consistent extension of the continuous definition. Given the ubiquity of differential algorithms and the ever-increasing prevalence of digital signals, it seems surprising that this problem has received relatively little attention. In fact, many authors that describe applications based on discrete differentiation do not even describe the method by which derivatives are computed. In this paper, we define a set of principled constraints for multidimensional derivative filter design, and demonstrate their use in the design of a set of high-quality filters.

Manuscript received June 10, 2003; revised November 12, 2003. The work of H. Farid was supported by an Alfred P. Sloan Fellowship, a National Science Foundation (NSF) CAREER Grant (IIS-99-83806), and a departmental NSF Infrastructure Grant (EIA-98-02068). The work of E. P. Simoncelli was supported by an NSF CAREER Grant (MIP-9796040), an Alfred P. Sloan Fellowship, the Sloan-Swartz Center for Theoretical Visual Neuroscience at New York University, and the Howard Hughes Medical Institute. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Phillippe Salembier.

H. Farid is with the Computer Science Department, Dartmouth College, Hanover, NH 03755 USA (e-mail: farid@cs.dartmouth.edu).

E. P. Simoncelli is with the Center for Neural Science and the Courant Institute of Mathematical Sciences, New York University, New York, NY 10012 USA (e-mail: eero.simoncelli@nyu.edu).

Digital Object Identifier 10.1109/TIP.2004.823819

A. One-Dimensional (1-D) Derivatives

The lack of attention to derivative filter design probably stems from the fact that the most natural solution—the difference between adjacent samples—appears at first glance to be completely acceptable. This solution arises from essentially dropping the limit in the continuous definition of the differential operator

$$D\{f\}(x) \equiv \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon) - f(x)}{\varepsilon} \tag{1}$$

and holding ε fixed at the distance between neighboring samples. These "finite differences" are widely used, for example, in numerical simulation and solution of differential equations. But in these applications, the spacing of the sampling lattice is chosen by the implementor, and thus can be chosen small enough to accurately represent the variations of the underlying signal. In many digital signal processing applications, however, the sampling lattice is fixed beforehand, and finite differences can provide a very poor approximation to a derivative when the underlying signal varies rapidly relative to the spacing of the sampling lattice.

An alternative definition comes from differentiating a continuous signal that is interpolated from the initial discrete signal. In particular, if one assumes the discrete signal, $f[k]$, was obtained by sampling an original continuous function containing frequencies no higher than $2\pi/T$ cycles/length at a sampling rate of T samples/length, then the Nyquist sampling theorem implies that the continuous signal may be reconstituted from the samples

$$f(x) = \sum_k f[k] \cdot s_T(x - kT) \tag{2}$$

where $s_T(x) = (\sin(\pi x/T))/(\pi x/T)$ is a (continuous) "sinc" function, $f(\cdot)$ is the continuous-time signal, and $f[\cdot]$ is its discretely sampled counterpart. Assuming that the sum in the above equation converges, we can differentiate the continuous function on each side of the equation, yielding

$$\begin{aligned} \mathcal{D}\{f\}(x) &= \sum_k f[k] \cdot \mathcal{D}\{s_T\}(x - kT) \\ &= \sum_k f[k] \cdot d(x - kT) \end{aligned} \tag{3}$$

where $d(x)$ is the derivative of the sinc function $d(x) = (\pi^2 x/T^2 \cos(\pi x/T) - \pi/T \sin(\pi x/T))/(\pi x/T)^2$. Note that the derivative operator is only being applied to continuous functions, $f(\cdot)$ and $s_T(\cdot)$.

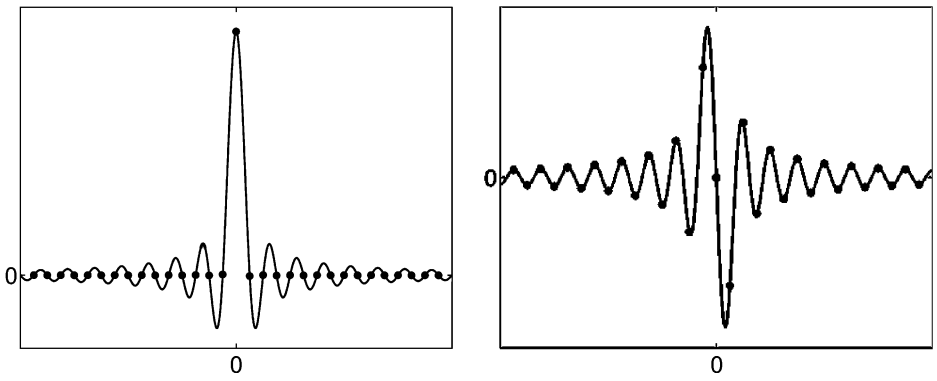

Fig. 1. The ideal interpolator (sinc) function (left) and its derivative (right). Dots indicate sample locations.

One arrives at a definition of discrete differentiation by sampling both sides of the above equation on the original sampling lattice

$$\begin{aligned}\mathcal{D}\{f\}[n] &\equiv \mathcal{D}\{f\}(x)|_{x=nT} \\ &= \sum_k f[k] \cdot d\left((n-k)T\right) \\ &= \sum_k f[k] \cdot d_T[n-k] \end{aligned} \tag{4}$$

where $d_T[\cdot]$ is the T-sampled sinc derivative. Note that the right side of this expression is a convolution of the discretely sampled function $f[\cdot]$, with the discrete filter $d_T[\cdot]$, and thus the continuous convolution need never be performed. If the original function was sampled at or above the Nyquist rate, then convolution with the sampled derivative of the sinc function will produce the correct result. In practice, however, the coefficients of the sinc derivative decay very slowly, as shown in Fig. 1, and accurate implementation requires very large filters. In addition, the sinc derivative operator has a large response at high frequencies, making it fragile in the presence of noise. Nevertheless, this solution (and approximations thereof) are widely used in 1-D signal processing (e.g., [1]–[4]).

B. Multidimensional Derivatives

For multidimensional signals, the derivative is replaced with the gradient: the vector of partial derivatives along each axis of the signal space. Consideration of this problem leads to an additional set of constraints on the choice of derivative filters. Assume again that the sampled function has been formed by uniformly sampling a continuous function above the Nyquist rate. As before, we can reconstruct the original continuous function as a superposition of continuous interpolation functions. In two dimensions, for example, we have the following:

$$f(x,y) = \sum_{k,l} f[k,l] \cdot c(x-kT, y-lT) \tag{5}$$

where T is the sample spacing (assumed to be identical along both the x and y axes), $f(x,y)$ is the continuous function, $f[x,y]$ is the discretely sampled function, and the interpolation function $c(x,y)$ is a separable product of sinc functions

$$\begin{aligned} c(x,y) &= s_T(x) \cdot s_T(y) \\ &= \frac{\sin\left(\frac{\pi x}{T}\right)}{\frac{\pi x}{T}} \cdot \frac{\sin\left(\frac{\pi y}{T}\right)}{\frac{\pi y}{T}}. \end{aligned} \tag{6}$$

Again assuming that the sum in (5) converges, we can differentiate both sides of the equation. Without loss of generality, consider the partial derivative with respect to x

$$\mathcal{D}_x\{f\}(x,y) = \sum_{k,l} f[k,l] \cdot \mathcal{D}_x\{c\}(x-kT, y-lT) \tag{7}$$

where $\mathcal{D}_x\{\cdot\}$ indicates a functional that computes the partial derivative of its argument in the horizontal direction. Again, one arrives at a definition of the derivative of the discrete function by sampling both sides of the above equation on the original sampling lattice

$$\begin{aligned}\mathcal{D}_x\{f\}[n,m] &\equiv \mathcal{D}_x\{f\}(x,y)|_{x=nT,y=mT} \\ &= \sum_{k,l} f[k,l] \cdot \mathcal{D}_x\{c\}\left((n-k)T, (m-l)T\right) \\ &= \sum_{k,l} f[k,l] \cdot \mathcal{D}\{s\}\left((n-k)T\right) \cdot s\left((m-l)T\right) \\ &= \sum_{k,l} f[k,l] \cdot d_T[n-k] \cdot \delta[m-l] \end{aligned} \tag{8}$$

where T-sampling the sinc function gives the Kroenecker delta function $\delta[\cdot]$ and $d_T[\cdot]$ is the T-sampled sinc derivative. Once again, however, the coefficients of this filter decay very slowly and accurate implementation requires very large filters.

Furthermore, the resulting two-dimensional (2-D) filter does not take into consideration the primary use of derivatives when applied to signals of more than one dimension (e.g., images or video). In the field of computer vision, directional derivatives are used to compute, for example, local edge orientation, motion (optical flow), or depth from stereo. In these applications, one relies on the linear algebraic property of gradients that the derivative in an arbitrary direction can be computed from a linear combination of the axis derivatives

$$\mathcal{D}_{\vec{u}}\{f\}(x,y) = u_x \mathcal{D}_x\{f\}(x,y) + u_y \mathcal{D}_y\{f\}(x,y) \tag{9}$$

where $\vec{u} = (u_x \;\; u_y)^t$ is a unit vector [5]–[7]. In this regard, the separable sinc seems somewhat odd, because the resulting 2-D x-derivative filter has nonzero samples only along a row of the input lattice, whereas the y-derivative filter has nonzero samples only along a column. The directional derivative at an angle of $\pi/4$ will contain nonzero samples from a single row and column, as illustrated in Fig. 2.

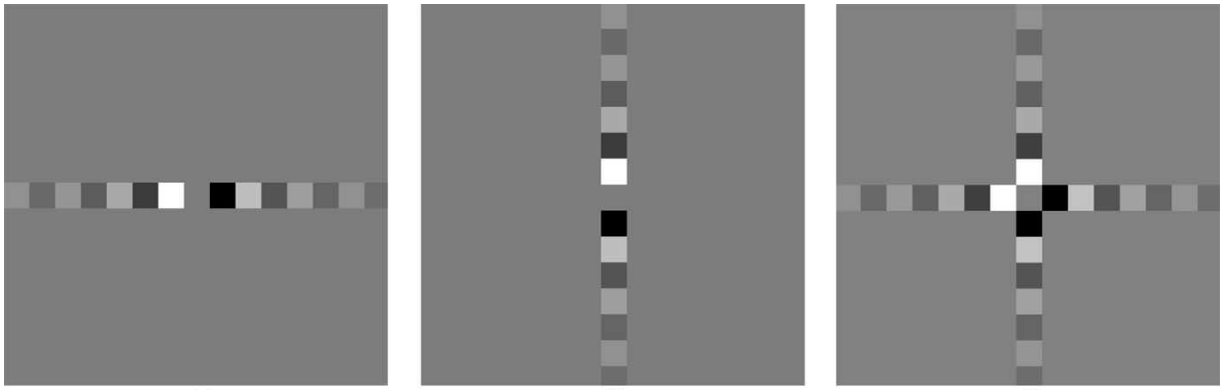

Fig. 2. Illustration of sinc-based 2-D differentiators. Shown are: (a) horizontal; (b) vertical; and (c) oblique (45°) differentiation filters. The last of these is constructed from a linear combination of the on-axis filters, as specified by (9).

Thus, both local differences and approximations to sinc derivatives seem inadequate as discrete extensions of differential operators. The solution to this dilemma is to consider alternative interpolation functions $c(x,y)$ in (5). Two alternative interpolation solutions appear commonly in the literature: fitting polynomial approximations (e.g., [8]–[13]), and smoothing with a (truncated) Gaussian filter (e.g., [14]–[17]). These choices are, however, somewhat arbitrary, and the primary topic of this paper is the development of a more principled choice of interpolator function. We formulate the desired constraints for first-order differentiation, with the second-order and higher order constraints building upon this constraint. We then develop a practical design algorithm for interpolator filters with matched derivative filters of arbitrary order and length.

II. Derivative Filter Constraints

We start with an assumption that the interpolation function $c(x,y)$ should be separable. This both simplifies the design problem, and improves the efficiency of subsequent derivative computations. We have also explored nonseparable filters, but the marginal improvement in accuracy appears to be not worth the considerably higher computational expense. We also assume that the interpolator is symmetric about the origin. Finally, we assume that all axes should be treated the same. In summary, the interpolator function is a separable product of identical symmetric 1-D functions.

Thus, for example, the 2-D interpolator is written as a separable product $c(x,y) = p(x) \cdot p(y)$. The partial derivative (with respect to x) of this interpolator is

$$\mathcal{D}_x\{c\}(x,y) = d_1(x) \cdot p(y) \tag{10}$$

where $d_1(\cdot)$ is the derivative of $p(\cdot)$. With this interpolator, the sampled derivative (as in (8)) becomes

$$\begin{aligned}\mathcal{D}_x\{f\}[n,m] &= \mathcal{D}_x\{f\}(x,y)|_{x=nT,y=mT} \\ &= \sum_{k,l} f[k,l] \cdot d_1\left((n-k)T\right) \cdot p\left((m-l)T\right) \\ &= \sum_{k,l} f[k,l] \cdot d_1[n-k] \cdot p[m-l]. \end{aligned} \tag{11}$$

The discrete derivatives are computed using two discrete 1-D filters, $p[\cdot]$ and $d_1[\cdot]$, which are the T-sampled versions of $p(\cdot)$ and $d_1(\cdot)$, respectively. More precisely, differentiation in x is accomplished by separable convolution with the differentiation filter $d_1[\cdot]$ along the x-dimension and with the interpolator $p[\cdot]$ in the y-dimension.

How then do we choose p? Rather than trying to approximate the sinc, which is virtually impossible with short filters, we directly impose the constraint of (9). That is, we seek filters such that the derivative in an arbitrary direction can be computed from a linear combination of the axis derivatives. This type of rotation-invariance property was first examined by Danielsson, who compared various standard filter sets based on a similar criterion [5]. It is important to note that giving up the sinc approximation means that the interpolator, $c(x,y)$, will not be spectrally flat. Thus, the resulting derivative filters will not compute the derivative of the original signal, but of a spectrally reshaped signal.

Furthermore, we note that the directional derivative filters computed using (9) will not necessarily be rotated versions of a common filter (a property that has been termed *steerability* in the computer vision literature [7]). If we were to include a steerability constraint with our assumption that the interpolator be a separable product of symmetric 1-D functions, then the resulting interpolator would necessarily be a Gaussian. As such, we chose not to impose steerability as a primary constraint, although we discuss its use as a regularizing constraint in the filter design stage in Section III.

A. First Order

We first compute the Fourier transform of both sides of (9), relying on the well-known fact that differentiation corresponds to multiplication by a unit-slope ramp function in the frequency domain

$$\begin{aligned} j(\vec{u}^t \cdot \vec{\omega}) P(\omega_x) P(\omega_y) F(\omega_x, \omega_y) = u_x D_1(\omega_x) P(\omega_y) F(\omega_x, \omega_y) \\ + u_y P(\omega_x) D_1(\omega_y) F(\omega_x, \omega_y) \end{aligned} \tag{12}$$

where $P(\cdot)$ and $D_1(\cdot)$ are the discrete-space Fourier transforms of the 1-D filters $p[x]$ and $d_1[x]$, respectively [2]

$$P(\omega) = \sum_n p[n] e^{-j2\pi\omega n}. \tag{13}$$

This constraint should hold for $-\pi < \{\omega_x, \omega_y\} < \pi$ and for all orientations, as represented by the unit vector $\vec{u}$.

We now define an error functional by dropping the factor of $F(\omega_x, \omega_y)$ and integrating the squared error in this equality over the frequency domain, as well as the unit vector $\vec{u}$ that specifies the direction of differentiation. In addition, in order to make the error independent of the magnitude of the filters (e.g., to avoid the trivial zero solution), we divide the expression by the total Fourier energy of the prefilter as shown in (14), at the bottom of page. Expanding the square in the numerator, we note that the cross terms (those containing $u_x u_y$) will integrate to zero because they are anti-symmetric. The remaining error expression is as shown in the equation at the bottom of page. The integrands in the numerator differ only by permutation of the axes, and thus the value of the two integrals will be identical. Combining these into a single term (and dropping the irrelevant factor of two) gives

$$E\{P, D_1\} = \frac{\int_{\vec{\omega}} \left|\left(j\omega_x P(\omega_x) - D_1(\omega_x)\right) P(\omega_y)\right|^2}{\int_{\vec{\omega}} \left|P(\omega_x)P(\omega_y)\right|^2}. \quad (15)$$

Finally, noting that $P^2(\omega_y)$ is strictly real due to the assumed symmetry of $p(y)$, we can factor it out of both numerator and denominator expressions and eliminate it from the quotient

$$\begin{aligned} E\{P, D_1\} &= \frac{\int_{\omega_x} \left|j\omega P(\omega_x) - D_1(\omega_x)\right|^2 \int_{\omega_y} P^2(\omega_y)}{\int_{\omega_x} P^2(\omega_x) \int_{\omega_y} P^2(\omega_y)} \\ &= \frac{\int_{\omega} \left|j\omega P(\omega) - D_1(\omega)\right|^2}{\int_{\omega} P^2(\omega)}. \end{aligned} \quad (16)$$

Thus, the 2-D constraint has been reduced to a 1-D constraint, in the form of a Rayleigh quotient. Equation (16) is the fundamental error functional used within this paper, but we note that the integral in the numerator can be augmented to include a weighting function over frequency. For applications in which the signals to be differentiated are known to occupy a particular range of frequencies, inclusion of such a weighting function can improve differentiation accuracy over this range. Before discussing the optimization method, we show how these same basic concepts can be generalized to both higher order derivatives and higher dimensions.

B. Second Order

Similar to the first-order derivative, the constraint embodying the desired linear-algebraic properties of the second-order derivative is

$$\begin{aligned} \mathcal{D}^2_{\vec{u}}\{f\}(x,y) &= (u_x\mathcal{D}_x + u_y\mathcal{D}_y)^2\{f\}(x,y) \\ &= u_x^2\mathcal{D}_x^2\{f\}(x,y) + 2u_xu_y\mathcal{D}_x\left\{\mathcal{D}_y\{f\}\right\}(x,y) \\ &\quad + u_y^2\mathcal{D}_y^2\{f\}(x,y). \end{aligned} \quad (17)$$

In the Fourier domain, this constraint takes the form

$$\begin{aligned} \left(j\vec{u}^t\cdot\vec{\omega}\right)^2 P(\omega_x)P(\omega_y) &= u_x^2 D_2(\omega_x)P(\omega_y) \\ &\quad + 2u_xu_y D_1(\omega_x)D_1(\omega_y) + u_y^2 P(\omega_x)D_2(\omega_y) \end{aligned} \quad (18)$$

where $P(\cdot)$ and $D_1(\cdot)D_2(\cdot)$ are the Fourier transforms of the 1-D filters $p[x]$, $d_1[x]$ and $d_2[x]$, respectively. As before, we define the numerator of our error functional by integrating over both frequency and orientation variables

$$\begin{aligned} N\{P, D_1, D_2\} = \int_{\vec{\omega}}\int_{\vec{u}} \Big| j^2(\vec{u}^t\cdot\vec{\omega})^2 P(\omega_x)P(\omega_y) - \big(u_x^2 D_2(\omega_x) \\ \times P(\omega_y) + 2u_xu_y D_1(\omega_x)D_1(\omega_y) + u_y^2 P(\omega_x)D_2(\omega_y)\big)\Big|^2. \end{aligned} \quad (19)$$

This can again be simplified by noting that cross terms containing $u_x^3u_y$ and $u_xu_y^3$ are antisymmetric and will integrate to zero, leaving

$$\begin{aligned} N\{P, D_1, D_2\} &= \int_{\vec{\omega}} \left|\left(j^2\omega_x^2 P(\omega_x) - D_2(\omega_x)\right) P(\omega_y)\right|^2 \\ &\quad + \left|j^2\omega_x\omega_y P(\omega_x)P(\omega_y) - D_1(\omega_x)D_1(\omega_y)\right|^2. \end{aligned} \quad (20)$$

As in the first-order case, the full error functional is formed by dividing by the L^2-norm of the prefilter, as shown in (21), at the bottom of the page. The first term reduces to a 1-D constraint, but the second term does not, and thus we are forced to optimize the full 2-D expression.

$$E\{P, D_1\} = \frac{\int_{\vec{\omega}}\int_{\vec{u}} \left|j(\vec{u}^t\cdot\vec{\omega})P(\omega_x)P(\omega_y) - (u_x\mathcal{D}_1(\omega_x)P(\omega_y) + u_yP(\omega_x)D_1(\omega_y))\right|^2}{\int_{\vec{\omega}} \left|P(\omega_x)P(\omega_y)\right|^2} \quad (14)$$

$$E\{P, D_1\} = \frac{\int_{\vec{\omega}} \left|(j\omega_x P(\omega_x)P(\omega_y) - \mathcal{D}_1(\omega_x)P(\omega_y))\right|^2 + \int_{\vec{\omega}} \left|(j\omega_y P(\omega_x)P(\omega_y) - P(\omega_x)D_1(\omega_y))\right|^2}{\int_{\vec{\omega}} \left|P(\omega_x)P(\omega_y)\right|^2}$$

$$E\{P, D_1, D_2\} = \frac{\int_{\vec{\omega}} \left|\left(j^2\omega_x^2 P(\omega_x) - D_2(\omega_x)\right) P(\omega_y)\right|^2 + \left|j^2\omega_x\omega_y P(\omega_x)P(\omega_y) - D_1(\omega_x)D_1(\omega_y)\right|^2}{\int_{\vec{\omega}} \left|P(w_x)P(w_y)\right|^2} \quad (21)$$

C. Higher Order and Higher Dimensions

The constraint embodying the desired linear algebraic properties of the Nth-order derivative is

$$j^N(\vec{u}^t \cdot \vec{\omega})^N P(\omega_x)P(\omega_y) = (u_x D(\omega_x)P(\omega_y) + u_y P(\omega_x)D(\omega_y))^N \quad (22)$$

where the right-hand side will require nth order derivative filters $D^n(\omega) \equiv D_n(\omega)$. Following the same general framework as in the previous two sections yields a constraint on the form of the prefilter and first- through Nth-order derivative filters.

The problem formulation defined above in two dimensions extends naturally to higher dimensions. In full generality, K dimensions, Nth-order derivatives come from expanding the following expression:

$$j^N(\vec{u}^t \cdot \vec{\omega})^N \prod_k P(w_k) = \left[\sum_k \left(u_k D(w_k) \prod_{l \neq k} P(w_l)\right)\right]^N \quad (23)$$

where we have used a numerical index variable $k \in [1 \ldots K]$ as an axis label in place of the previous mnemonic labels (x, y). As before, this can be used to generate an error functional for filter design.

III. Discrete Derivative Filter Design

With the desired constraints in place, we now describe a practical method for designing derivative filters of a given order and support (length). For notational clarity, we assume the design of odd-length filters—the formulation for even-length filters requires only minor modifications. We also assume that the prefilter and all even-order derivative kernels are symmetric about the central tap and that all odd-order derivative kernels are antisymmetric.

A. First Order

In order to design a filter of length L, we define a parameter vector $\vec{p}$ of length $(L+1)/2$ containing the independent prefilter samples (the others are determined by the symmetry assumption). Similarly, we define a parameter vector $\vec{d_1}$ of length $(L-1)/2$ containing the independent derivative kernel samples (the remaining samples determined by antisymmetry of the kernel). We construct a discrete version of the error functional in (16)

$$E(\vec{p}, \vec{d_1}) = \frac{|j\omega F_s \vec{p} - F_a \vec{d_1}|^2}{|F_s \vec{p}|^2} \quad (24)$$

where F_s and F_a are matrices whose columns contain the real (symmetric) and imaginary (asymmetric) components of the discrete Fourier basis of size $K \gg L$, such that $F_s\vec{p}$ gives the discrete Fourier transform (DFT) of the (symmetric) prefilter, and $F_a\vec{d_1}$ gives the DFT of the (antisymmetric) derivative filter. This may be bundled more compactly as

$$E(\vec{u}) = \frac{|M_1 \vec{u}|^2}{|M_2 \vec{u}|^2} \quad (25)$$

with concatenated matrices $M_1 = (jwF_s \; -F_a)$, $M_2 = (F_s \; 0)$, and vector $\vec{u} = \binom{\vec{p}}{\vec{d_1}}$. This error function is in the form of a Rayleigh quotient, and thus the solution may be found using standard techniques. Specifically, we solve for the generalized eigenvector associated with the minimal eigenvalue of M_1^2 and M_2^2 (i.e., $M_1^2\vec{u} = \lambda M_2^2 \vec{u}$). This eigenvector is then rescaled so that the resulting prefilter has unit sum, and the prefilter and derivative filters are then constructed by symmetrizing or antisymmetrizing the corresponding portions of $\vec{u}$ (i.e., the prefilter is constructing by concatenating a reversed copy of $\vec{u}$ with itself, and the derivative filter is constructing by concatenating a reversed and negated copy of $\vec{u}$ with itself). We note that it would be preferable to introduce the unit-sum constraint directly into the error function, but this would destroy the Rayleigh quotient and would lead to a less elegant solution.

1) Uniqueness: The uniqueness of the solution depends on the isolation of the minimal eigenvalue. In experimenting with the design procedure, we found the minimal eigenvalue comes within the machine tolerance of zero when designing filters of length $L = 11$ samples. For larger filters, the second eigenvalue also approaches zero. Thus, for filters of length $L > 11$, an additional constraint must be invoked to uniquely constrain (i.e., regularize) the solution. As mentioned earlier, our primary design constraint does not guarantee that directional derivative filters will be steerable. The sinc-based derivative filters are extreme examples of the failure of this property, as shown in Fig. 2.

TABLE I
Example Filter Taps for Optimal Differentiators of Various Orders and Sizes. Shown are Half of the Filter Taps, the Other Half are Determined by Symmetry: the Prefilter and Even-Order Derivatives are Symmetric and the Odd-Order Derivatives Anti-Symmetric About the Origin (Sample Number 0)

	Sample Number				
	0	1	2	3	4
$\vec{p}$	0.540242	0.229879			
$\vec{d_1}$	0.000000	-0.425287			
$\vec{p}$	0.426375	0.249153	0.037659		
$\vec{d_1}$	0.000000	-0.276691	-0.109604		
$\vec{p}$	0.439911	0.249724	0.030320		
$\vec{d_1}$	0.000000	-0.292315	-0.104550		
$\vec{d_2}$	-0.471147	0.002668	0.232905		
$\vec{p}$	0.361117	0.245410	0.069321	0.004711	
$\vec{d_1}$	0.000000	-0.193091	-0.125376	-0.018708	
$\vec{d_2}$	-0.273118	-0.056554	0.137778	0.055336	
$\vec{p}$	0.365406	0.246217	0.067088	0.003992	
$\vec{d_1}$	0.000000	-0.193357	-0.121482	-0.015964	
$\vec{d_2}$	-0.288736	-0.057325	0.147520	0.054174	
$\vec{d_3}$	0.000000	0.336539	0.012759	-0.111680	
$\vec{p}$	0.317916	0.234494	0.090341	0.015486	0.000721
$\vec{d_1}$	0.000000	-0.143928	-0.118739	-0.035187	-0.003059
$\vec{d_2}$	-0.191974	-0.061661	0.085598	0.061793	0.010257
$\vec{d_3}$	0.000000	0.203718	0.053614	-0.065929	-0.027205

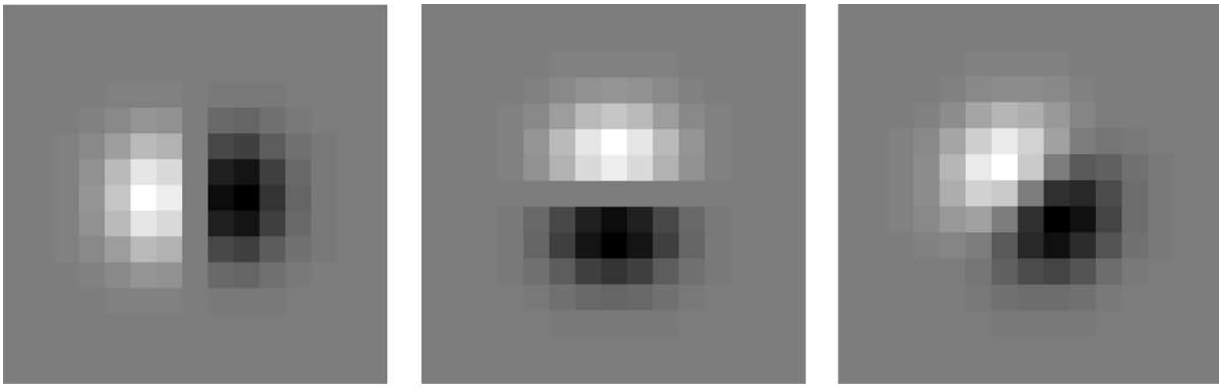

Fig. 3. Shown are: (a) horizontal; (b) vertical; and (c) oblique (30°) differentiation filters. The last of these is constructed from a linear combination of the on-axis filters, as specified by (9). These filters should be compared with those of Fig. 2.

One possible secondary constraint is that the low-pass filter be as flat as possible over the entire frequency range [4]. But we choose instead a constraint that is more consistent with our emphasis on rotation-invariance. In particular, we have chosen to use steerability as a secondary constraint in order to choose from amongst those filters that satisfy the primary constraint. Specifically, we seek a solution of the form

$$\vec{u} = \vec{u}_0 + U_1 \vec{s} \tag{26}$$

where $\vec{u}_0$ is any solution of minimizing (25) and U_1 is a matrix containing the zero-eigenvalue eigenvectors from (25). The parameter vector $\vec{s}$ is chosen to minimize the steerability error functional

$$E_1(\vec{s}) = \sum_{\theta=0}^{\pi} \left[R_\theta \left\{ \left| \mathcal{F}\left(\vec{p} \cdot \vec{d}_1^{\,t}\right) \right| \right\} - \left| \mathcal{F}\left(\cos(\theta)\vec{p} \cdot \vec{d}_1^{\,t} + \sin(\theta)\vec{d}_1 \cdot \vec{p}^{\,t} \right) \right| \right]^2 \tag{27}$$

where R_θ is the 2-D rotation operator (implemented on a discrete lattice using bicubic interpolation), and $\mathcal{F}$ is the 2-D Fourier operator. Note that this secondary steerability constraint does not interfere with the primary constraint of (25), since the two constraints are imposed in complementary orthogonal subspaces. This nonlinear minimization is initialized with an optimally designed 11-tap filter padded with zeros to accommodate the desired filter length. A gradient descent minimization is then performed on $\vec{s}$ in order to optimize, across all orientations, the steerability constraint of (27). The resulting minimization yields a pair of stable filters that perfectly satisfy the original constraint of (16) and are maximally steerable.

B. Second Order

Unlike the first-order filter design problem, the second-order error functional (21) does not have a closed-form solution. As such, we have adopted an iterative approach whereby first-order filters are used to initialize the design of the second-order filters. Specifically, assuming the first-order solutions for the prefilter $\vec{p}$ and the derivative filter $\vec{d}_1$, the error functional given in (21) may be simplified to

$$E\{D_2\} = \int_\omega \left| \left(j^2\omega^2 P(\omega) - D_2(\omega) \right) \right|^2. \tag{28}$$

As before, a discrete version of the error functional on the filter values is written as

$$E(\vec{d}_2) = \left| j^2\omega^2 F_s \vec{p} - F_s \vec{d}_2 \right|^2 \tag{29}$$

where $\vec{d}_2$ contains one half of the full filters taps. The minimum of this quadratic error function is easily seen to be

$$\vec{d}_2 = F_s^t j^2 \omega^2 F_s \vec{p}. \tag{30}$$

This filter, along with the previously designed prefilter and first-order filter, are used to initialize a gradient descent minimization of the full error functional of (21).

C. Higher Order

It is relatively straightforward to generalize the filter design from the previous two sections to filters of arbitrary order. Specifically, the design of a set of first- through Nth-order derivative filters begins by designing a prefilter and first-order derivative pair (Section III-A). The second- through Nth-order filters are then designed in an iterative fashion, each initialized with the design of the previous orders. In general, given the prefilter and first- through N-1st derivative filters, the error functional in (23) reduces to the constraint $|j^N\omega^N F_s\vec{p} - F_s\vec{d}_N|^2$, from which the Nth-order derivative filter is initially estimated as

$$\vec{d}_N = F_s^t j^N \omega^N F_s \vec{p} \tag{31}$$

for even-order derivatives; for odd-order, the estimate is given by $F_a^t j^N \omega^N F_s \vec{p}$. This estimate, along with the lower order derivative filters, are used to initialize a gradient descent minimization of the (nonlinear) error functional that describes the desired derivative relationships [e.g., (23)].

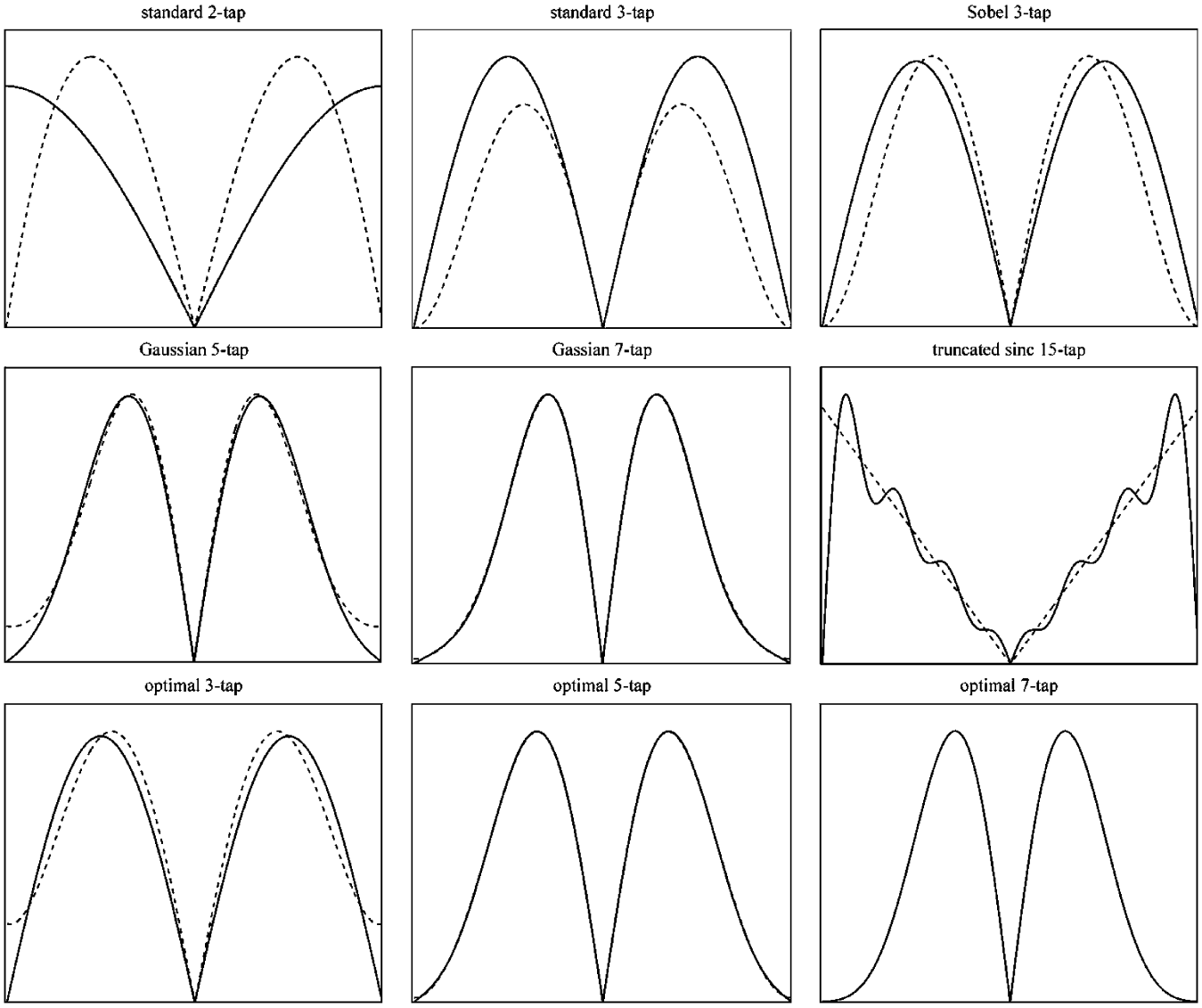

Fig. 4. First order derivatives: Fourier magnitude, plotted from $-\pi$ to π. The dashed line corresponds to the product of a ramp and prefilter. The solid line corresponds to the derivative filter.

D. Summary

A summary of the algorithm is as follows.

1) Choose dimensionality D, order N, and kernel size L.
2) Choose number of Fourier samples $K \gg L$.
3) Precompute the Fourier matrices F_s and F_a.
4) Solve for first-order filters of size $\min(L, 11)$, using (25).
5) If $L > 11$, solve for unique filters of size L by numerically optimizing (27), initialized with the $L = 11$ solution.
6) For each order $n = 2$ up to N:
   a) solve for the nth-order filter using (31);
   b) numerically optimize (23) over all filter orders 1 through n, initializing with solution from part a), together with the filters designed for order $n - 1$.

IV. Results

We have designed filters of various sizes and orders—some example filter taps are given in Table I.[1] Each design produces a set of matched 1-D differentiation filters that are meant to be used jointly in computing the full set of separable derivatives of the chosen order. Note that when designing, for example, a second-order derivative, the accompanying first-order differentiation filter is not necessarily optimal with respect to the first-order constraint in isolation. Note also that although we have constrained all differentation filters in each set to be the same size, there is no inherent reason why different filters could not be designed with a different number of taps.

[1]Source code (MATLAB) is available at: [Online] http://www.cs.dartmouth.edu/farid/research/derivative.html.

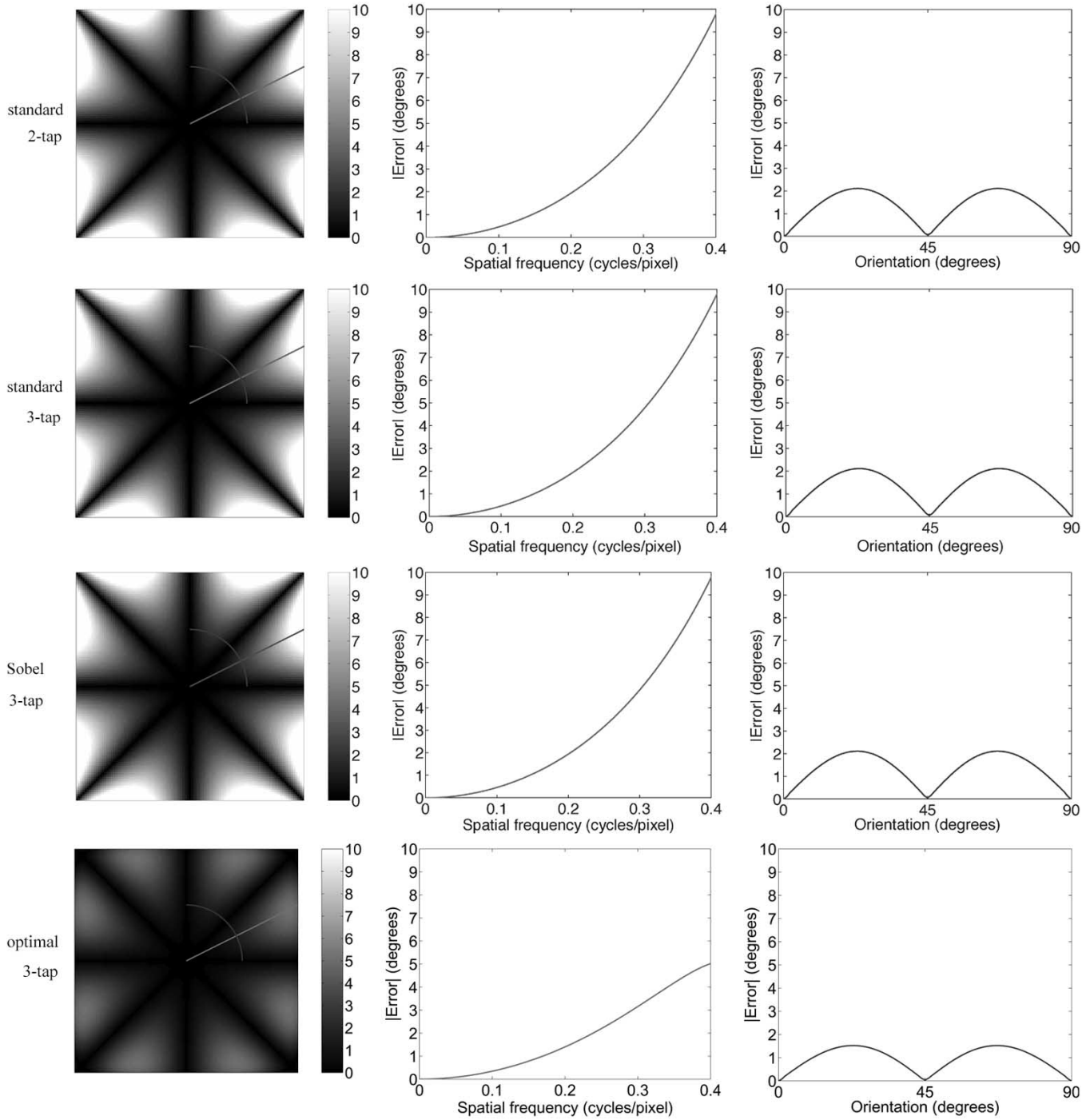

Fig. 5. First-order derivatives: errors in estimating the orientation of a sinusoidal grating of varying spatial frequencies and orientations. The intensity image in the left column shows the orientation error (brighter pixels correspond to larger errors) across all spatial frequencies—the axes span the range $[-\pi, \pi]$. The plots in the second and third columns correspond to the indicated radial and angular slices through this 2-D frequency space.

Shown in Fig. 3 are the horizontal and vertical derivatives and the synthesized directional derivative at an angle of $\pi/4$. Note that these filters are much closer to being rotation invariant than the sinc-based derivatives of Fig. 2. In the following sections, the accuracy of these filters is compared with a number of standard derivative filters.

A. First Order

Shown in Fig. 4 is the frequency response of several first-order derivative filters and the product of an imaginary ramp and the frequency response of their associated prefilters. If the filters were perfectly matched, as per our design criteria, then these responses should be identical. Shown are the responses from our optimally designed 3-, 5-, and 7-tap filters. For comparison, we show the responses of a variety of commonly used differentiators: a standard (binomial) 2-tap ($p = (1\ \ 1)/2$, $d_1 = (-1\ \ 1)$), a standard (binomial) 3-tap ($p = (1\ \ 2\ \ 1)/4$, $d_1 = (-1\ \ 0\ \ 1)/2$), a 3-tap Sobel operator [18] ($p = (1\ \ 2\ \ 1)/4$), $d = (-\sqrt{2}\ \ 0\ \ \sqrt{2})/3.5$), a 5- and 7-tap Gaussian ($p = 1/(\sqrt{2\pi}\sigma)e^{-x^2/2\sigma^2}$, $d = (-x/\sigma^2)p$), and a

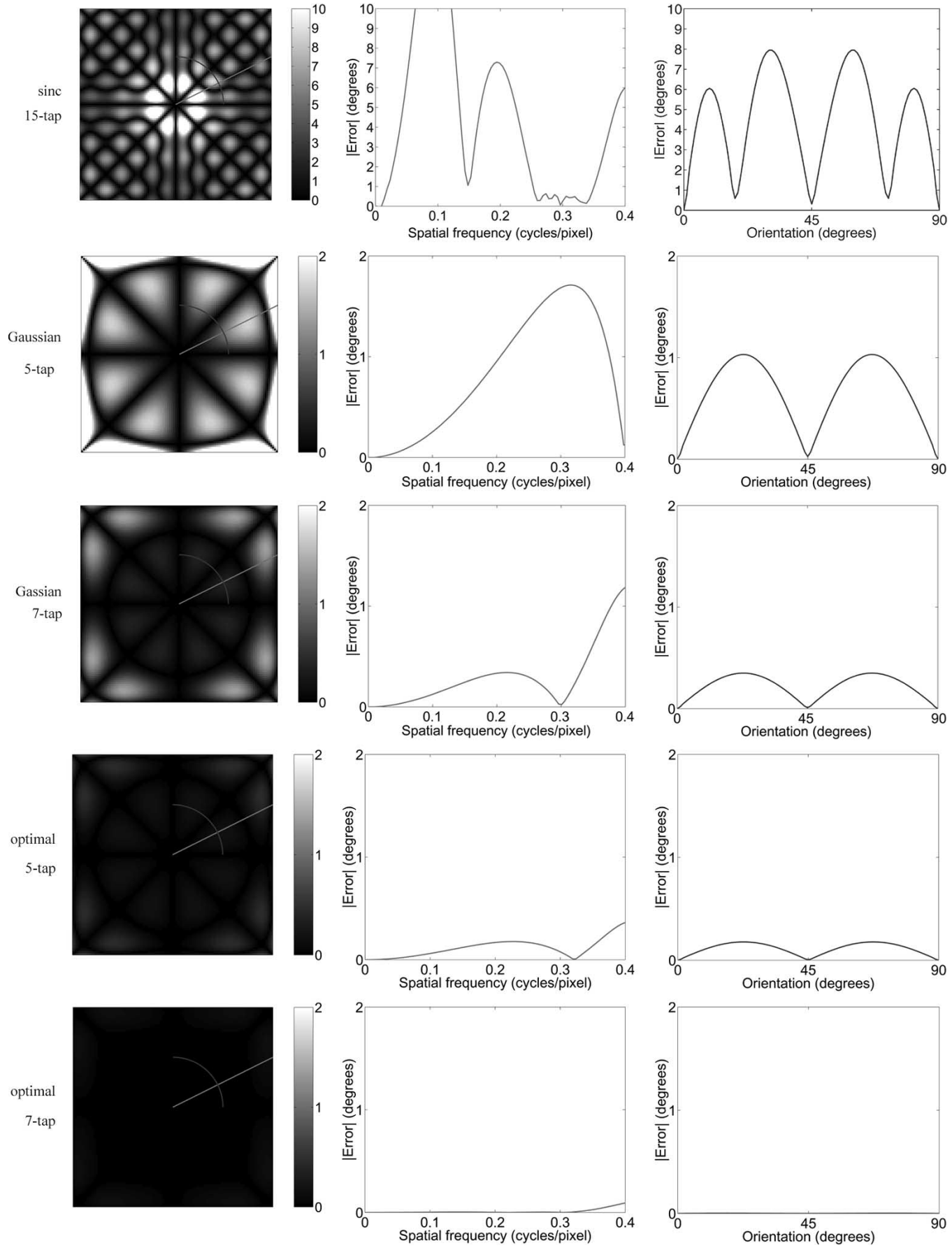

Fig. 6. First-order derivatives: errors in estimating the orientation of a sinusoidal grating of varying spatial frequencies and orientations (see caption of Fig. 5). Note that the scaling of the truncated sinc filter errors is different than the others.

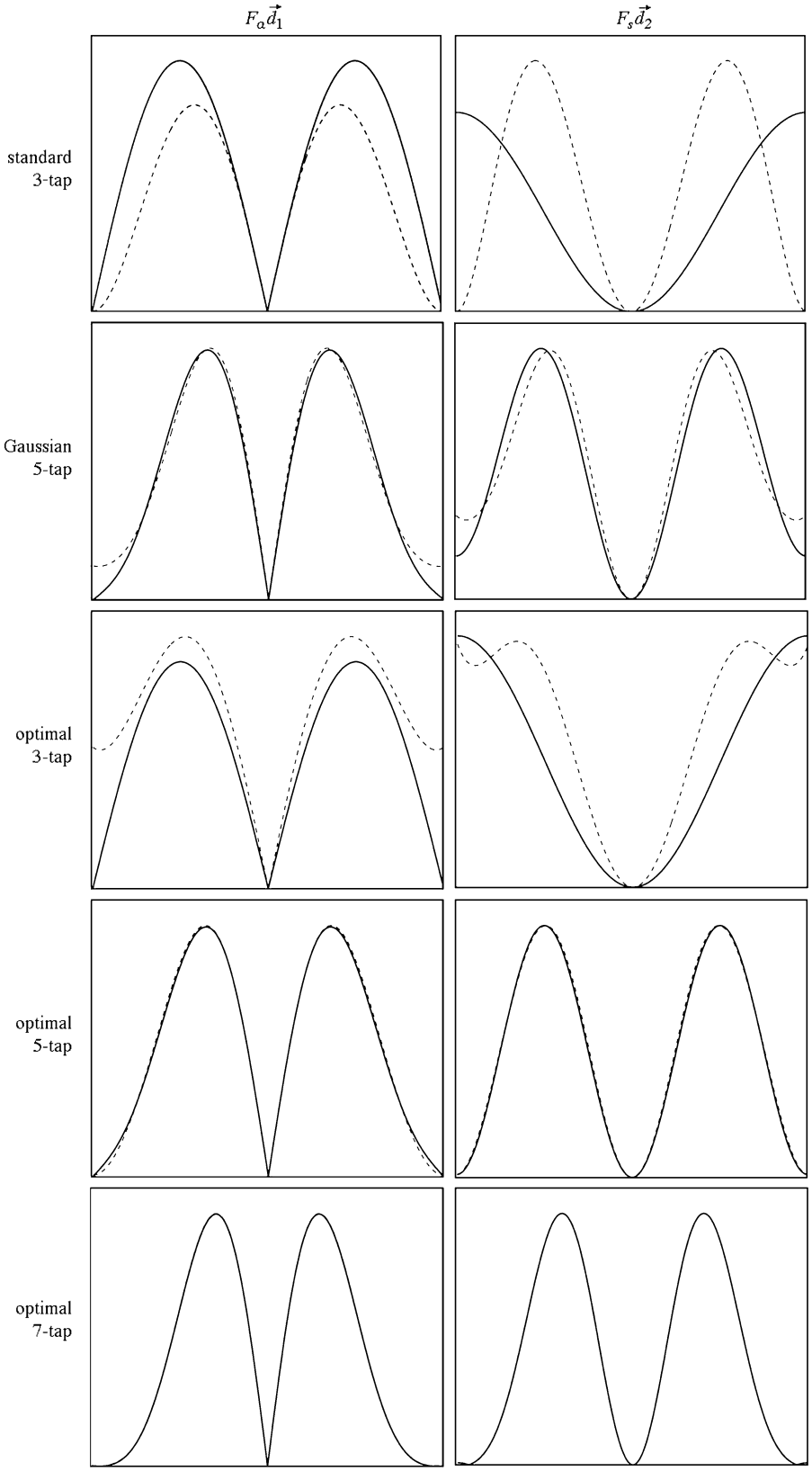

Fig. 7. Second-order derivatives: Fourier magnitudes, plotted from $-\pi$ to π. Left side: first-order filter (solid line), compared with product of a ramp and the prefilter (dashed line). Right side: second-order filter compared with product of squared ramp and prefilter.

15-tap truncated sinc function (see Fig. 1). The optimal filters of comparable (or smaller) support are seen to outperform all of the conventional filters, usually by a substantial margin. Note that the standard deviation σ of the 5- and 7-tap Gaussian filter was set to 0.96 and 1.12 pixels, respectively. These values minimize the rms error in the frequency response matching between the

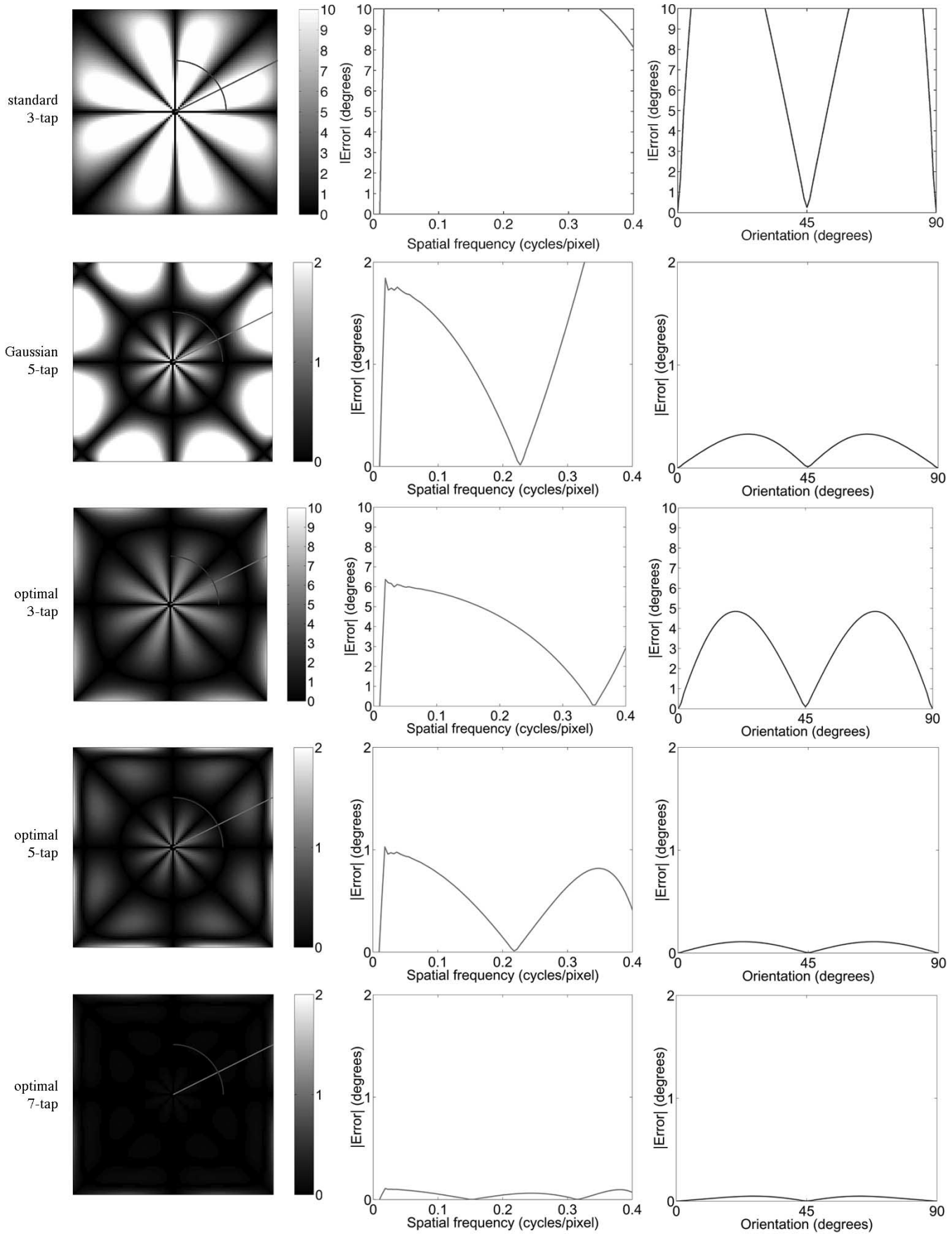

Fig. 8. Second-order derivatives: errors in estimating the orientation of a sinusoidal grating of varying spatial frequencies and orientations. Note that the scaling of the 3-tap filter's errors are different than the others.

prefilter and derivative filter, and thus are superior to most Gaussian differentiators used in the literature. For example, Barron *et al.* suggest a 5-tap Gaussian of width $\sigma = 1.5$ pixels [16].

Recall that our filters were designed so as to optimally respect the linear algebraic properties of multidimensional differentiation. This property is critical when estimating, for example, local orientation in an image (by taking the arctangent of the two components of the gradient). Figs. 5 and 6 show the errors in estimating the orientation of a 2-D sinusoidal grating, $g(x, y)$ of size 32×32 with horizontal and vertical spatial frequency ranging from 0.0078 to 0.4062 cycles/pixel. The orientation is computed as a least-squares estimate across the entire 32×32 image. Specifically, we solve for the orientation θ that minimizes

$$E(\theta) = \sum_{x,y} [g_x(x, y)\cos(\theta) + g_y(x, y)\sin(\theta)]^2$$

where the sum is taken over all image positions. The errors for our optimal filters are substantially smaller than all of the conventional differentiators.

B. Second Order

Shown in Fig. 7 is the frequency response of several first- and second-order derivative filters. Also shown for the first-order (second-order) derivative is the product of a negative parabola (squared imaginary ramp) and the frequency response of the prefilter. If the filters were perfectly matched, as per our design criteria, then these responses should be identical to the frequency responses of the corresponding derivative filters. Shown are the responses from our optimally designed 3-, 5-, and 7-tap filters. For comparison, we show the response from a standard 3-tap ($p = (0.25 \;\; 0.5 \;\; 0.25)$, $d_1 = (-0.5 \;\; 0.0 \;\; 0.5)$, $d_2 = (0.25 \;\; -0.5 \;\; 0.25)$), and a 5-tap Gaussian (as in the previous section the width of the Gaussian is 0.96 pixels).

Shown in Fig. 8 are the errors from estimating the orientation of a 2-D sinusoidal grating, $g(x, y)$, using second-order derivatives. As before, orientation estimation was performed on gratings of size 32×32 with horizontal and vertical spatial frequency ranging from 0.0078 to 0.4062 cycles/pixel. g_{xy} is computed by applying the first-order derivative filter in the horizontal and vertical directions, and g_{xx} (g_{yy}) is computed by applying the second-order derivative filter in the horizontal (vertical) direction and the prefilter in the vertical (horizontal) direction. The orientation is computed using a least-squares estimator across the entire 32×32 image. Specifically, we solve for the angle θ that minimizes

$$E(\theta) = \sum_{x,y} \big[g_{xx}(x, y)\sin^2(\theta) - g_{xy}(x, y)\sin(\theta)\cos(\theta) + g_{yy}(x, y)\cos^2(\theta)\big]^2$$

where the sum is taken over all positions in the image. Note that the errors for our optimal filters are substantially smaller than the standard and Gaussian filters.

V. Discussion

We have described a framework for the design of discrete multidimensional differentiators. Unlike previous methods, we formulate the problem in terms of minimizing errors in the estimated gradient direction, for a fixed size kernel. This emphasis on the accuracy of the *direction* of the gradient vector is advantageous for many multidimensional applications. The result is a matched set of filters—low-pass prefilter and differentiators up to the desired order—a concept first introduced in [19].

We have also enforced a number of auxiliary properties, including a fixed finite extent, symmetry, unit D.C. response (in the prefilter), and separability. Although we also tested nonseparable designs (see also [20]), the marginal improvement in accuracy appears to not be worth the considerably higher computational expense. Finally, we incorporated steerability as an orthogonal regularizing constraint for large filters, where the primary constraint was insufficient to give a unique solution.

A number of enhancements could be incorporated into our design method. We have ignored noise, and have not incorporated a prior model on images. Simple spectral models can easily be included, but a more sophisticated treatment might make the design problem intractable. We have also not explicitly modeled the image sensing process (e.g., [9], [10]). Again, simple spectral models could be included, but a more realistic treatment would likely prove impractical. The basic design method might also be optimized for specific applications (e.g., [21]).

Finally, the basic case of finite finite-impulse response (FIR) filter design can be extended in several ways. A number of authors have considered infinite-impulse response (IIR) solutions, for use in temporal differentiation [22], [23]. It would be interesting to consider the joint design of IIR and FIR filters for differentiation of video signals. It is also natural to combine multiscale decompositions with differential measurements (e.g., [24]–[27]), and thus might be worth considering the problem of designing a multiscale decomposition that provides optimized differentiation.

References

[1] B. Kumar and S. C. D. Roy, "Coefficients of maximally linear, FIR digital differentiators for low frequencies," *Electron. Lett.*, vol. 24, no. 9, pp. 563–565, Apr. 1988.

[2] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[3] J. le Bihan, "Maximally linear FIR digital differentiators," *J. Circuits, Syst., Signal Processsing*, vol. 14, no. 5, pp. 633–637, 1995.

[4] I. Selesnick, "Maximally flat lowpass digital differentiators," *IEEE Trans. Circuits Syst. II*, vol. 49, pp. 219–223, Mar. 2002.

[5] P. Danielsson, "Rotation-invariant linear operators with directional response," in *Proc. 5th Int. Conf. Pattern Recognition*, Miami, FL, Dec. 1980.

[6] J. Koenderink and A. vanDoorn, "Representation of local geometry in the visual system," *Biol. Cybern.*, vol. 55, pp. 367–375, 1987.

[7] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 891–906, Sept. 1991.

[8] J. M. S. Prewitt, *Object Enhancement and Extraction*. New York: Academic, 1970, pp. 75–149.

[9] R. M. Haralick, "Digital step edges from zero crossing of second directional derivatives," *IEEE Trans. Pattern Anal. Machine Intell*, vol. PAMI-6, pp. 58–68, Jan. 1984.

[10] T. Vieville and O. Faugeras, "Robust and fast computation of unbiased intensity derivatives in images," in *Proc. European Conf. Computer Vision*, Santa Margherita Ligure, Italy, 1992, pp. 203–211.

[11] J. de Vriendt, "Fast computation of unbiased intensity derivatives in images using separable filters," *Int. J. Comput. Vis.*, vol. 13, no. 3, pp. 259–269, 1994.
[12] M. Azaria, I. Vitsnudel, and Y. Y. Zeevi, "The design of two-dimensional gradient estimators based on one-dimensional operators," *IEEE Trans. Image Processing*, vol. 5, pp. 155–159, Jan. 1996.
[13] I. R. Khan and R. Ohba, "New design of full band differentiators based on Taylor series," *Proc. Inst. Elect. Eng. Visual Image Signal Processing*, vol. 146, no. 4, pp. 185–189, Aug. 1999.
[14] P. Meer and I. Weiss, "Smoothed differentiation filters for images," *J. Vis. Commun. Image Represent.*, vol. 3, no. 1, pp. 58–72, 1992.
[15] T. Lindeberg, "Discrete derivative approximations with scale-space properties: a basis for low-level feature extraction," *J. Math. Imaging Vis.*, vol. 3, no. 4, pp. 349–376, 1993.
[16] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 43–77, 1994.
[17] B. M. ter Haar Romeny, W. J. Niessen, J. Wilting, and L. M. J. Florack, "Differential structure of images: accuracy of representation," in *Proc. IEEE Int. Conf. Image Processing*, 1994, pp. 21–25.
[18] R. Gonzalez and R. Woods, *Digital Image Processing*. New York: Addison-Wesley, 1992.
[19] E. P. Simoncelli, "Design of multi-dimensional derivative filters," in *Proc. 1st Int. Conf. Image Processing*, vol. I. Austin, TX, Nov. 1994, pp. 790–793.
[20] H. Farid and E. Simoncelli, "Optimally rotation-equivariant directional derivative kernels," in *Int. Conf. Computer Analysis of Images and Patterns*, Kiel, Germany, Sept. 1997, pp. 207–214.
[21] M. Elad, P. Teo, and Y. Hel-Or, "Optimal filters for gradient-based motion estimation," in *Proc. Int. Conf. Computer Vision*, vol. 1, Kerkyra, Greece, Sept. 1999, pp. 559–565.
[22] B. Carlsson, A. Ahlen, and M. Sternad, "Optimal differentiators based on stochastic signal models," *IEEE Trans. Signal Processing*, vol. 39, pp. 341–353, Feb. 1991.
[23] D. Fleet and K. Langley, "Recursive filters for optical flow," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 61–67, Jan. 1995.
[24] G. H. Granlund, "In search of a general picture processing operator," *Comput. Graph. Image Processing*, vol. 8, pp. 155–173, 1978.
[25] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artificial Intelligence*, Vancouver, BC, Canada, 1981, pp. 674–679.
[26] P. Burt, "Pyramid methods in image processing," in *Proc. 1st Int. Conf. Image Processing*, Austin, TX, Nov. 1994.
[27] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: a flexible architecture for multi-scale derivative computation," in *Proc. 2nd Int. Conf. Image Processing*, vol. III, Washington, DC, Oct. 1995, pp. 444–447.

Hany Farid received the B.S. degree in computer science and applied mathematics from the Unversity of Rochester, Rochester, NY, in 1988 and the Ph.D. degree in computer science from the University of Pennsylvania, Philadelphia, in 1997.

He joined the faculty at Dartmouth College, Hanover, NH, in 1999, following a two year post-doctoral position in brain and cognitive sciences at the Massachusetts Institute of Technology, Cambridge.

Eero P. Simoncelli (S'92–M'93–SM'04) received the B.S. degree in physics from Harvard University, Cambridge, MA, in 1984, and the M.S. and Ph.D. degrees in 1988 and 1993, respectively, both in electrical engineering, from the Massachusetts Institute of Technology, Cambridge.

He was an Assistant Professor in the Computer and Information Science Department, University of Pennsylvania, Philadelphia, from 1993 to 1996. He moved to New York University, New York, in September 1996, where he is currently an Associate Professor in Neural Science and Mathematics. In August 2000, he became an Associate Investigator of the Howard Hughes Medical Institute, under their new program in computational biology. His research interests span a wide range of topics in the representation and analysis of visual images, in both machine and biological systems.