Fast space-variant elliptical filtering using box splines

Kunal Narayan Chaudhury, Arrate Muñoz-Barrutia, Michael Unser

Abstract

The efficient realization of linear space-variant (non-convolution) filters is a challenging computational problem in image processing. In this paper, we demonstrate that it is possible to filter an image with a Gaussian-like elliptic window of varying size, elongation and orientation using a fixed number of computations per pixel. The associated algorithm, which is based on a family of smooth compactly supported piecewise polynomials, the *radially-uniform box splines*, is realized using pre-integration and local finite-differences.

The radially-uniform box splines are constructed through the repeated convolution of a fixed number of box distributions, which have been suitably scaled and distributed radially in an uniform fashion. The attractive features of these box splines are their asymptotic behavior, their simple covariance structure, and their quasi-separability. They converge to Gaussians with the increase of their order, and are used to approximate anisotropic Gaussians of varying covariance simply by controlling the scales of the constituent box distributions. Based on the second feature, we develop a technique for continuously controlling the size, elongation and orientation of these Gaussian-like functions. Finally, the quasi-separable structure, along with a certain scaling property of box distributions, is used to efficiently realize the associated space-variant elliptical filtering, which requires $O(1)$ computations per pixel irrespective of the shape and size of the filter.

Index – Space-variant filter, Finite-differences, Running-sums, Anisotropic Gaussian, Box spline, Zwart-Powell (ZP) element.

1 INTRODUCTION

THE most widely used smoothing operator in image processing is the Gaussian filter. As far as isotropic Gaussians are concerned, a fast implementation is achievable simply by decomposing the filter into two orthogonal 1-D Gaussians operating along the image axes. The 1-D filters are in turn implemented using efficient recursive algorithms, e.g., the ones proposed by Deriche [5] and Young et al. [20]. We refer the readers to this survey article [16] for an exhaustive account of such recursive schemes.

A fundamental limitation of isotropic filtering is that it does not take into account the anisotropic nature of image features, which results in blurring of oriented patterns and textures. The development of fast anisotropic filtering in general, and anisotropic

Gaussian filtering in particular, have therefore gained momentum over the past decade. Worth mentioning in this regard is the work of Geusebroek et al. [7], who developed an efficient recursive technique based on the factorization of the 2D-anisotropic Gaussian into two 1-D Gaussians, one along the image axes and the other along a generally off-grid direction. A drawback of this technique is that one has to interpolate the image along the off-grid direction to implement the corresponding 1-D filter. To avoid interpolation and, in effect, to improve the spatial homogeneity and the Gaussian-like structure of the filters in [7], Lam et al. came up with the alternative "triple-axis" solution. Instead of using two directions, they chose to decompose the anisotropic Gaussian into three 1-D Gaussians operating along one ofthe four cardinal directions: the horizontal, the vertical, and the two diagonals [10]. The focus of these papers has largely been on space-invariant filtering, where the entire image is convolved with a single anisotropic Gaussian. On the other hand, a variety of space-variant filtering strategies have also been developed, including image statistics driven filtering [11], non-linear diffusion filtering [14, 19] and gradient inverse-weighted filtering [18], to name a few.

1.1 Linear space-variant filtering

In this paper, we focus on the paradigm of linear space-variant filtering using Gaussian-like kernels of different shapes and sizes. From a purely discrete[1] perspective, this calls for the design of a family of Gaussian filters $\{g_\lambda[\boldsymbol{n}]\}_\lambda$, so that, given an input image $f[\boldsymbol{n}]$, one can evaluate the filtered samples $\bar{f}[\boldsymbol{n}]$ through the averaging

$$\bar{f}[\boldsymbol{n}] = \sum_{\boldsymbol{k}} f[\boldsymbol{k}]\, g_{\lambda(\boldsymbol{n})}[\boldsymbol{n}-\boldsymbol{k}]. \tag{1}$$

The parameter $\lambda(\boldsymbol{n})$, which specifies the covariance of the filter applied at location $\boldsymbol{n}$, allows one to continuously adjust the scale, orientation and elongation of the filter in keeping with the anisotropy of the local image features. There are however certain practical problems involved in an efficient realization of (1). It is obvious that (1) cannot be written as a convolution, and hence cannot be realized using the FFT algorithm. In fact, the available options are either (i) to compute the filters $g_\lambda[\boldsymbol{n}]$ by sampling the continuous Gaussian on-the-fly, or (ii) to discretize λ a priori, and to store the pre-compute filters in a look-up table. The problem with the former is that it proves to be extremely slow for wide kernels, while the latter restricts the control on the anisotropy of the filters. By appropriately modifying the recursive filtering strategies in [5, 20], Tan et al. developed an algorithm for realizing (1) for the particular case where $\{g_\lambda[\boldsymbol{n}]\}_\lambda$ are isotropic [16].

Spline kernels can also yield efficient algorithms for space-variant filtering, particularly when the space-variance is in terms of the scale (or size) of the kernel. For instance, Heckbert proposed an algorithm for adaptive image blurring using tensor-products of polynomial splines, where the image is filtered with kernels of various scales using repeated integrations and finite-differences [8]. Based on similar principles,

[1] We associate the term "discrete" with functions defined on the Cartesian lattice $\mathbf{Z}^d$, where d is the dimensionality of the function.

namely, the scaling properties of B-splines, Muñoz-Barrutia et al. have developed an algorithm for fast computation of the continuous wavelet transform of 1-D signals [13]. Recently, the method was extended to perform space-variant filtering using Gaussian-like functions of arbitrary size, which can be elongated along the image axes [12]. To achieve this, the authors choose to approximate the Gaussian using separable B-splines. We propose to take this approach one step further. In particular, we overcome the limited steerability and ellipticity of the separable B-splines by considering certain quasi-separable analogues of B-splines, the so-called box splines [2]. As was demonstrated for the separable filters in [12], we show that these quasi-separable box splines can also be used to approximate the Gaussian, and that the associated space-variant filter can be decomposed into recursive pre-filters and scale-dependent finite difference filters. These together allow us to achieve a fast space-variant filtering of images using elliptical Gaussian-like filters.

To date, there have only been few applications of such multivariate splines in the fields of image processing and computer graphics. Noteworthy among them are the works of Richter [15] and Asahi et al. [1], concerning the development of image approximation and reconstruction algorithms, and that of Condat et al. [4] and Entezari et al. [6], concerning the development of interpolation formulas for hexagonal and BCC lattices.

1.2 Summary of the results

In this contribution, we propose a fast space-variant filtering algorithm using a family of Gaussian-like box splines whose size, elongation and orientation can be continuously controlled. The attractive feature of our approach is that we use a continuous-discrete formalism which avoids the necessity of sampling a continuously-defined filter on-the-fly, or of storing a discrete set of pre-computed kernels. The developments in the paper are centered around two main ideas, which are as follows:

(1) **The use of quasi-separable box splines**. The construction of bivariate box splines, conceived as the "shadow" of N-dimensional ($N \geq 2$) polytopes in 2-dimensions, often turns out to be rather intricate (see [2] for instance). In this paper, we consider an alternative straightforward recipe for constructing box splines, namely, through repeated convolutions of dilated and rotated box distributions (see Fig. 3). In particular, we realize the so-called radially-uniform box spline $\beta^N_{\mathbf{a}}(\boldsymbol{x})$ through the convolution of N rotated box distributions, where $\mathbf{a} = (a_1, \ldots, a_N)$ is a scale-vector with a_k being the scale of the box distribution positioned along the direction $(k-1)\pi/N$ (see §2.2 for a precise definition).

The reason why the radially-uniform box splines are of interest in the current context is twofold. The first of these is that we can make them arbitrarily close to a Gaussian by increasing N (see §2.2.3). The second reason, which has a more practical significance, is that we can continuously control their size, elongation, and orientation simply by acting on the scales.

(2) **An efficient strategy for space-variant averaging**. To convey this idea, we exam-

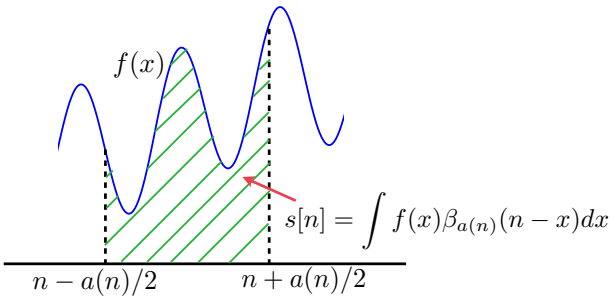


Figure 1: Computation of the space-variant average $\bar{f}[n]$: The signal $f(x)$ is first localized (hatched zone) using the shifted box function $\beta_{a(n)}(n-x)$, and then the area is computed. The central idea of our algorithm is to determine this area by taking the finite-difference of the primitive of $f(x)$.

ine the following formula

$$\bar{f}[n] = \frac{1}{2W(n)+1} \sum_{k=-W(n)}^{W(n)} f[n-k] \tag{2}$$

for computing the space-variant averages of a discrete 1-dimensional signal $f[n]$. We interpret the factor $W(n)$, which controls the amount of smoothing, as the size of the "discrete box filter" applied at location n. The disadvantage of using (2) is that its computational cost scales linearly with $W(n)$, which even gets worse in higher dimensions. This can be circumvented (with a mild interpolation cost) by considering instead the space-variant averaging

$$\bar{f}[n] = \frac{1}{a(n)} \int_{n-a(n)/2}^{n+a(n)/2} f(y)dy = \int f(y)\beta_{a(n)}(n-y)dy, \tag{3}$$

where we have replaced the discrete signal $f[n]$ by its interpolated version $f(x)$, and the discrete box filter by the box function

$$\beta_a(x) = \begin{cases} 1/a & \text{for } -a/2 < x \le a/2, \\ 0 & \text{otherwise.} \end{cases}$$

The main advantage of this formulation is that we can realize (3) using $O(1)$ computations per position, independent of the size of $a(n)$. This is based on the observation that (3) can computed by first evaluating the primitive

$$F(x) = \int_{-\infty}^{x} f(y)dy,$$

and then using the formula

$$\bar{f}[n] = \frac{1}{a(n)} \Big(F(n + a(n)/2) - F\big(n - a(n)/2\big) \Big), \tag{4}$$

which requires one addition and multiplication per position. This idea is illustrated in figure 1. The other advantage is that, as opposed to the integer-valued window $W(n)$ in (2), this gives access to the real-valued scale $a(n)$ for continuously controlling the smoothing. Indeed, if $f(x)$ is integrable (at least locally), then it can be shown that the use of small scales results in less smoothing, namely that $\bar{f}[n] \longrightarrow f[n]$ as $a(n) \longrightarrow 0$, whereas $\bar{f}[n]$ can be made negligibly small by making $a(n)$ sufficiently large.

The contribution of this paper is the generalization of the filtering strategy in (3) to the bivariate setting using the radially-uniform box splines. In particular, given a discrete image $f[\boldsymbol{n}]$, we consider the space-variant filtering

$$\bar{f}[\boldsymbol{n}] = \int f(\boldsymbol{y}) \beta^N_{\mathbf{a}(\boldsymbol{n})}(\boldsymbol{n} - \boldsymbol{y}) d\boldsymbol{y} \tag{5}$$

where $f(\boldsymbol{x})$ represents a suitable interpolation of $f[\boldsymbol{n}]$. The significance of the quasi-separable characterization of $\beta^N_{\mathbf{a}}(\boldsymbol{x})$ in terms of the box distributions is that it allows us to relate (5) to the 1-D problem in (3). Indeed, we demonstrate in §2.2 that (5) can be implemented using an appropriate bivariate extension of pre-integrations and finite-differences, together with few evaluations of a fixed piecewise polynomial function (the coefficients of which are pre-computed). Although the derivation of the algorithm is rather involved, the final solution turns out to be remarkably simple (see §3, Algorithm 1), and easy to implement.

1.3 Notations

We use $\hat{f}(\boldsymbol{\omega})$ to denote the Fourier transform of a function $f(\boldsymbol{x})$ on $\mathbf{R}^d$, specified by $\hat{f}(\boldsymbol{\omega}) = \int_{\mathbf{R}^d} f(\boldsymbol{x}) \exp(-j\boldsymbol{\omega}^T \boldsymbol{x}) d\boldsymbol{x}$. We use $f(\cdot - \boldsymbol{s})$ to denote the function obtained by translating $f(\boldsymbol{x})$ by $\boldsymbol{s}$. The convolution of two functions $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ is given by $(f * g)(\boldsymbol{x}) = \int_{\mathbf{R}^d} f(\boldsymbol{s}) g(\boldsymbol{x} - \boldsymbol{s}) d\boldsymbol{s}$. The notation $\circledast_{k=1}^N f_k(\boldsymbol{x})$ is used to denote the convolution of a collection of functions $f_1(\boldsymbol{x}), \ldots, f_N(\boldsymbol{x})$; the order of the convolutions is immaterial. We suppress the domain of an integral (or summation) if this is obvious from the context. For a bivariate function $f(\boldsymbol{x}) = f(x_1, x_2)$, the partial derivative along x_i is denoted by $\partial_i f(\boldsymbol{x})$. Given operators T_1 and T_2 on a domain $\mathscr{D}$, we use $T_1 \circ T_2$ (often simply $T_1 T_2$) to denote their composition: $(T_1 \circ T_2)(f) = T_1(T_2(f))$ for every f in $\mathscr{D}$. We use $\mathbf{M}^n$ to denote the $(n-1)$-fold matrix multiplication of $\mathbf{M}$ with itself. The integral $\int \mathbf{M}(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x}$, corresponding to a real-valued function $f(\boldsymbol{x})$ and a matrix-valued function $\mathbf{M}(\boldsymbol{x})$ on $\mathbf{R}^2$, denotes a matrix of the same dimension as $\mathbf{M}(\boldsymbol{x})$, whose (i,j)-th component is given by $\int \mathbf{M}_{i,j}(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x}$. If $\mathbf{P}$ and $\mathbf{Q}$ are constant matrices, we then have $\int \mathbf{P}\mathbf{M}(\boldsymbol{x})\mathbf{Q} f(\boldsymbol{x}) d\boldsymbol{x} = \mathbf{P}(\int \mathbf{M}(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x})\mathbf{Q}$. The notation $f(x) = O(g(x)), x \longrightarrow 0$, signifies that there exists a constant C (independent of x) such that $|f(x)| \leq C g(x)$ for sufficiently small x. The space of bivariate finite-energy signals is denoted by $\mathbf{L}^2(\mathbf{R}^2)$, or simply as $\mathbf{L}^2$; it is equipped with the norm $\|f\|_{\mathbf{L}^2} = [\int |f(\boldsymbol{x})|^2 d\boldsymbol{x}]^{1/2}$. The Dirac distribution is denoted by $\delta(\boldsymbol{x})$.

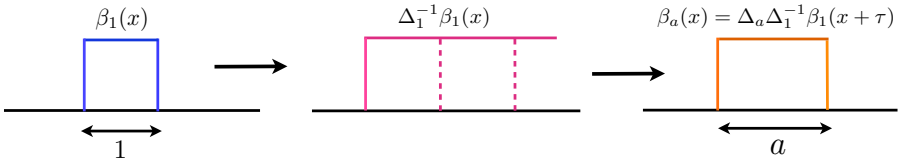


Figure 2: Box function rescaling through "addition and subtraction" of the unit box function: The step function is first reproduced from the unit box using the running-sum, and then the appropriate finite-difference is applied to recover the rescaled box function.

2 SPACE-VARIANT AVERAGING

We now derive (4) using an operator-based formalism. This helps set up the framework needed for the subsequent generalization of the idea to higher dimensions and multiple orders in §2.2.2.

2.1 Realization of (3)

We consider the *finite-difference* (FD) and the *running-sum* (RS) of a function $f(x)$, which are respectively defined by

$$\Delta_a f(x) = \frac{1}{a}\big(f(x) - f(x-a)\big), \tag{6}$$

and

$$\Delta_b^{-1} f(x) = b\sum_{k=0}^{\infty} f(x-bk). \tag{7}$$

The positive real numbers a and b are the scales of the operators[2]. We note that the operators Δ_a and Δ_b^{-1}, which takes $f(x)$ into $\Delta_a f(x)$ and $\Delta_b^{-1} f(x)$ respectively, are linear and translation-invariant, and that when b is an integer, Δ_b^{-1} can also be applied to a sequence $f[n]$ through the well-defined operation $\Delta_b^{-1} f[n] = b\sum_{k=0}^{\infty} f[n-bk]$. In particular, $g[n] = \Delta_1^{-1} f[n]$ can be implemented efficiently using the simple recursion $g[n] = g[n-1] + f[n]$, under appropriate boundary conditions [13].

The significance of these operators is that we can relate the variable-size box functions in (3) to the unit-width box function using the transformation

$$\beta_a(x) = \Delta_a\Delta_1^{-1}\beta_1(x+\tau) \tag{8}$$

where $\tau = (a-1)/2$. In particular, this means that box functions of variable widths can be derived from a fixed box function through the successive applications of running-sums and finite-differences (see Fig. 2). To derive (8), we note that $\Delta_1^{-1}\beta_1(x) =$

[2]The notation Δ_b^{-1} is justified by the fact that $\Delta_a\Delta_b^{-1}$ acts as the identity operator when $a = b$.

$\sum_{k=0}^{\infty} \beta_1(x-k) = u(x+1/2)$, where the step function $u(x)$ equals 1 for $x > 0$ and zero otherwise. The desired result follows immediately:

$$\Delta_a \Delta_1^{-1} \beta_1(x+\tau) = \frac{1}{a}\big(u(x+a/2) - u(x-a/2)\big) = \beta_a(x).$$

We use (8) to derive the algorithm for computing (3) as follows: Fix an arbitrary position n and the corresponding $a(n)$ in (3), and consider the function

$$s(x) = \int f(y)\beta_1(x-y)dy.$$

We claim that $\bar{f}[n] = \Delta_{a(n)}\Delta_1^{-1} s(n+\tau)$. Indeed, following the linearity and translation-invariance of $\Delta_{a(n)}\Delta_b^{-1}$, and using (8), we can write

$$\begin{aligned}\Delta_{a(n)}\Delta_1^{-1} s(x+\tau) &= \int f(y)[\Delta_{a(n)}\Delta_1^{-1}\beta_1(x+\tau-y)]dy \\ &= \int f(y)\beta_{a(n)}(x-y)dy,\end{aligned}$$

which establishes our claim.

Now if the input signal is discrete, of the form $f(x) = \sum_{n\in\mathbf{Z}} f[n]\delta(x-n)$, then $s(x)$ can simply be written as $s(x) = \sum f[n]\beta_1(x-n)$. A simple manipulation then shows that $\Delta_1^{-1} s(x) = \sum g[n]\beta_1(x-n)$, where $g[n]$ is the running-sum of $f[n]$. Thus, denoting the piecewise-constant interpolation of $g[n]$ by $F(x)$, we obtain

$$\bar{f}[n] = \Delta_{a(n)} F(n+\tau).$$

This leads us to the following two-step algorithm for realizing (3):

(1) (**Space-invariant step**) Compute $g[n] = \Delta_1^{-1} f[n]$ using the recursion $g[n] = g[n-1] + f[n]$;

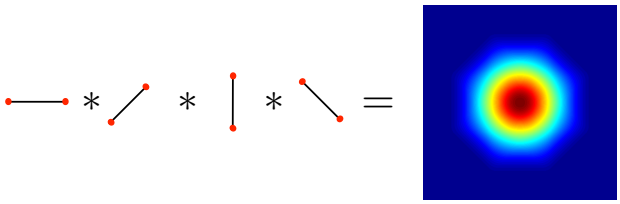
(2) (**Space-variant step**) For every position n, set $\bar{f}[n] = \Delta_{a(n)} F(n+\tau)$, where $\tau = (a(n)-1)/2$.

The steps of the algorithm can be visualized for the particular case when the input is an impulse and when $a(n) = a$ for every n using Fig. 2. The second and third plots in this figure then correspond to steps (1) and (2) of the algorithm, respectively.

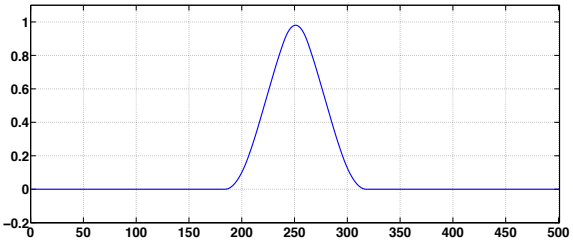
The remarkable fact about the algorithm is that the space-variant aspect of the transformation $f[n] \mapsto \bar{f}[n]$ gets transferred to the scale-dependent operator Δ_a, which in turn is implemented at a fixed computational cost per pixel, namely, one addition and multiplication per position. We would also like to note that (8) can more generally be expressed as

$$\beta_a(x) = \Delta_a \Delta_b^{-1} \beta_b(x+\tau) \qquad (\tau = (a-b)/2). \tag{9}$$

The significance of this relation is that, if the lattice spacing b is different from unity, one can still realize the running-sum (without interpolation) by replacing the operator Δ_1^{-1} by Δ_b^{-1}. We will use this in the sequel.



(A) Four-directional box spline



(B) Scan profile along $\theta = \pi/8$

Figure 3: Construction of the radially-uniform box spline through the convolution of four directional box distributions. (**A**) The four box distributions, distributed uniformly over $[0, \pi)$, were assigned equal scales in this example; (**B**) Scan profile along $\theta = \pi/8$.

2.2 Bivariate extension

2.2.1 Radially-uniform box splines

We now extend the space-variant filtering strategy discussed in the previous section to the bivariate setting, where the additional notion of directionality is appropriately addressed. As a first step, we devise an appropriate directional extension of the box function. In particular, corresponding to a real-valued scale a and direction θ, we define the *directional box distribution* $\varphi_{a,\theta}(\boldsymbol{x})$ as the tensor product of the box function $\beta_a(x)$ and the Dirac distribution $\delta(x)$ operating along orthogonal directions,

$$\varphi_{a,\theta}(\boldsymbol{x}) = \beta_a\left(\boldsymbol{u}_\theta^T \boldsymbol{x}\right) \delta\left(\boldsymbol{u}_{\theta_\perp}^T \boldsymbol{x}\right).$$

Here the orthogonal directions are specified by the unit vectors $\boldsymbol{u}_\theta = (\cos\theta, \sin\theta)$, and $\boldsymbol{u}_{\theta_\perp} = (-\sin\theta, \cos\theta)$. The scale a controls the amount of smoothing applied along the orientation of the box distribution, whereas no smoothing is applied along the transverse direction. The idea then is to construct the box spline by convolving an arbitrary number of such directional box distributions (cf. Fig 3). Thus, corresponding to an integer $N > 1$, a set of real-valued scales $a_1, \ldots, a_N$, and uniform rotation-angles $\theta_k = (k-1)\pi/N, k = 1, \ldots, N$, we specify the *radially-uniform box spline* through the convolution

$$\beta_{\mathbf{a}}^N(\boldsymbol{x}) = (\varphi_{a_1,\theta_1} * \cdots * \varphi_{a_N,\theta_N})(\boldsymbol{x}). \tag{10}$$

We shall refer to N and the tuple $\mathbf{a} = (a_1, \ldots, a_N)$ as the directional-order and the scale-vector of the box spline, respectively.

Following definition (10), it can be verified that $\beta^N_{\mathbf{a}}(\boldsymbol{x})$ is a piecewise polynomial of degree $\leq N-2$, where the partitions are specified by lines running along the directions $\theta_1,\ldots,\theta_N$. Moreover, $\beta^N_{\mathbf{a}}(\boldsymbol{x})$ is symmetric with respect to the origin, and is compactly supported on a convex N-sided polygon consisting of the points

$$\Big\{\sum_{k=1}^{N} t_k a_k \boldsymbol{u}_{\theta_k} : -1/2 \leq t_k \leq 1/2\Big\}.$$

The radially-uniform box splines are non-separable for $N>2$. However, in keeping with the spirit of the underlying tensor construction, the term quasi-separable would be more appropriate. The scale-vector $\mathbf{a}$ plays a vital role in determining the size and shape of the box spline. It is clear that the box spline can be arbitrarily elongated along the principal directions θ_k $(1\leq k\leq N)$ simply by rescaling the box distribution φ_{a_k,θ_k}, that is by making a_k large compared to the other scales. Moreover, we will demonstrate in the sequel that one can elongate the box spline along *any* arbitrary direction by appropriately acting on the scale-vector. The role of the directional-order is more subtle; it determines the degrees of freedom available for controlling the geometry of the box spline and also its regularity (smoothness). We will discuss these aspects in detail for the particular four-directional box spline ($N=4$) in §2.2.3.

2.2.2 Realization of (5)

We now formulate the algorithm for realizing (5) by appropriately extending the domain of definition of the FD and the RS operator to bivariate functions. The main idea is to derive a relation similar to (8) for the radially-uniform box splines. Thus, corresponding to positive real-valued scales a and b, and direction $0\leq\theta<\pi$, we consider the *directional finite-difference*

$$\Delta_{a,\theta} f(\boldsymbol{x}) = \frac{1}{a}\Big(f(\boldsymbol{x}) - f(\boldsymbol{x}-a\boldsymbol{u}_\theta)\Big), \tag{11}$$

and the *directional running-sum*

$$\Delta^{-1}_{b,\theta} f(\boldsymbol{x}) = b\sum_{k=0}^{\infty} f(\boldsymbol{x}-kb\boldsymbol{u}_\theta). \tag{12}$$

In keeping with the definition of the box spline, the radially-uniform FD and RS operators, $\Delta^N_{\mathbf{a}}$ and $\Delta^{-N}_{\mathbf{b}}$, are then specified by the combined action of (11) and (12) along the directions $\theta_k=(k-1)\pi/N$. In particular, we set

$$\Delta^N_{\mathbf{a}} = \Delta_{a_1,\theta_1}\circ\cdots\circ\Delta_{a_N,\theta_N}, \tag{13}$$

and

$$\Delta^{-N}_{\mathbf{b}} = \Delta^{-1}_{b_1,\theta_1}\circ\cdots\circ\Delta^{-1}_{b_N,\theta_N}, \tag{14}$$

where the scale-vectors $\mathbf{a}=(a_1,\ldots,a_N)$ and $\mathbf{b}=(b_1,\ldots,b_N)$ specify the scale along each direction. The operators $\Delta^N_{\mathbf{a}}$ and $\Delta^{-N}_{\mathbf{b}}$ are closely related to the radially-uniform

box splines. Indeed, it can readily be verifed that

$$\Delta_{\mathbf{a}}^{N}\Delta_{\mathbf{b}}^{-N}=\Delta_{a_1,\theta_1}\Delta_{b_1,\theta_1}^{-1}\circ\cdots\circ\Delta_{a_N,\theta_N}\Delta_{b_N,\theta_N}^{-1},$$

and that using (9) we can write

$$\Delta_{a,\theta}\Delta_{b,\theta}^{-1}\varphi_{b,\theta}(\boldsymbol{x}+\tau\boldsymbol{u}_{\theta})=\varphi_{a,\theta}(\boldsymbol{x}).$$

Thus, if we let $\tau=\sum\tau_k\boldsymbol{u}_{\theta_k}$, where $\tau_k=(a_k-b_k)/2$, then following definitions (10), (13), and (14), we see that

$$\begin{aligned}
&\Delta_{\mathbf{a}}^{N}\Delta_{\mathbf{b}}^{-N}\beta_{\mathbf{b}}^{N}(\boldsymbol{x}+\tau)\\
&=\Delta_{a_1,\theta_1}\Delta_{b_1,\theta_1}^{-1}\circ\cdots\circ\Delta_{a_N,\theta_N}\Delta_{b_N,\theta_N}^{-1}\left[\circledast_{k=1}^{N}\varphi_{b_k,\theta_k}(\boldsymbol{x}+\tau_k\boldsymbol{u}_{\theta_k})\right]\\
&=\circledast_{k=1}^{N}\Delta_{a_k,\theta_k}\Delta_{b_k,\theta_k}^{-1}\varphi_{b_k,\theta_k}(\boldsymbol{x}+\tau_k\boldsymbol{u}_{\theta_k})\\
&=\circledast_{k=1}^{N}\varphi_{a_k,\theta_k}(\boldsymbol{x}).
\end{aligned}$$

This provides the following crucial connection between the box splines and the 2-D operators.

Proposition 2.1 *The box spline $\beta_{\mathbf{a}}^{N}(\boldsymbol{x})$ can be expressed as*

$$\beta_{\mathbf{a}}^{N}(\boldsymbol{x})=\Delta_{\mathbf{a}}^{N}\Delta_{\mathbf{b}}^{-N}\beta_{\mathbf{b}}^{N}(\boldsymbol{x}+\tau). \tag{15}$$

Before discussing the filtering algorithm, we briefly elaborate on the implementation of $\Delta_{\mathbf{a}}^{N}$ and $\Delta_{\mathbf{b}}^{-N}$. We can show that (13) can be written as

$$\Delta_{\mathbf{a}}^{N}f(\boldsymbol{x})=\sum_{i=0}^{2^N-1}w_i f(\boldsymbol{x}-\boldsymbol{x}_i), \tag{16}$$

where $w_i=(-1)^{q_1+\cdots+q_N}(a_1\cdots a_N)^{-1}$ and $\boldsymbol{x}_i=\sum_{k=1}^{N}q_k a_k\boldsymbol{u}_{\theta_k}$, and the index i taking values between 0 and (2^N-1) is the decimal counterpart of the binary number $(q_N,q_{N-1},\ldots,q_1)$, which takes values from $(0,\ldots,0)$ to $(1,\ldots,1)$. We identify the points $\boldsymbol{x}_i$ with the vertices of an affine mesh, and w_i with the corresponding mesh taps (cf. Fig. 5).

As far as the application of $\Delta_{\mathbf{b}}^{-N}$ to a discrete sequence $f[\boldsymbol{n}]$ is concerned, the unfortunate difficulty is that the vectors $b_k\boldsymbol{u}_{\theta}$ must necessarily lie on the lattice for (12) to be well-defined. In fact, it is easily seen that one cannot associate a digital filter with the RS operators in general. However, the good news is that, when N equals 2 and 4, the transformation $f[\boldsymbol{n}]\mapsto\Delta_{\mathbf{b}}^{-N}f[\boldsymbol{n}]$ can be exactly realized without the need for interpolation by appropriately setting the scales b_k of the directional running-sums. We will discuss the latter case in detail in §3.

The algorithm for realizing (5) corresponding to a specified scale-vector map $\mathbf{a}(\boldsymbol{n})$ is based on relation (15). In particular, by considering the function $f(\boldsymbol{x})=$

$\sum_n f[\boldsymbol{n}]\delta(\boldsymbol{x}-\boldsymbol{n})$, and by proceeding exactly along lines of the 1-D derivation, we express the filtered samples in (5) as

$$\bar{f}[\boldsymbol{n}] = \sum_{i=0}^{2^N-1} w_i F(\boldsymbol{n}+\tau-\boldsymbol{x}_i), \tag{17}$$

where $F(\boldsymbol{x}) = \sum g_{\mathbf{b}}[\boldsymbol{n}]\beta^N_{\mathbf{b}}(\boldsymbol{x}-\boldsymbol{n})$ denotes the interpolated version of the pre-integrated signal $g_{\mathbf{b}}[\boldsymbol{n}] = \Delta^{-N}_{\mathbf{b}} f[\boldsymbol{n}]$; $\tau = 0.5(\sum(a_k(\boldsymbol{n}) - b_k)\cos\theta_k, \sum(a_k(\boldsymbol{n}) - b_k)\sin\theta_k)$; and the pairs $(\boldsymbol{x}_i, w_i)$ are the vertices and taps of the affine FD mesh in (16). Note that τ, w_i and $\boldsymbol{x}_i$ are defined pointwise in (17); we dropped the index $\boldsymbol{n}$ to simplify the notation. We will discuss the implementation aspects of the algorithm, particularly the computation of $g_{\mathbf{b}}[\boldsymbol{n}]$ and its interpolated form $F(\boldsymbol{x})$ for the case $N = 4$ in §3.

2.2.3 Characterization of the radially-uniform box splines

The motivation behind introducing the radially-uniform box splines was to develop elliptical Gaussian-like filters, whose shape (size, elongation and orientation) can be continuously controlled, and a fast space-variant algorithm using such filters. Indeed, it turns out that the radially-uniform box splines (and its iterated versions) form close approximates of the Gaussian. To substantiate our claim we present the following result (proof in Appendix §6.1) that can well be seen as a "radial" version of the central limit theorem.

Theorem 2.2 *Consider the sequence of box splines $\beta^2_{\mathbf{a}(2)}(\boldsymbol{x}), \beta^3_{\mathbf{a}(3)}(\boldsymbol{x}), \ldots$ corresponding to the scale-vectors $\mathbf{a}(2), \mathbf{a}(3), \ldots$, where $a_k(N) = \sigma\sqrt{(24/N)}$ for $1 \le k \le N$. Then the following holds*

$$\lim_{N\longrightarrow\infty} \beta^N_{\mathbf{a}(N)}(\boldsymbol{x}) = \frac{1}{2\pi\sigma^2}\exp\Big(-\frac{\|\boldsymbol{x}\|^2}{2\sigma^2}\Big). \tag{18}$$

In fact, the radially-uniform box splines constructed using uniform scale-vectors are supported on a N-sided uniform polygon, and it can be shown that they have continuous derivatives of order $(N-3)$. The above result is then consistent with the fact that the isotropy and smoothness of such box splines progressively improves with the increase in the directional-order. Moreover, it is also possible to mimic certain anisotropic Gaussians by using a sequence of non-uniform scale-vectors. Indeed, as a direct extension of Theorem 2.2, one can construct sequences of box splines which converge to anisotropic Gaussians as N increases.

Yet another useful form of anisotropic convergence is achievable based on the serial convolutions of a radially-uniform box spline, of a fixed directional-order, with itself. In particular, corresponding to fixed integers N and m ($m \ge 1$), and a scale-vector $\mathbf{a} = (a_1, \ldots, a_N)$, we consider the *iterated radially-uniform box spline*

$$\beta^{N,m}_{\mathbf{a}}(\boldsymbol{x}) = (\beta^N_{\mathbf{a}} * \cdots * \beta^N_{\mathbf{a}})(\boldsymbol{x}) \tag{19}$$

obtained through the $(m-1)$-fold convolution of $\beta^N_{\mathbf{a}}(\boldsymbol{x})$ with itself. Then, for the particular sequence of box splines $\{\beta^{N,m}_{\mathbf{a}(m)}(\boldsymbol{x})\}$ corresponding to the scale-vectors

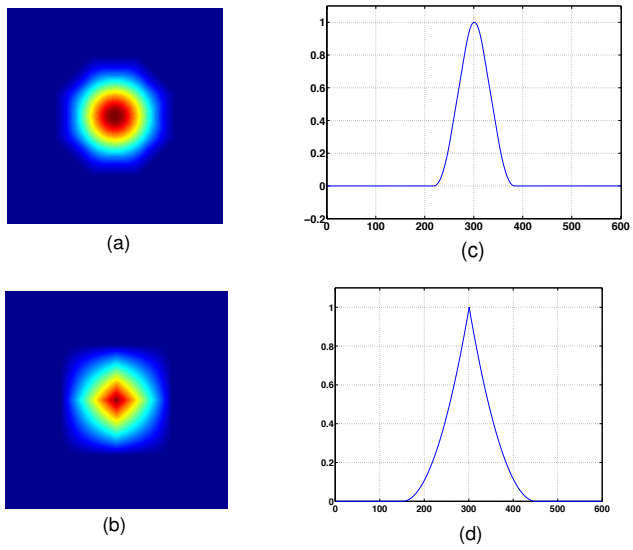


Figure 4: Intensity distribution of (a) the radially-uniform box spline, and (b) the separable B-spline, of order four. The respective scan profiles along $\pi/8$ are shown in (c) and (d).

$\mathbf{a}(m) = (a_1/\sqrt{m}, \ldots, a_N/\sqrt{m})$, we have the result

$$\lim_{m \longrightarrow \infty} \beta^{N,m}_{\mathbf{a}(m)}(\boldsymbol{x}) = \frac{1}{2\pi\,|\det(\mathbf{C})|^{1/2}} \exp\Big(-\frac{1}{2}\boldsymbol{x}^T \mathbf{C}^{-1}\boldsymbol{x}\Big), \tag{20}$$

where

$$\mathbf{C} = \frac{1}{12}\begin{pmatrix} \sum a_k^2 \cos^2\theta_k & \frac{1}{2}\sum a_k^2 \sin 2\theta_k \\ \frac{1}{2}\sum a_k^2 \sin 2\theta_k & \sum a_k^2 \sin^2\theta_k \end{pmatrix}. \tag{21}$$

Indeed, this follows directly from a certain version of the central limit theorem, which tells us that the each of the components

$$\circledast_{j=1}^{m}\, \varphi_{a_k/\sqrt{m},\theta_k}(\boldsymbol{x}) \qquad (1 \le k \le N)$$

converge to a "directional" Gaussian distribution as $m \longrightarrow \infty$. The covariance $\mathbf{C}$ is then given by the limiting sum of the covariances $\mathbf{C}_k$ of the constituent box distributions. The utility of such iterated box splines will be discussed in §3.3.

Having characterized the asymptotic behavior of the box splines, we now focus on the problem of approximating an anisotropic Gaussian using a box spline of a fixed directional-order. Since a centered Gaussian is uniquely specified by its covariance, we propose a finite-order box spline approximation of the same based on its covariance. This, in fact, amounts to a moment-based approximation of the Gaussian, that is,

the box spline resembles the Gaussian up to its second-order moments. Moreover, since the level-sets of Gaussians are ellipses, this equivalently amounts to constructing elliptical filters of different size, elongation, and orientation.

The covariance of the radially-uniform box spline, namely,

$$\mathbf{C}_{\mathbf{a}}^{N} = \int \boldsymbol{x}\boldsymbol{x}^{T} \beta_{\mathbf{a}}^{N}(\boldsymbol{x}) d\boldsymbol{x},$$

can be expressed as the sum of the covariances of the box distributions (cf. Appendix §6.2 for details) as follows:

$$\mathbf{C}_{\mathbf{a}}^{N} = \frac{1}{12} \begin{pmatrix} \sum a_k^2 \cos^2\theta_k & \frac{1}{2}\sum a_k^2 \sin 2\theta_k \\ \frac{1}{2}\sum a_k^2 \sin 2\theta_k & \sum a_k^2 \sin^2\theta_k \end{pmatrix}. \tag{22}$$

This provides the explicit dependence of $\mathbf{C}_{\mathbf{a}}^{N}$ on the scale-vector. In particular, based on the eigen decomposition of $\mathbf{C}_{\mathbf{a}}^{N}$, we propose the following characterization of the elliptical parameters of the box spline: Let $\lambda_{\max}$ and $\lambda_{\min}$ denote the largest and smallest eigenvalues of $\mathbf{C}_{\mathbf{a}}^{N}$, and (v_1, v_2) the eigenvector corresponding to the eigenvalue $\lambda_{\max}$. The size $s_{\mathbf{a}}^{N}$, elongation $\varrho_{\mathbf{a}}^{N}$, and orientation $\theta_{\mathbf{a}}^{N}$ of the radially-uniform box spline $\beta_{\mathbf{a}}^{N}(\boldsymbol{x})$ are then defined as

$$\begin{aligned} s_{\mathbf{a}}^{N} &= \lambda_{\max} + \lambda_{\min} = \frac{1}{12}\sum a_k^2, \\ \varrho_{\mathbf{a}}^{N} &= \frac{\lambda_{\max}}{\lambda_{\min}} = \frac{\sum a_k^2 + \sqrt{D}}{\sum a_k^2 - \sqrt{D}}, \\ \theta_{\mathbf{a}}^{N} &= \tan^{-1}\left(\frac{v_2}{v_1}\right) = \tan^{-1}\left(\frac{-\sum a_k^2 \cos(2\theta_k) + \sqrt{D}}{\sum a_k^2 \sin(2\theta_k)}\right), \end{aligned} \tag{23}$$

where $D = \left(\sum a_k^2 \cos 2\theta_k\right)^2 + \left(\sum a_k^2 \sin 2\theta_k\right)^2$.

Since $\mathbf{C}_{\mathbf{a}}^{N}$ is strictly positive (see Appendix §6.2), all the above parameters are indeed well-defined. Note that the covariance matrix in (22) and the triple in (23) provide equivalent descriptions of the box spline geometry. The motivation behind introducing the latter is its convenient rotation-invariant nature: while $\mathbf{C}_{\mathbf{a}}^{N}$ changes with the rotations of a given box spline, $s_{\mathbf{a}}^{N}$ and $\varrho_{\mathbf{a}}^{N}$ remains fixed. It is for this reason that we use the latter description for studying the dependence of the elliptical geometry on the scale-vector in the next section.

3 FOUR-DIRECTIONAL BOX SPLINES

We now study the particular *four-directional* box spline

$$\beta_{\mathbf{a}}^{4}(\boldsymbol{x}) = (\varphi_{a_1,0} * \varphi_{a_2,\pi/4} * \varphi_{a_3,\pi/2} * \varphi_{a_4,3\pi/4})(\boldsymbol{x}),$$

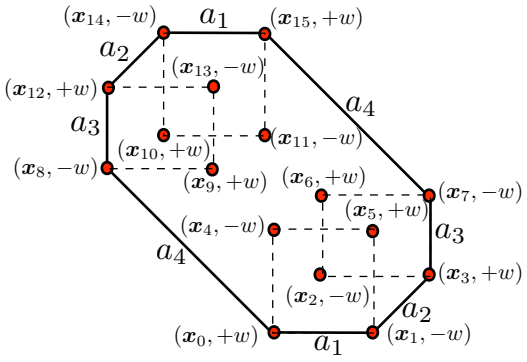


Figure 5: The distribution of the taps of the FD mesh. The pairs (u, v) denote the positions (u) and the corresponding weights (v) of the taps of the FD mesh.

and the corresponding implementation aspects. This particular box spline is composed of patches of quadratic polynomials (degree ≤ 2), is continuously differentiable, and is compactly supported on a convex octagon (cf. Fig. 3).

We note that in [12] the authors have used separable B-splines to approximate the Gaussian. Although these functions are built from the same constituent box distributions, the advantage of the four-directional box spline over the separable ones is that they are more isotropic. As seen in Fig. 4, the basic four-directional box spline, besides having a smaller support, exhibits a more Gaussian-like profile than the separable counterpart of identical order. In addition, the anisotropic four-directional box spline can be rotated to arbitrary orientations, while the separable ones are constrained to the image axes.

3.1 Fast space-variant elliptical filtering

The four-directional box spline is of particular interest in the context of the space-variant filtering following the fact that $\Delta_{\mathbf{b}}^{-4}$ can be implemented without interpolation when $\mathbf{b} = (1, \sqrt{2}, 1, \sqrt{2})$. The corresponding interpolating function, $\beta_{\mathbf{b}}^4(\boldsymbol{x})$, turns out to be well-known in the box spline community, and is popularly referred to as the Zwart-Powell (ZP) element [2, 21]. The steps for realizing (5) using the four-directional box spline are as follows:

(1) (**Pre-integration**) The crucial point is the choice of the scale-vector $\mathbf{b} = (1, \sqrt{2}, 1, \sqrt{2})$ corresponding to which the RS operator

$$\Delta_{\mathbf{b}}^{-4} = \Delta_{1,0}^{-1} \circ \Delta_{\sqrt{2},\pi/4}^{-1} \circ \Delta_{1,\pi/2}^{-1} \circ \Delta_{\sqrt{2},3\pi/4}^{-1}$$

can be directly applied to $f[\boldsymbol{n}]$. In particular, the running-sum $g_{\mathbf{b}}[\boldsymbol{n}] = \Delta_{\mathbf{b}}^{-4} f[\boldsymbol{n}]$ can be computed using the following four steps.

(RS1) Horizontal running-sum:

$$g_0[n_1, n_2] = \Delta_{1,0}^{-1} f[n_1, n_2] = \sum_{k=0}^{\infty} f[n_1 - k, n_2].$$

(RS2) First-diagonal running-sum:

$$g_{\pi/4}[n_1, n_2] = \Delta_{\sqrt{2},\pi/4}^{-1} g_0[n_1, n_2] = \sqrt{2} \sum_{k=0}^{\infty} g_0[n_1 - k, n_2 - k].$$

(RS3) Vertical running-sum:

$$g_{\pi/2}[n_1, n_2] = \Delta_{1,\pi/2}^{-1} g_{\pi/4}[n_1, n_2] = \sum_{k=0}^{\infty} g_{\pi/4}[n_1, n_2 - k].$$

(RS4) Second-diagonal running-sum:

$$g_{\mathbf{b}}[n_1, n_2] = \Delta_{\sqrt{2},3\pi/4}^{-1} g_{\pi/2}[n_1, n_2] = \sqrt{2} \sum_{k=0}^{\infty} g_{\pi/2}[n_1 + k, n_2 - k].$$

(2) (**Finite-difference**) At each position $\boldsymbol{n}$, the FD mesh is computed using the scale-vector $\mathbf{a}(\boldsymbol{n})$. The weights w_i and the vertices $\boldsymbol{x}_i$ are listed in Table 1 with the convention that $a'_k = a_k/\sqrt{2}$ for $k = 2, 4$. The mesh has a total of $4 \times 4 = 16$ vertices; in particular, there are 4 clusters corresponding to the four boxes with 4 vertices per cluster, as shown in Fig. 5. The shift $\tau = (\tau_1, \tau_2)$ is given by $\tau_1 = (\sqrt{2}a_1 + a_2 - a_4 - \sqrt{2})/2\sqrt{2}$ and $\tau_2 = (a_2 + \sqrt{2}a_3 + a_4 - 3\sqrt{2})/2\sqrt{2}$. The filtered sample is then computed using the formula

$$\bar{f}[\boldsymbol{n}] = \sum_{i=0}^{15} w_i F(\boldsymbol{n} + \tau - \boldsymbol{x}_i). \tag{24}$$

The interpolation samples $F(\boldsymbol{x}) = \sum g_{\mathbf{b}}[\boldsymbol{n}] \beta_{\mathbf{b}}^4(\boldsymbol{x} - \boldsymbol{n})$ in (24) are computed efficiently by taking advantage of the piecewise polynomial structure of the compactly supported ZP element (see Appendix 6.6).

As in the 1-D setting, the running-sums are efficiently evaluated using recursions as summarized in Algorithm 1. The decisive computational advantage, especially for wider kernels, is derived from the fact that the number of vertices of the FD mesh is completely independent of the scale-vector. As a result, the algorithm has a fixed computational cost per pixel, modulo the cost of the running-sum and the interpolations (see Table 2).

Table 1: Specification of the taps of the FD mesh associated with the operator $\Delta_{\mathbf{a}}^4$. The weight w is given by $(a_1a_2a_3a_4)^{-1}$, where $\mathbf{a}=(a_1,a_2,a_3,a_4)$ is the corresponding scale-vector.

i	$\boldsymbol{x}_i$	w_i	i	$\boldsymbol{x}_i$	w_i
0	$(0,0)$	$+w$	8	$(-a_4',a_4')$	$-w$
1	$(a_1,0)$	$-w$	9	(a_1-a_4',a_4')	$+w$
2	(a_2',a_2')	$-w$	10	$(a_2'-a_4',a_2'+a_4')$	$+w$
3	(a_1+a_2',a_2')	$+w$	11	$(a_1+a_2'-a_4',a_2'+a_4')$	$-w$
4	$(0,a_3)$	$-w$	12	$(-a_4',a_3+a_4')$	$+w$
5	(a_1,a_3)	$+w$	13	(a_1-a_4',a_3+a_4')	$-w$
6	(a_2',a_3+a_2')	$+w$	14	$(a_2'-a_4',a_3+a_2'+a_4')$	$-w$
7	(a_1+a_2',a_3+a_2')	$-w$	15	$(a_1+a_2'-a_4',a_3+a_2'+a_4')$	$+w$

3.2 Size, elongation and orientation of the box splines

As was mentioned earlier, the size and shape of the radially-uniform box spline can be controlled by appropriately adjusting the scales of the constituent box distributions. In this regard, we now discuss the following: (i) the forward problem of controlling the anisotropy of the four directional box spline by acting on the scale-vector, and (ii) the inverse problem of uniquely specifying the scale-vector of the box spline corresponding to a given covariance (geometry). For notational ease, we shall henceforth drop the superscript $N=4$ when referring to the four-directional box spline and its related parameters.

3.2.1 Control on the anisotropy

The elliptical geometry of this box spline is specified using parameters defined in (23), namely,

$$s_{\mathbf{a}}=\frac{1}{12}\sum a_k^2,\quad \theta_{\mathbf{a}}=\tan^{-1}\left(\frac{a_3^2-a_1^2+\sqrt{D}}{a_2^2-a_4^2}\right),\ \text{and}\ \varrho_{\mathbf{a}}=\frac{\sum a_k^2+\sqrt{D}}{\sum a_k^2-\sqrt{D}},$$

where $D=(a_3^2-a_1^2)^2+(a_2^2-a_4^2)^2$. It turns out that the size and orientation can be arbitrarily controlled by adjusting the scale-vector. Indeed, the size can be easily manipulated by multiplying the scale-vector $\mathbf{a}$ with a constant factor, since this leaves both the orientation and elongation unchanged. The elongation can be arbitrarily controlled in the neighborhood of the four principal directions. However, there exists a finite upper bound on the elongation along other directions (cf. Appendix §6.3).

Proposition 3.1 *For every ϕ in $[0,\pi)$, there exists a scale-vector $\mathbf{a}$ such that $\theta_{\mathbf{a}}=\phi$. There is however a finite bound on the elongation, and is given by*

$$\sup\ \varrho_{\mathbf{a}}<U(\phi)=\frac{1+|\nu_\phi|+\sqrt{1+\nu_\phi^2}}{1+|\nu_\phi|-\sqrt{1+\nu_\phi^2}},\tag{25}$$

Algorithm 1 Space-variant elliptical filtering

1. Input: $f[\boldsymbol{n}]$ and $\mathbf{a}(\boldsymbol{n})$
2. Perform recursions:

$$g_0[n_1,n_2] \leftarrow f[n_1,n_2] + g_0[n_1-1,n_2]$$
$$g_{\pi/4}[n_1,n_2] \leftarrow \sqrt{2}g_0[n_1,n_2] + g_{\pi/4}[n_1-1,n_2-1]$$
$$g_{\pi/2}[n_1,n_2] \leftarrow g_{\pi/4}[n_1,n_2] + g_{\pi/2}[n_1,n_2-1]$$
$$g_{\mathrm{b}}[n_1,n_2] \leftarrow \sqrt{2}g_{\pi/2}[n_1,n_2] + g_{\mathrm{b}}[n_1+1,n_2-1]$$

3. Local FD operation:
for each position $\boldsymbol{n}$ **do**
  compute $w_i, \boldsymbol{x}_i$ and τ using $\mathbf{a}(\boldsymbol{n})$
  evaluate $F(\boldsymbol{n}+\tau-\boldsymbol{x}_i)$ using ZP interpolation
  $\bar{f}[\boldsymbol{n}] \leftarrow \sum_i w_i F(\boldsymbol{n}+\tau-\boldsymbol{x}_i)$
end for
4. Return $\bar{f}[\boldsymbol{n}]$

where $\nu_\phi = \frac{1}{2}(\tan\phi - \cot\phi)\mathrm{sign}\left(\frac{\pi}{2}-\phi\right)$. *The supremum is over the set of* $\mathbf{a}$ *for which* $\theta_{\mathbf{a}} = \phi$.

Fig. 6 illustrates the variation of $1/U(\phi)$ as a function of the orientation; the rationale behind showing the inverse plot is to avoid the blowups $U(\phi) \longrightarrow +\infty$ as $\phi \longrightarrow \theta_k$. In particular, a bound of $3+2\sqrt{2} \approx 5.8$ is attained along the orientations $\phi = (2k-1)\pi/8, 1 \le k \le 4$, exactly mid-way between two adjacent primal directions. This is perfectly reasonable since the control on the geometry of the box spline is minimal along these directions.

In order to specify the elliptical geometry of the box spline, we use either of the following equivalent descriptors as per convenience:

(D1) Size, elongation and orientation (s, ϱ, θ).

(D2) Length of the major and minor axes, and the orientation $(\sqrt{\lambda}_{\max}, \sqrt{\lambda}_{\min}, \theta)$.

(D3) Covariance matrix $\mathbf{C}$.

Descriptor (D1) stipulates the lengths of the major and minor axes as $\lambda_{\max} = s\varrho/(1+\varrho)$ and $\lambda_{\min} = s/(1+\varrho)$, respectively, whereas, (D2) gives the corresponding covariance as

$$\mathbf{C} = \begin{pmatrix} \lambda_{\max}\cos^2\theta + \lambda_{\min}\sin^2\theta & \frac{1}{2}(\lambda_{\max}-\lambda_{\min})\sin 2\theta \\ \frac{1}{2}(\lambda_{\max}-\lambda_{\min})\sin 2\theta & \lambda_{\min}\cos^2\theta + \lambda_{\max}\sin^2\theta \end{pmatrix}.$$

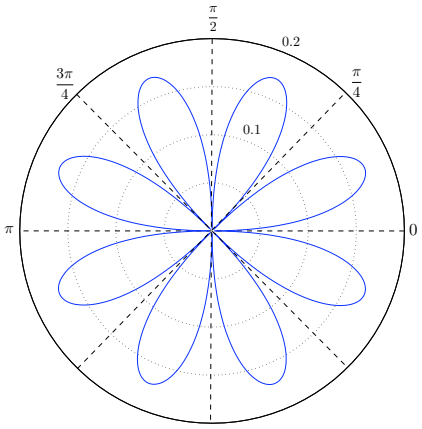


Figure 6: Polar plot of the symmetric variation of $1/U(\phi)$ as a function of the filter orientation ϕ, where $U(\phi)$ is the bound on the elongation. The bound reaches its minimum when the orientation of the filter is exactly midway between two principal axes, whereas arbitrary elongation is achievable in the neighborhood of the four principal directions $\phi = 0, \pi/4, \pi/2$ and $3\pi/4$.

3.2.2 Optimal scale-vector for a given anisotropy

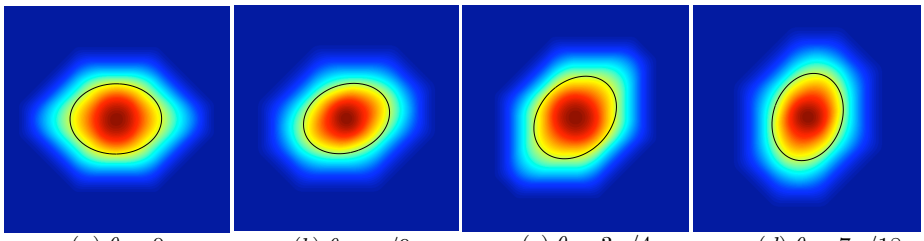
Since the covariance matrix of the four-directional box spline (cf. Eqn. (22)) is given by

$$\mathbf{C}_{\mathbf{a}} = \frac{1}{24}\begin{pmatrix} 2a_1^2 + a_2^2 + a_4^2 & a_2^2 - a_4^2 \\ a_2^2 - a_4^2 & 2a_3^2 + a_2^2 + a_4^2 \end{pmatrix}, \tag{26}$$

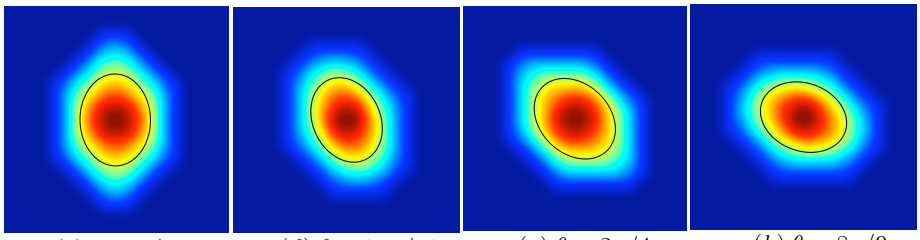
the inverse problem is that of specifying a scale-vector $\mathbf{a}$ such that $\mathbf{C}_{\mathbf{a}} = \mathbf{C}$. By introducing the positive vector $\mathbf{p} = (a_1^2, a_2^2, a_3^2, a_4^2)$, the problem can be reformulated as: find $\mathbf{p} > 0$, such that $\mathbf{M}\mathbf{p} = \mathbf{c}$, where

$$\mathbf{M} = \begin{pmatrix} 2 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 1 & 2 & 1 \end{pmatrix}, \quad \text{and} \quad \mathbf{c} = 24\Big(\mathbf{C}(1,1),\ \mathbf{C}(1,2),\ \mathbf{C}(2,2)\Big).$$

The scale-vector solution is then given by $\mathbf{a} = \sqrt{\mathbf{p}}$. As far as existence of solutions is concerned, proposition 3.1 ensures that the linear system $\mathbf{M}\mathbf{p} = \mathbf{c}, \mathbf{p} > 0$, corresponding to a given geometry $(\lambda_{\min}, \lambda_{\max}, \theta)$, is always solvable provided that the elongation $\varrho < U(\theta)$. Moreover, as it turns out, the linear system is under-determined and has infinitely many solutions. The idea then would be to use a scale-vector that is "optimal" in some sense. But first, we try to characterize the solution space of the system $\mathbf{M}\mathbf{p} = \mathbf{c}, \mathbf{p} \geq \varepsilon\mathbf{1}$. For reasons that will soon be obvious, we propose to modify the positivity constraint as $\mathbf{p} \geq \varepsilon\mathbf{1}$, where ε is some arbitrarily small positive number. We observe that $\mathbf{M}$ is full-rank, and hence the null-space is of dimension $4 - 3 = 1$.



(a) $\theta = 0$ (b) $\theta = \pi/9$ (c) $\theta = 3\pi/4$ (d) $\theta = 7\pi/18$



(e) $\theta = \pi/2$ (f) $\theta = 23\pi/36$ (g) $\theta = 3\pi/4$ (h) $\theta = 8\pi/9$

Figure 7: Intensity distributions of the four-directional box splines of identical size ($s = 1$) and elongation ($\varrho = 2.5$), but with different orientations. The ellipse in each figure represents a level-set of the Gaussian having the same covariance as the corresponding box spline.

In particular, this signifies that the solutions of $\mathbf{Mp} = \mathbf{c}$ lie on the affine subspace $\{\bar{\mathbf{p}} + t\mathbf{e} : t \in \mathbf{R}\}$, where $\bar{\mathbf{p}}$ is a particular solution ($\mathbf{M}\bar{\mathbf{p}} = \mathbf{c}$) and $\mathbf{e}$ is in the null-space ($\mathbf{Me} = \mathsf{0}$). Moreover, one can easily verify that in order to adhere to the positivity constraint, $\bar{\mathbf{p}} + t\mathbf{e} = (\bar{p}_1 + t, \bar{p}_2 - t, \bar{p}_3 + t, \bar{p}_4 - t) \geq \varepsilon\mathbf{1}$, it is both necessary and sufficient that t lies in the closed interval $[t_\ell, t_r]$, where $t_\ell = \max(-\bar{p}_1 + \varepsilon, -\bar{p}_3 + \varepsilon)$ and $t_r = \min(\bar{p}_2 - \varepsilon, \bar{p}_4 - \varepsilon)$. In particular, we set $\mathbf{e} = (1, -1, 1, -1)$ which is in the null-space of $\mathbf{M}$. Note that one can easily compute $\bar{\mathbf{p}}$ by pivoting one of its components and solving for the remaining three; since $\mathbf{M}$ is of full-rank, the reduced system is always solvable. We now use the available degree of freedom to select a solution that maximizes a certain measure of Gaussianity.

A classical measure of the Gaussianity of a 1-D function is its kurtosis (the fourth-order cumulant). For a centered function $f(x)$, this is defined as $\varkappa = \mu_4 - 3\mu_2^2$, where μ_4 and μ_2 are the fourth-order and second-order moments of $f(x)$, respectively. The central property of this measure is that $\varkappa = 0$ for a true Gaussian function, and as a result, the absolute value of the kurtosis provides a measure of Gaussianity of the function. In particular, smaller absolute values correspond to more Gaussian-like functions.

As for a bivariate function $f(\boldsymbol{x})$, we shall use the following matrix-valued extension

$$\mathbf{K} = \mathbf{L} - \mathrm{tr}(\mathbf{C})\mathbf{C} - 2\mathbf{C}^2, \tag{27}$$

where $\mathbf{C} = \int (\boldsymbol{x}\boldsymbol{x}^T) f(\boldsymbol{x}) d\boldsymbol{x}$ and $\mathbf{L} = \int (\boldsymbol{x}\boldsymbol{x}^T)^2 f(\boldsymbol{x}) d\boldsymbol{x}$ are the second-order and fourth-order moment matrices of $f(\boldsymbol{x})$, respectively [17]. This constitutes a valid extension

of the 1-D kurtosis since (27) reduces to $\varkappa = \mu_4 - 3\mu_2^2$ when $d = 1$. Moreover, we also have the following desirable properties:

(i) If $f(\boldsymbol{x})$ is a multivariate Gaussian, then $\mathbf{K} = \mathbf{0}$ (cf. [17] for a proof).
(ii) The Frobenius norm of $\mathbf{K}$, namely,

$$||\mathbf{K}|| = \Big(\sum_{i,j} |\mathbf{K}(i,j)|^2 \Big)^{1/2}$$

is rotation-invariant, i.e., the kurtosis matrices of the rotations of $f(\boldsymbol{x})$ have the same Frobenius norm (proof in §6.4).

Following the above arguments, we propose to solve the optimization problem

$$\mathbf{p}_0 = \text{argmin}_{\mathbf{p}} \, ||\mathbf{K}_{\mathbf{p}}||^2, \qquad \mathbf{M}\mathbf{p} = \mathbf{c}, \; \mathbf{p} \geq \varepsilon \mathbf{1}. \tag{28}$$

This yields the optimal scale-vector $\mathbf{a}_0 = \sqrt{\mathbf{p}_0}$ corresponding to the most Gaussian-like box spline. The rotation-invariance property ensures that the box splines of identical size and elongation but different orientation, obtained via the solutions of the above optimization problem, are as homogenous as possible.

The norm of the kurtosis matrix of $\beta_{\mathbf{a}}(\boldsymbol{x})$ turns out to be $||\mathbf{K}_{\mathbf{p}}||^2 = \sum_k p_k^4 + (p_1^2 + p_3^2)(p_2^2 + p_4^2)$ (see Appendix §6.5). Substituting $p_k = \bar{p}_k + t\, e_k$ into this expression, we arrive at the quartic polynomial

$$\zeta(t) = \sum_k (\bar{p}_k + e_k t)^4 + \Big\{ (\bar{p}_1 + t)^2 + (\bar{p}_3 + t)^2 \Big\} \Big\{ (\bar{p}_2 - t)^2 + (\bar{p}_4 - t)^2 \Big\},$$

which, together with the parameterization $\mathbf{p} = \bar{\mathbf{p}} + t\,\mathbf{e}$, simplifies the problem to one of finding

$$t_0 = \arg\min_t \, \zeta(t), \quad t \in [t_\ell, t_r]. \tag{29}$$

The optimal solution is then given by $\mathbf{a}_0 = \sqrt{\bar{\mathbf{p}} + t_0 \mathbf{e}}$. This problem however is easily solved, since the minimum is attained either at one of the interior points (t_ℓ, t_r) where $\zeta'(t) = 0$, or at one of the boundary points. In particular, we have the following simple algorithm for designing optimized Gaussian-like box splines of a specified covariance:

(i) Set $p_4 = 1$, and compute $\bar{\mathbf{p}}$ by solving the sytem $\mathbf{M}\bar{\mathbf{p}} = \mathbf{c}$.
(ii) Use $\bar{\mathbf{p}}$ to compute t_ℓ, t_r and the coefficients of $\zeta'(t)$.
(iii) Find the real roots of $\zeta'(t) = 0$ over the interval (t_ℓ, t_r); denote the set of real roots by R. Then $\mathbf{a}_0 = (\bar{\mathbf{p}} + t_0 \mathbf{e})^{1/2}$, where[3] $t_0 = \text{argmin}_t \, \zeta(t), t \in R \cup \{t_\ell, t_r\}$.

In particular, the coefficients of the cubic equation $\zeta'(t) = \zeta_1 t^3 + \zeta_2 t^2 + \zeta_3 t + \zeta_4 = 0$ in (iii) are given by $\zeta_1 = 32, \zeta_2 = 24(\bar{p}_1 - \bar{p}_2 + \bar{p}_3 - \bar{p}_4), \zeta_3 = 16 \sum \bar{p}_k^2 - 8(\bar{p}_1 + \bar{p}_3)(\bar{p}_2 + \bar{p}_4), \zeta_4 = 4(\bar{p}_1^3 - \bar{p}_2^3 + \bar{p}_3^3 - \bar{p}_4^3) + 2(\bar{p}_1 + \bar{p}_3)(\bar{p}_2^2 + \bar{p}_4^2) - 2(\bar{p}_2 + \bar{p}_4)(\bar{p}_1^2 + \bar{p}_3^2)$.

[3]The tie is randomly broken if $\zeta(t)$ has multiple minimizers over $[t_\ell, t_r]$ (this was rarely reported in practice).

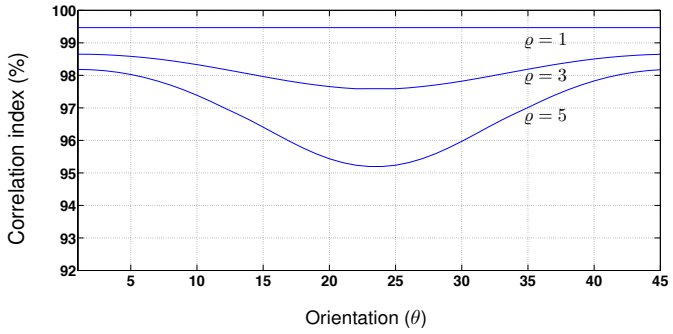


Figure 8: Normalized correlation between the optimal four-directional box spline and the target Gaussian at different elongations and orientations. For a fixed elongation, the correlation is minimum at the critical orientation $\theta = 22.5°$, and improves symmetrically as θ approaches the principal orientations (cf. Fig. 6).

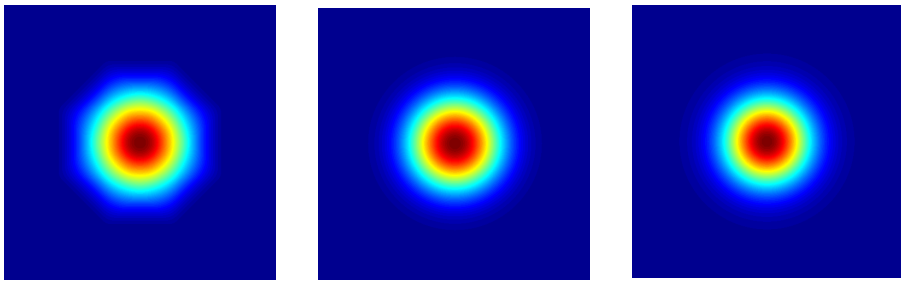
The box splines obtained using the above optimization at various orientations are shown in Fig. 7. The quality of the Gaussian approximation under different practical settings of the orientation and the elongation is quantified in Fig. 8.

The correspondences $(1, \varrho, \theta) \leftrightarrow (a_1, a_2, a_3, a_4)$ for $0 < \theta < \pi$ and $1 \leq \varrho < U(\theta)$ can be pre-computed and stored in a look-up table. Note that for a given ϱ, the set of correspondences $(1, \varrho, \theta) \leftrightarrow (a_1, a_2, a_3, a_4)$ have an inherent four-fold symmetry in θ owing to the presence of the four principal directions. Hence, it suffices to store the scale-vector correspondences for $0 < \theta < \pi/4$ which reduces the size of the LUT by a factor of four. Indeed, for any arbitrary size $s > 1$, orientations $0 < \theta < \pi$, and elongation $1 \leq \varrho < U(\theta)$, the corresponding scale-vector is then obtained through the following operations:
(O1) Rotation:

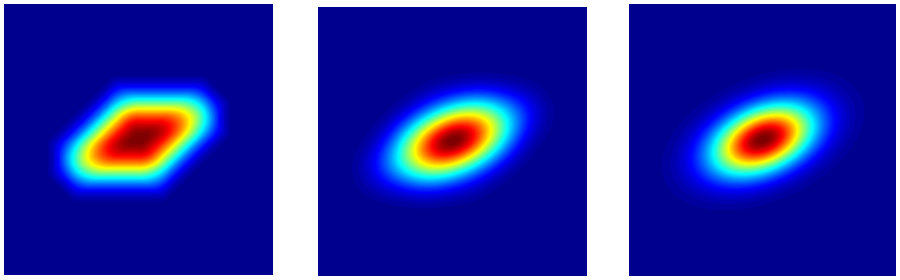
$$\phi = \begin{cases} \theta & \text{for} \quad 0 < \theta < \pi/4 \\ \theta - \pi/4 & \text{for} \quad \pi/4 < \theta < \pi/2 \\ \theta - \pi/2 & \text{for} \quad \pi/2 < \theta < 3\pi/4 \\ \theta - 3\pi/4 & \text{for} \quad 3\pi/4 < \theta < \pi. \end{cases}$$

(O2) Find (a_1, a_2, a_3, a_4) corresponding to $(1, \varrho, \phi)$ using the LUT. The desired scale-vector is then given by the following permutation and rescaling:

$$(a_1, a_2, a_3, a_4) \mapsto \begin{cases} \sqrt{s}(a_1, a_2, a_3, a_4) & \text{for} \quad 0 < \theta < \pi/4 \\ \sqrt{s}(a_2, a_3, a_4, a_1) & \text{for} \quad \pi/4 < \theta < \pi/2 \\ \sqrt{s}(a_3, a_4, a_1, a_2) & \text{for} \quad \pi/2 < \theta < 3\pi/4 \\ \sqrt{s}(a_4, a_1, a_2, a_3) & \text{for} \quad 3\pi/4 < \theta < \pi. \end{cases}$$



(a) Isotropic forms ($s = 1, \varrho = 1$)



(b) Anisotropic forms ($s = 1, \varrho = 3, \theta = \pi/8$)

Figure 9: Higher-order box splines through iterative convolutions. **Left**: The reference four-directional box spline; **Center**: Iterated box spline obtained by convolving the (rescaled) four-directional box spline with itself; **Right**: Target Gaussian having identical covariance.

3.3 Higher-order box splines

As suggested by the convergence result (18), the Gaussian-like nature of the four-directional box splines can be improved by using more directions. Implementing the corresponding space-variant filtering using the algorithm in §2.2.2 however turns out to be challenging and not very practical—the principal axes of these box splines are generally along off-grid directions, and one needs to interpolate the image for implementing the associated running-sums.

The iterated four-directional box splines $\beta_{\mathbf{a}}^{4,m}(\boldsymbol{x})$ introduced in §2.2.3 provide a practical alternative. These box splines rapidly converge to a Gaussian with the increase in m. Also, note that the four-directional box spline and its iterates have identical covariances. This implies that the algorithm in §3.2.2 can be used for optimizing the iterated box splines as well. The first two iterates of the four-directional box spline along with the target Gaussian are shown in Fig. 9. It is seen that $\beta_{\mathbf{a}}^{4,2}(\boldsymbol{x})$ resembles the target Gaussian very closely. In fact, the minimum correlation coefficient rises from 95% to 99% for $m = 2$ (cf. Fig. 8). In practice, we can thus implement a higher-order Gaussian-like filtering by simple iterations of the algorithm in §3.1. It suffices to set

the scale-vector in the algorithm as $\mathbf{a}/\sqrt{m}$, where m is the number of iterations.

4 EXPERIMENTAL RESULTS

4.1 Computation time

The space-variant filtering using the four-directional box spline was implemented in Java on a 2.66 GHz Intel system. The typical execution times required for convolving a 512×512 image with kernels of various sizes are shown in Table 2. It is clear that the run time is independent of the size of the kernel.

Table 2: Average computation time for box splines of different sizes.

Size (s)	1	2	4	8	16
Time (millisec.)	101	100	103	101	100

4.2 Application: Feature-preserving smoothing

We now present an application to demonstrate the space-variant algorithm described in §3.1. Filtering of noisy images using isotropic Gaussian filters often results in excessive blurring of the anisotropic image features. Diffusion filters are known to perform better in such cases [19]. As an alternative, we propose to filter the corrupted image using our anisotropic Gaussian-like filters, where we adapt the size, elongation and orientation of the filter to the local image features. The main idea is to locally average the image using elliptical windows that have been elongated along the image feature (orthogonal to the local gradient). This induces more smoothing along the direction of minimal intensity variation resulting in the suppression of the ambient noise, while preserving the sharpness of the image features.

To derive an estimate of the local image anisotropy, we use the paradigm of *structure tensors* [9], where the local orientation $\theta(\boldsymbol{x})$ is estimated through the minimization of a certain weighted norm of the directional derivative. In particular, if we denote the directional derivative of $f(\boldsymbol{x})$ along along $\boldsymbol{u}_\theta = (\cos\theta, \sin\theta)$ by $D_\theta f(\boldsymbol{x})$, then $\theta(\boldsymbol{x})$ is given by the minimizer of

$$\int_\Omega w(\boldsymbol{s}) |(D_\theta f)(\boldsymbol{x} - \boldsymbol{s})|^2 \, d\boldsymbol{s}, \tag{30}$$

where Ω is the support of the isotropic averaging window $w(\boldsymbol{x})$. The solubility of the above optimization problem follows from the observation that this can be recast as an eigenvalue problem. In particular, by expressing the directional derivative in terms of the gradient $\mathbf{g}(\boldsymbol{x})$, namely as $D_\theta f(\boldsymbol{x}) = \boldsymbol{u}_\theta^T \mathbf{g}(\boldsymbol{x})$, one can rewrite (30) as

$$\boldsymbol{u}_\theta^T \mathbf{J}(\boldsymbol{x}) \boldsymbol{u}_\theta, \tag{31}$$

where the structure tensor $\mathbf{J}(\boldsymbol{x})$ is the 2×2 positive-definite matrix

$$\int_\Omega w(\boldsymbol{s}) (\mathbf{g}\mathbf{g}^T)(\boldsymbol{x} - \boldsymbol{s}) d\boldsymbol{s}.$$

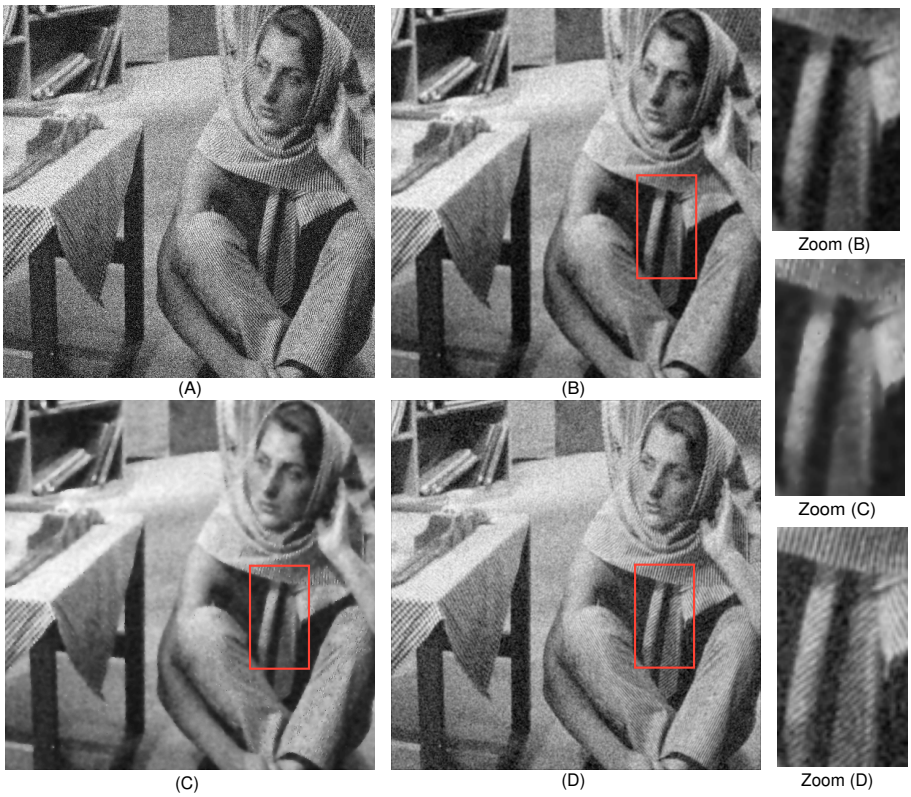


Figure 10: Results on a test image. (**A**) *Barbara* corrupted with additive Gaussian noise, PSNR = 18.0 dB; (**B**) Isotropic smoothing, PSNR = 23.10 dB; (**C**) Diffusion filtering, PSNR = 23.25 dB; (**D**) Our algorithm, PSNR = 23.58 dB.

The local orientation $\theta^\star(\boldsymbol{x})$ is then given by the minimizer of (31) associated with the minimum eigenvalue of $\mathbf{J}(\boldsymbol{x})$.

In view of the definitions in (23) and the fact that the eigenvalues of $\mathbf{J}(\boldsymbol{x})$ are always non-negative, we propose to estimate the elongation as follows: We set $\varrho^\star = \lambda_{\max}/\lambda_{\min}$ if both eigenvalues are non-zero, equal to 1 if both are zero (locally isotropic intensity), and equal to $\max(1, \lambda)$ if only one of the eigenvalues λ is non-zero. Finally, we estimate the size of the box spline as $s^\star = \lambda_{\max} + \lambda_{\min}$. The triple $(s^\star, \varrho^\star, \theta^\star)$ is then used to compute the optimal scale-vector using the algorithm described in §3.2. The components of $\mathbf{J}$ can be efficiently computed using simple convolution and pointwise operations; we refer the reader to [9, Chapter 13] for implementation details. The main steps of the proposed smoothing algorithm are:

- Computation of the structure-tensor.
- Pre-integration of the corrupted image using the running-sum filters.

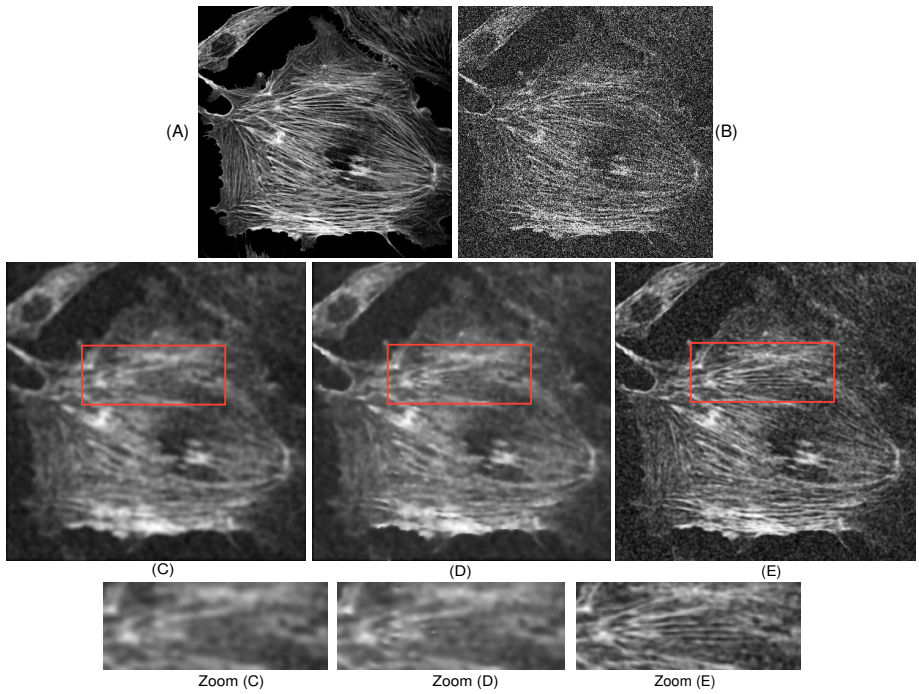


Figure 11: Results on a real image. (**A**) Noise-free immunofluorescence image of actin fibres (Courtesy of C. Aemisegger, CMIA, University of Zürich); (**B**) Image corrupted with additive Gaussian noise, PSNR = 12.20 dB; (**C**) Isotropic smoothing, PSNR = 15.38 dB; (**D**) Diffusion filtering, PSNR = 15.50 dB; (**E**) Our algorithm, PSNR = 15.80 dB.

- Computation of the triple $(s^\star, \varrho^\star, \theta^\star)$ at every feature location using the structure tensor. This is used to compute the scale-vector $\mathbf{a}(\boldsymbol{n})$ of the optimal Gaussian-like box spline using the algorithm in §3.2. Isotropic box splines are used in the uniform-intensity regions; we set $\mathbf{a}(\boldsymbol{n}) = (\sigma, \sigma, \sigma, \sigma)$, where σ is proportional to the noise variance.

- Computation of the FD mesh using $\mathbf{a}(\boldsymbol{n})$, and its application to the pre-integrated image.

To demonstrate the effectiveness of our strategy in preserving oriented patterns in noisy images, we compare the results obtained from our algorithm with those obtained using the (fixed-scale) isotropic Gaussian filter and the Perona-Malik diffusion filter [14]. We use the standard test image of *Barbara* and corrupt it with additive Gaussian noise. The variance of the noise is used to set the size of the Gaussian for the isotropic smoothing. The parameters used for the Perona-Malik filter were typical: time step of 0.1, conductance in the range of $10 \sim 30$, and a total of $15 \sim 30$ iterations. The parameters were manually tuned to optimize the PSNR, and also to avoid blocking artifacts. Fig. 10 shows the results obtained from the different filters. As far as the quantitative evaluation of the filters is concerned, our algorithm clearly outperforms both isotropic and diffusion filters in terms of the Peak-Signal-to-Noise-Ratio (PSNR). Moreover, as shown in the zoomed-in sections of the respective images, the oriented stripes on the clothes are quite faithfully restored by our algorithm. A significant amount of blurring of the stripes is seen in the results obtained using isotropic and diffusion filtering. The non-linear diffusion filter, however, tends to perform better at low PSNRs in the range of 5-10 dB (cf. Table 3).

Next, we compare the results on a real biological image and at a much lower PSNR of around 12 dB. We consider the fluorescence image shown in Fig. 11, which exhibits numerous elongated fiber-like structures. The parameters of the isotropic filter and the diffusion filter are set as in the previous case, except that the iteration count for the latter is increased to 15. As before, the improvement of the PSNR obtained using our filter is higher. Importantly, as seen from the zooms, our algorithm results in significantly less merging of the close fibers and blurring of the finer ones. The average execution time of our algorithm is 0.6 seconds for a 512×512 image, which includes the computation of the structure-tensor, the running-sums, the optimal scale-vector, the interpolated samples and the finite-differences.

The four-directional box splines can also be used to derive fast space-variant detectors based on Gaussian forms, e.g., the Laplacian-of-the-Gaussian (LoG) or the so-called Mexican-hat detector. We refer the interested reader to [3], where the isotropic forms of the four-directional box spline were used to realize a fast and scalable Mexican-hat-like detector. In particular, a modified version of the space-variant algorithm described in §3.1 is used to design an efficient coarse-to-fine strategy for the detection of centers and radii of cells/nuclei in fluorescence images.

Table 3: Comparison of the filters at different noise levels using the test image of *Barbara*. The table shows the PSNR of the outputs.

Input PSNR (dB)	10.0	12.0	14.0	16.0	18.0	20.0
Isotropic filter	15.38	18.20	20.20	21.65	23.10	24.30
Diffusion filter	**15.48**	18.31	20.30	21.70	23.25	24.35
Our filter	15.45	**18.38**	**20.57**	**21.94**	**23.58**	**24.56**

5 CONCLUSION

In this paper, we presented a framework for elliptical filtering using the radially-uniform box splines. The associated space-variant filtering was efficiently realized using running-sums and local finite-differences. The attractive features of our algorithm are:

- the $O(1)$ computational complexity per pixel, and
- the use of real-valued parameters for continuously controlling the shape and size of the filter.

Our filtering paradigm offers a nice trade-off between the quality of approximation of Gaussians and the computational complexity of linear space-variant filtering.

We also presented a closed form solution for the problem of constructing four-directional box splines with given covariances. The scope of our algorithm was demonstrated through the realization of a smoothing filter that can adapt to the local image characteristics.

6 APPENDIX

6.1 Proof of theorem 2.2

We first establish that the Fourier sequence $\widehat{\beta}^2_{\mathbf{a}(2)}(\boldsymbol{\omega}), \widehat{\beta}^3_{\mathbf{a}(3)}(\boldsymbol{\omega}), \ldots$ convergences pointwise to a Gaussian:

$$\lim_{N \longrightarrow \infty} \widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega}) = \exp\Big(-\frac{\sigma^2}{2} \|\boldsymbol{\omega}\|^2 \Big). \tag{32}$$

We then show that the above convergence is also in the $\mathbf{L}^2(\mathbf{R}^2)$ norm. This will establish the theorem, since it is well-known that the Fourier transform of a Gaussian is a Gaussian, and that $f_n \longrightarrow g$ in $\mathbf{L}^2$ if $\hat{f}_n \longrightarrow \hat{g}$ in $\mathbf{L}^2$.

To derive (32), we note that $\widehat{\varphi}_{a,\theta}(\boldsymbol{\omega}) = \widehat{\beta}_a(\boldsymbol{u}_\theta^T \boldsymbol{\omega}) = \mathrm{sinc}\left(a \boldsymbol{u}_\theta^T \boldsymbol{\omega}/2\right)$, where $\mathrm{sinc}(x) = \sin(x)/x$ for $x \neq 0$, and equals 1 at the origin. Then, the convolution-multiplication rule gives

$$\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega}) = \prod_{k=1}^{N} \widehat{\varphi}_{a_k(N),\theta_k}(\boldsymbol{\omega}) = \prod_{k=1}^{N} \mathrm{sinc}\left(\frac{a_k(N)}{2} \boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega} \right). \tag{33}$$

Using the estimate $\mathrm{sinc}(x) = 1 - x^2/6 + O(x^4)$ for $|x| < 1$, and substituting $a_k(N) = \sigma\sqrt{(24/N)}$ into (33), we have

$$\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega}) = \prod_{k=1}^{N} \left(1 - \frac{\sigma^2}{N}(\boldsymbol{u}^T_{\theta_k}\boldsymbol{\omega})^2 + O\left(N^{-2}\right)\right) \qquad (\|\boldsymbol{\omega}\| < cN) \tag{34}$$

where c is some positive constant. By developing the quadratic factor $(\boldsymbol{u}^T_{\theta_k}\boldsymbol{\omega})^2$ and the product in (34), we arrive at the estimate

$$\begin{aligned}
\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega}) &= \prod_{k=1}^{N} \left\{1 - \frac{\sigma^2}{2N}\|\boldsymbol{\omega}\|^2 + \frac{\sigma^2}{2N}(\omega_1^2 - \omega_2^2)\cos 2\theta_k + \frac{\sigma^2}{N}\omega_1\omega_2 \sin 2\theta_k + O\left(N^{-2}\right)\right\} \\
&= \left(1 - \frac{\sigma^2}{2N}\|\boldsymbol{\omega}\|^2\right)^N + \frac{\sigma^2}{2N}(\omega_1^2 - \omega_2^2)\left(1 - \frac{\sigma^2}{2N}\|\boldsymbol{\omega}\|^2\right)^{N-1} \sum_{k=1}^{N} \cos 2\theta_k \\
&\quad + \frac{\sigma^2}{N}\omega_1\omega_2\left(1 - \frac{\sigma^2}{2N}\|\boldsymbol{\omega}\|^2\right)^{N-1} \sum_{k=1}^{N} \sin 2\theta_k + O\left(N^{-2}\right) \\
&= \left(1 - \frac{\sigma^2}{2N}\|\boldsymbol{\omega}\|^2\right)^N + O\left(N^{-2}\right) \qquad (\|\boldsymbol{\omega}\| < cN).
\end{aligned} \tag{35}$$

This is exactly where the fact that θ_k are uniformly distributed over $[0, \pi)$ is invoked: the cancellation of the linear factors in the second step is based on the identities $\sum_{k=1}^{N} \cos 2\theta_k = 0$, and $\sum_{k=1}^{N} \sin 2\theta_k = 0$, where $\theta_k = (k-1)\pi/N$. Since $(1 - x/m)^m$ converges to $\exp(-x)$ as $m \longrightarrow \infty$, it can now be readily seen that (32) follows as the limiting case of (35) .

To demonstrate that (32) holds in the $\mathbf{L}^2$ norm sense, it suffices to show the sequence of error functions $\mathscr{E}_N(\boldsymbol{\omega}) = \widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega}) - \exp(-\sigma^2\|\boldsymbol{\omega}\|^2/2)$ converge to zero in the above norm, i.e., $\|\mathscr{E}_N\|_{\mathbf{L}^2} \longrightarrow 0$ as $N \longrightarrow \infty$. Since we have already demonstrated that $\mathscr{E}_N(\boldsymbol{\omega}) \longrightarrow 0$ pointwise, all we need to show in order to invoke the dominated convergence theorem is that the sequence $|\mathscr{E}_2(\boldsymbol{\omega})|, |\mathscr{E}_3(\boldsymbol{\omega})|, \ldots$ is uniformly bounded by a $\mathbf{L}^2$ function. Moreover, since

$$|\mathscr{E}_N(\boldsymbol{\omega})| \leq |\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega})| + \exp(-\sigma^2\|\boldsymbol{\omega}\|^2/2),$$

it, in fact, suffices to show that each $|\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega})|$ admits such a bound.

The main idea behind establishing such a bound is that the above mentioned sequence can be covered by a Gaussian in a neighborhood of the origin and by a function with sufficient decay at the tails, both of which are independent of N. Indeed,

using the estimate $\mathrm{sinc}(u) \leq 1 - u^2/\pi^2$ for $|u| \leq \pi$, one can verify that

$$\begin{aligned}\left|\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega})\right| &= \prod_{k=1}^{N} \left|\mathrm{sinc}\Big(\frac{\sqrt{6}\sigma}{\sqrt{N}} \boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega}\Big)\right| \\ &\leq \prod_{k=1}^{N} \Big(1 - \frac{6\sigma^2 \left|\boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega}\right|^2}{\pi^2 N}\Big) \\ &\leq \exp\big(-C_1 \|\boldsymbol{\omega}\|^2\big) \qquad (\|\boldsymbol{\omega}\| < \delta)\end{aligned}$$

As far as the tail is concerned, the Cauchy-Schwarz inequality, $|\boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega}| \leq \|\boldsymbol{u}_{\theta_k}\| \cdot \|\boldsymbol{\omega}\| = \|\boldsymbol{\omega}\|$, gives

$$\left|\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega})\right| = \prod_{k=1}^{N} \left|\mathrm{sinc}\Big(\frac{\sqrt{6}\sigma}{\sqrt{N}} \boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega}\Big)\right| \leq \frac{C_2}{\|\boldsymbol{\omega}\|^2} \quad (\|\boldsymbol{\omega}\| \geq \delta).$$

Here C_1, C_2 and δ are appropriate positive constants that are independent of N. Combining the above estimates, we see that

$$\left|\widehat{\beta}^N_{\mathbf{a}(N)}(\boldsymbol{\omega})\right| \leq \exp\big(-C_1 \|\boldsymbol{\omega}\|^2\big) + \frac{C_2}{\|\boldsymbol{\omega}\|^2}\left[1 - \mathrm{rect}\left(\frac{\|\boldsymbol{\omega}\|}{\delta}\right)\right]$$

for all $\boldsymbol{\omega}$. Since the function on the right is indeed in $\mathbf{L}^2(\mathbf{R}^2)$, this establishes the desired bound, and consequently, the norm convergence.

6.2 Covariance matrix

We begin with the observation that if $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are symmetric (about the origin) and have a total mass of unity, then $\mathbf{C}_{f*g} = \mathbf{C}_f + \mathbf{C}_g$, where $\mathbf{C}_f$ denotes the covariance matrix of $f(\boldsymbol{x})$. Indeed, by noting that $\hat{f}(0) = \hat{g}(0) = 1$ (unit mass) and $\partial_i \hat{f}(0) = \partial_i \hat{g}(0) = 0$ (by symmetry), and by recalling the multiplication-differentiation rule $\int x_i x_j f(\boldsymbol{x}) d\boldsymbol{x} = -\partial_i \partial_j \hat{f}(0)$, we have

$$\begin{aligned}\mathbf{C}_{f*g}(i,j) &= \int x_i x_j (f * g)(\boldsymbol{x}) d\boldsymbol{x} \\ &= -\hat{g}(0)\partial_i\partial_j \hat{f}(0) - \hat{f}(0)\partial_i\partial_j \hat{g}(0) - \partial_i \hat{f}(0)\partial_j \hat{g}(0) - \partial_i \hat{g}(0)\partial_j \hat{f}(0) \\ &= -\partial_i\partial_j \hat{f}(0) - \partial_i\partial_j \hat{g}(0) \\ &= \mathbf{C}_f(i,j) + \mathbf{C}_g(i,j).\end{aligned}$$

Since the directional box distributions $\varphi_{a_k,\theta_k}(\boldsymbol{x})$ satisfy these criteria, we have that $\mathbf{C}^N_{\mathbf{a}} = \sum_k \mathbf{C}_k$, where $\mathbf{C}_k$ is the covariance matrix of $\varphi_{a_k,\theta_k}(\boldsymbol{x})$. We explicitly compute the component $\mathbf{C}(1,2)$; the remaining components can be similarly derived. Using

the multiplication-differentiation rule again, we have

$$\begin{aligned}\mathbf{C}_k(1,2) &= \int x_1 x_2 \varphi_{a_k,\theta_k}(\boldsymbol{x}) d\boldsymbol{x} \\ &= -\partial_1\partial_2 \hat{\beta}_{a_k}(\boldsymbol{u}_{\theta_k}^T \boldsymbol{\omega})\Big|_{\omega=0} \\ &= \frac{a_k^2}{24}\sin 2\theta_k.\end{aligned}$$

Therefore, $\mathbf{C}_{\mathbf{a}}^N(1,2) = \sum_k \mathbf{C}_k(1,2) = \sum_k a_k^2 \sin 2\theta_k/24$.

As far as the positive-definite nature of $\mathbf{C}_{\mathbf{a}}^N$ is concerned, it will suffice to show that its eigenvalues

$$\lambda_{\max} = \frac{1}{2}\left(\sum a_k^2 + \sqrt{D}\right) \text{ and } \lambda_{\min} = \frac{1}{2}\left(\sum a_k^2 - \sqrt{D}\right)$$

where $D = (\sum a_k^2 \cos(2\theta_k))^2 + (\sum a_k^2 \sin(2\theta_k))^2$, are strictly positive. This is obviously the case for $\lambda_{\max}$. Moreover, the inequality

$$\begin{aligned}\left(\sum_k a_k^2\right)^2 - D &= \left(\sum_k a_k^2\right)^2 - \left(\sum_k a_k^2 \cos(2\theta_k)\right)^2 - \left(\sum_k a_k^2 \sin(2\theta_k)\right)^2 \\ &= 2\sum_{k\neq\ell} a_k^2 a_\ell^2 \Big(1 - \cos(2\theta_k - 2\theta_\ell)\Big) > 0\end{aligned}$$

tells us that $\sum a_k^2 > \sqrt{D}$. Hence, $\lambda_{\min}$ is strictly positive as well.

6.3 Proof of proposition 3.1

Following definition (23), the dependence of orientation of the box spline $\beta_{\mathbf{a}}^4(\boldsymbol{x})$ on the scale-vector can be expressed as

$$\tan\theta_{\mathbf{a}} = \nu + \text{sign}(a_2 - a_4)\sqrt{1+\nu^2} \qquad (0 < \theta < \pi) \tag{36}$$

where $\nu = (a_3^2 - a_1^2)/(a_2^2 - a_4^2)$.

We note the following: it is both necessary and sufficient that $a_2 > a_4$ (resp. $a_2 < a_4$) for the box spline to be oriented between $0 < \theta_{\mathbf{a}} < \pi/2$ (resp. $\pi/2 < \theta_{\mathbf{a}} < \pi$), and it is the map $(a_1, a_2, a_3, a_4) \mapsto (\nu, \text{sign}(a_2 - a_4))$ that uniquely determines the orientation of the box spline. Indeed, the uniqueness aspect is based on the argument that, for $0 < \theta < \pi/2$, (36) reduces to $\tan\theta_{\mathbf{a}} = \nu + \sqrt{1+\nu^2}$ as a consequence of the necessary condition $a_2 > a_4$. This implicitly represents a one-to-one between $\theta_{\mathbf{a}}$ and ν over the domains $(0, \pi/2)$ and $(-\infty, \infty)$, since the map $\theta_{\mathbf{a}} \mapsto \tan\theta_{\mathbf{a}}$ from $(0, \pi/2)$ into $(0, \infty)$, and the map $\nu \mapsto \nu + \sqrt{1+\nu^2}$ from $(-\infty, \infty)$ into $(0, \infty)$ are both strictly monotonic. In a similar vein, a one-to-one between $\theta_{\mathbf{a}}$ and ν over the domains $(\pi/2, \pi)$ and $(-\infty, \infty)$ can be established. In particular, we have

$$\nu = \frac{1}{2}(\tan\theta_{\mathbf{a}} - \cot\theta_{\mathbf{a}})\text{sign}\left(\frac{\pi}{2} - \theta_{\mathbf{a}}\right). \tag{37}$$

That is, given any orientation $\theta_{\mathbf{a}} = \phi$, the corresponding ν_ϕ is uniquely specified by (37). This establishes the first part of the proposition, since there trivially exists some positive vector $(a_1, \ldots, a_4)$ such that $(a_3^2 - a_1^2)/(a_2^2 - a_4^2) = \nu_\phi$.

As far as the bound is concerned, we observe that the elongation can be expressed as $\varrho_{\mathbf{a}} = 1 + 2/(\gamma - 1)$, where $\gamma = \sum a_k^2 / \sqrt{D_{\mathbf{a}}} \geq 1$. For a given orientation $\theta_{\mathbf{a}} = \phi$, the components of the feasible scale-vectors bear the relation $(a_1^2 - a_3^2) = \nu_\phi (a_4^2 - a_2^2)$, and thus we have that

$$\begin{aligned}
\gamma &= \frac{\sum a_k^2}{\sqrt{(a_3^2 - a_1^2)^2 + (a_2^2 - a_4^2)^2}} \\
&= \frac{a_1^2 + a_3^2}{\sqrt{(a_3^2 - a_1^2)^2 + (a_2^2 - a_4^2)^2}} + \frac{a_2^2 + a_4^2}{\sqrt{(a_3^2 - a_1^2)^2 + (a_2^2 - a_4^2)^2}} \\
&= \frac{1}{\sqrt{1 + \nu_\phi^2}} \frac{a_1^2 + a_3^2}{|a_1^2 - a_3^2|} + \frac{|\nu_\phi|}{\sqrt{1 + \nu_\phi^2}} \frac{a_2^2 + a_4^2}{|a_2^2 - a_4^2|} > \frac{1 + |\nu_\phi|}{\sqrt{1 + \nu_\phi^2}}
\end{aligned}$$

following the trivial inequalities $a_1^2 + a_3^2 > |a_1^2 - a_3^2|$, and $a_2^2 + a_4^2 > |a_2^2 - a_4^2|$. Consequently,

$$\varrho_{\mathbf{a}} = 1 + \frac{2}{\gamma - 1} < \frac{1 + |\nu_\phi| + \sqrt{1 + \nu_\phi^2}}{1 + |\nu_\phi| - \sqrt{1 + \nu_\phi^2}}. \tag{38}$$

The above bound is tight since it can be approached arbitrary closely by making the scales a_ℓ and a_k $(\theta_\ell < \phi < \theta_k)$ arbitrarily large.

6.4 Rotation-invariance

Let $\mathbf{K}$ and $\mathbf{K}_\theta$ denote the kurtosis matrices of $f(\boldsymbol{x})$ and its rotation $f(\mathbf{R}_\theta^T \boldsymbol{x})$, respectively, where $\mathbf{R}_\theta$ is the rotation matrix. Observe that the matrices $\mathbf{L}_\theta$ and $\mathbf{L}$ are related as

$$\begin{aligned}
\mathbf{L}_\theta = \int (\boldsymbol{x}\boldsymbol{x}^T)^2 f(\mathbf{R}_\theta^T \boldsymbol{x}) d\boldsymbol{x} &= \int \mathbf{R}_\theta (\boldsymbol{y}\boldsymbol{y}^T)^2 \mathbf{R}_\theta^T f(\boldsymbol{y}) d\boldsymbol{y} \qquad (\boldsymbol{y} = \mathbf{R}_\theta^T \boldsymbol{x}) \\
&= \mathbf{R}_\theta \left(\int (\boldsymbol{y}\boldsymbol{y}^T)^2 f(\boldsymbol{y}) d\boldsymbol{y} \right) \mathbf{R}_\theta^T \\
&= \mathbf{R}_\theta \mathbf{L} \mathbf{R}_\theta^T.
\end{aligned}$$

Similarly, we have $\mathbf{C}_\theta = \mathbf{R}_\theta \mathbf{C} \mathbf{R}_\theta^T$. This also gives us the equivalence $\mathrm{tr}(\mathbf{C}_\theta) = \mathrm{tr}(\mathbf{R}_\theta \mathbf{C} \mathbf{R}_\theta^T) = \mathrm{tr}(\mathbf{C} \mathbf{R}_\theta^T \mathbf{R}_\theta) = \mathrm{tr}(\mathbf{C})$ following the identities $\mathrm{tr}(\mathbf{AB}) = \mathrm{tr}(\mathbf{BA})$ and $\mathbf{R}_\theta^T \mathbf{R}_\theta = \mathbf{I}$. We can then write

$$\begin{aligned}
\mathbf{K}_\theta &= \mathbf{L}_\theta - \mathrm{tr}(\mathbf{C}_\theta)\mathbf{C}_\theta - 2\mathbf{C}_\theta^2 \mathbf{R}_\theta \mathbf{L} \mathbf{R}_\theta^T - \mathrm{tr}(\mathbf{C}) \mathbf{R}_\theta \mathbf{C} \mathbf{R}_\theta^T - 2\mathbf{R}_\theta \mathbf{C}^2 \mathbf{R}_\theta^T \\
&= \mathbf{R}_\theta (\mathbf{L} - \mathrm{tr}(\mathbf{C})\mathbf{C} - 2\mathbf{C}^2) \mathbf{R}_\theta^T = \mathbf{R}_\theta \mathbf{K} \mathbf{R}_\theta^T.
\end{aligned}$$

Our claim follows immediately, since $||\mathbf{K}_\theta||^2 = \mathrm{tr}(\mathbf{K}_\theta^T \mathbf{K}_\theta) = \mathrm{tr}(\mathbf{R}_\theta \mathbf{K}^T \mathbf{K} \mathbf{R}_\theta^T) = \mathrm{tr}(\mathbf{K}^T \mathbf{K}) = ||\mathbf{K}||^2$.

6.5 Kurtosis measure

In order to compute the kurtosis matrix, we only need to evaluate the fourth-order moments; the second-order moments are already known. In particular, using Fourier identities similar to the ones used in §6.2, one can derive the following expressions:

$$\begin{aligned}
\int x_1^4 \beta_{\mathbf{a}}(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{4}\mu_4 \left(4a_1^4 + a_2^4 + a_4^4\right) + \frac{1}{2}\mu_2^2 \left(6a_1^2 a_2^2 + 6a_1^2 a_4^2 + 3a_2^2 a_4^2\right), \\
\int x_1^3 x_2 \beta_{\mathbf{a}}(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{4}\mu_4 \left(a_2^4 - a_4^4\right) + \frac{3}{2}\mu_2^2 a_1^2 \left(a_2^2 - a_4^2\right), \\
\int x_1^2 x_2^2 \beta_{\mathbf{a}}(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{4}\mu_4 \left(a_2^4 + a_4^4\right) + \frac{1}{2}\mu_2^2 \left(a_1^2 a_2^2 + a_1^2 a_4^2 + a_2^2 a_3^2 + a_3^3 a_4^2 - a_2^2 a_4^2 + 2a_1^2 a_3^2\right), \\
\int x_1 x_2^3 \beta_{\mathbf{a}}(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{4}\mu_4 \left(a_2^4 - a_4^4\right) + \frac{3}{2}\mu_2^2 a_3^2 \left(a_2^2 - a_4^2\right), \\
\int x_2^4 \beta_{\mathbf{a}}(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{4}\mu_4 \left(4a_3^4 + a_2^4 + a_4^4\right) + \frac{1}{2}\mu_2^2 \left(6a_2^2 a_3^2 + 6a_3^2 a_4^2 + 3a_2^2 a_4^2\right),
\end{aligned}$$

where $\mu_4 = 1/80$ and $\mu_2 = 1/12$ denote the fourth and second-order moments of $\beta_1(x)$, respectively. These provide the components of the matrix $\mathbf{L}_{\mathbf{a}}$, which in turn leads to the following simple expression for the kurtosis matrix

$$\mathbf{K}_{\mathbf{a}} = \mathbf{L}_{\mathbf{a}} - \mathrm{tr}(\mathbf{C}_{\mathbf{a}})\mathbf{C}_{\mathbf{a}} - 2\mathbf{C}_{\mathbf{a}}^2 = (\mu_4 - 3\mu_2^2) \begin{pmatrix} a_1^4 + \frac{1}{2}(a_2^4 + a_4^4) & \frac{1}{2}(a_2^4 - a_4^4) \\ \frac{1}{2}(a_2^4 - a_4^4) & a_3^4 + \frac{1}{2}(a_2^4 + a_4^4) \end{pmatrix}. \quad (39)$$

Finally, from (39), we get $||\mathbf{K}_{\mathbf{a}}||^2 = \sum_{k=1}^4 a_k^8 + (a_1^4 + a_3^4)(a_2^4 + a_4^4)$.

We note that the negative factor $(\mu_4 - 3\mu_2^2)$ in (39) is in fact the kurtosis of $\mathrm{rect}(x)$, the sub-Gaussian constituent of the box spline. The fact that $\mathbf{K}_{\mathbf{a}}$ is negative-definite is thus consistent with the sub-Gaussian nature of the resulting box spline.

6.6 Fast ZP interpolation

Given a discrete function $c[\boldsymbol{n}]$ and a point $\boldsymbol{x}$ on $\mathbf{R}^2$, we outline a technique for the fast evaluation of the sum

$$\sum_{\boldsymbol{n} \in \mathbf{Z}^2} \mathbf{c}[\boldsymbol{n}] \beta_{\mathbf{b}}^4(\boldsymbol{n} - \boldsymbol{x})$$

where $\mathbf{b} = (1, \sqrt{2}, 1, \sqrt{2})$. A sketch of the partitions of the piecewise polynomial $\beta_{\mathbf{b}}^4(\boldsymbol{x})$ is provided in Fig. 12. The exact functional forms of the ZP box spline $2\beta_{\mathbf{b}}^4(\boldsymbol{x})$ corresponding to these partitions can be found in [21]. Since $\beta_{\mathbf{b}}^4(\boldsymbol{x})$ has a compact support, this is in fact a finite sum, and requires at most seven evaluations of the function $\beta_{\mathbf{b}}^4(\cdot - \boldsymbol{x})$ for any arbitrary translation $\boldsymbol{x}$. This is illustrated in Fig. 12, where the red dots $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_6$ denote the lattice points that intersect the support of $\beta_{\mathbf{b}}^4(\cdot - \boldsymbol{x})$.

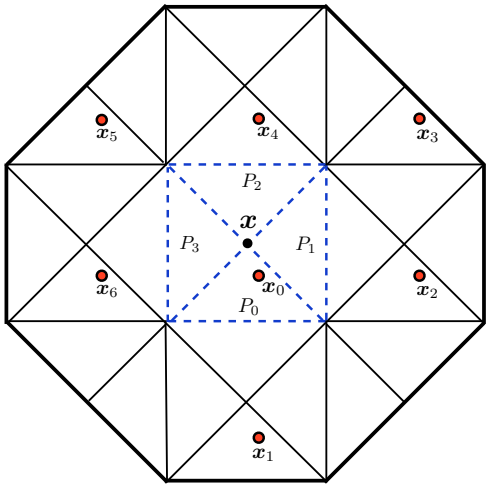


Figure 12: The figure shows the translated ZP box spline $\beta^4_{\mathbf{b}}(\cdot + x), \mathbf{b} = (1, \sqrt{2}, 1, \sqrt{2})$. The red dots $x_1, \ldots, x_7$ correspond to the points on the Cartesian lattice, and the triangular regions $P_1, \ldots, P_4$ are different partitions of the ZP, which together constitute a unit cell of the lattice.

Thus, one needs to evaluate the translated ZP at the points $x_0, \ldots, x_6$ in order to compute the sum. The drawback here is that naive evaluation of the spline at every x_j requires a decision-making to figure out the associated partition before computing the corresponding polynomial.

The redundancy that we exploit is as follows: Consider the triangular regions $P_0, \ldots, P_3$ marked with blue dashed lines in Fig. 12 corresponding to the four different partitions of the ZP. These together constitute a unit cell of the lattice, and hence only one lattice point intersects them. The figure shows a particular instance where this point, denoted by x_0, lies in P_0. This clearly fixes the partitions of the remaining lattice points $x_1, \ldots, x_6$. Thus, if we denote the polynomials corresponding to these partitions by $\rho_{0,0}(x), \ldots, \rho_{0,6}(x)$, then the sum in (12) is simply given by $\sum_{j=0}^{6} c[x_j]\rho_{0,j}(x_j)$. More generally, if x_0 intersects the partition P_i $(0 \leq i \leq 3)$, and if we denote the corresponding polynomials by $\rho_{i,j}(x)$, then the sum is given by $\sum_{j=0}^{6} c[x_j]\rho_{i,j}(x_j)$. Thus, we have the computational advantage that at most two binary decisions are required to simultaneously determine the ZP partitions corresponding to the points x_j, where the coefficients of the polynomials $\rho_{i,j}(x)$ can be pre-computed.

Acknowledgements

This work was supported by the Swiss National Science Foundation under grant 200020-109415, and the Ramón y Cajal program of the Spanish Ministry of Science

and Innovation.

References

[1] T. Asahi, K. Ichige, and R. Ishii, *An efficient algorithm for decomposition and reconstruction of images by box splines*, IEICE Trans. on Fundamentals of Electronics, Commun. and Comp. Sc. **E 84** (2001), no. 8, 1883–1891.

[2] C. de Boor, K. Höllig, and S. Riemenschneider, *Box Splines*, Springer-Verlag, 1993.

[3] K. N. Chaudhury, Zs. Püspöki, A. Muñoz Barrutia, D. Sage, and M. Unser, *Fast detection of cells using a continuously scalable Mexican-hat-like template*, IEEE International Symposium on Biomedical Imaging: From Nano to Macro (2010), in press.

[4] L. Condat and D. Van De Ville, *Three-directional box-splines: Characterization and efficient evaluation*, IEEE Signal Process. Lett. **13** (2006), no. 7, 417–420.

[5] R. Deriche, *Fast algorithms for low-level vision*, IEEE Trans. Pattern Anal. Mach. Intell. **12** (1990), no. 1, 78–87.

[6] A. Entezari, D. Van De Ville, and T. Möller, *Practical box splines for reconstruction on the body centered cubic lattice*, IEEE Trans. Vis. Comput. Graph. **14** (2008), no. 2, 313–328.

[7] J. M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer, *Fast anisotropic Gaussian filtering*, IEEE Trans. Image Process. **12** (2003), no. 8, 938–943.

[8] P. S. Heckbert, *Filtering by repeated integration*, Intl. Conf. on Computer Graphics and Interactive Techniques **20** (1986), no. 4, 315–321.

[9] B. Jähne, *Digital image processing*, Springer, 1997.

[10] S. Y. M. Lam and B. E. Shi, *Recursive anisotropic 2-D Gaussian filtering based on a triple-axis decomposition*, IEEE Trans. Image Process. **16** (2007), no. 7, 1925–1930.

[11] J.-S. Lee, *Digital image enhancement and noise filtering by use of local statistics*, IEEE Trans. Pattern Anal. Mach. Intell. (1980), 165–168.

[12] A. Muñoz-Barrutia, X. Artaechevarria, and C. Ortiz-de Solorzano, *Spatially variant convolution with scaled B-splines*, IEEE Trans. Image Process. **19** (2010), 11–24.

[13] A. Muñoz-Barrutia, R. Ertlé, and M. Unser, *Continuous wavelet transform with arbitrary scales and $\mathcal{O}(N)$ complexity*, Signal Processing **82** (2002), no. 5, 749–757.

[14] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, IEEE Trans. Pattern Anal. Mach. Intell. **12** (1990), no. 7, 629–639.

[15] M. Richter, *Use of box splines in computer tomography*, Computing **61** (1998), no. 2, 133–150.

[16] S. Tan, J. L. Dale, and A. Johnston, *Performance of three recursive algorithms for fast space-variant Gaussian filtering*, Real-Time Imaging **9** (2003), no. 3, 215–228.

[17] A. Tkacenko and P.P. Vaidyanathan, *Generalized kurtosis and applications in blind equalization of MIMO channels*, Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers **1** (2001), 742–746.

[18] X. Wang, *On the gradient inverse weighted filters*, IEEE Trans. Signal Process. **40** (1992), no. 2, 482–484.

[19] J. Weickert, *Theoretical foundations of anisotropic diffusion in image processing*, Theoretical foundations of Computer Vision. Computing **11** (1994), 221–236.

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

[21] P. B. Zwart, *Multivariate splines with nondegenerate partitions*, SIAM Journal on Numerical Analysis **10** (1973), no. 4, 665–673.