Three-directional Box-Splines: Characterization and Efficient Evaluation

Laurent Condat, *Student Member, IEEE,* and Dimitri Van De Ville, *Member, IEEE*

***Abstract*— We propose a new characterization of three-directional box-splines, which are well adapted for interpolation and approximation on hexagonal lattices. Inspired by a construction already applied with success for exponential splines [1] and hex-splines [2], we characterize a box-spline as a convolution of a generating function, that is a Green function of the spline's associated differential operator, and a discrete filter that plays the role of a localization operator. This process leads to an elegant analytical expression of three-directional box-splines. It also brings along a particularly efficient implementation.**

***Index Terms*— Box-splines, hexagonal sampling, three-directional mesh, interpolation, approximation.**

I. Introduction

The representation of a digital signal by means of a discrete/continuous model is essential for common tasks such as interpolation and resampling. For images and other two-dimensional (2-D) data, polynomial spline models based on B-splines are particularly popular, mainly due to their simplicity and excellent approximation capabilities [3].

For image data sampled on the traditional Cartesian lattice, separable B-splines can be obtained in a straightforward way using tensor products of one-dimensional (1-D) B-splines. However, in the case of sampling on a hexagonal lattice (a.k.a. three-directional mesh) separable B-splines are uncapable to exploit the highly praised isotropy and twelve-fold symmetry of this sampling scheme [4], [5]. *Box-splines* are a multi-dimensional extension of 1-D splines [6] that have found practical applications in geometric modelling, multiscale representation, and many other fields [7]. Among the large box-spline family, three-directional (non-separable) box-splines are particularly suitable for hexagonal lattices. They have been sucessfully applied in numerous problems where hexagonally sampled data are handled [8], [9].

Early algorithms to evaluate box-spline surfaces were very memory consuming and only resulted into an approximation of the surface within a given tolerance [10]–[12]. Later, more efficient methods were proposed based on the recursive properties of box-splines [13]–[16]. Here, we propose a new characterization of three-directional box-splines that provides us with a closed analytical formula, as well as an efficient implementation scheme. To this aim, we derive an explicit form of the generating function, which is the Green function of a three-directional differential operator associated with box-splines. Then, the box-spline can be expressed as the convolution of the generating function with a discrete filter, which plays the role of a localization operator. A similar construction was already applied on the Cartesian lattice to generalized polynomial splines (i.e., exponential splines and L-splines [1]) and to the design of another family of splines on the hexagonal lattice (i.e., hex-splines [2]).

The first author is with the Laboratory of Images and Signals (LIS), Institut National Polytechnique de Grenoble (INPG), 38031 Grenoble, France (e-mail: laurent.condat@lis.inpg.fr). The second author is with the Biomedical Imaging Group (BIG), École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland (e-mail: dimitri.vandeville@epfl.ch).

This work was initiated during the first author's visit at BIG, supported by Région Rhône-Alpes (EURODOC grant).

II. Box-splines on the hexagonal lattice

A. Mathematical preliminaries

A 2-D lattice is a set of points of the plane, characterized by two linearly independant vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ grouped in a matrix $\mathbf{R} = [\mathbf{v}_1\ \mathbf{v}_2]$, such that the lattice sites are the locations $\mathbf{Rk}$ for every $\mathbf{k} \in \mathbb{Z}^{2\times 1}$. Within this letter, we define the vectors $\mathbf{e}_1 = [1\ 0]^{\mathrm{T}}$, $\mathbf{e}_2 = [0\ 1]^{\mathrm{T}}$, and those shown in Fig. 1 as

$$\mathbf{r}_1 = \begin{bmatrix} 1/2 \\ -\sqrt{3}/2 \end{bmatrix}, \quad \mathbf{r}_2 = \begin{bmatrix} 1/2 \\ \sqrt{3}/2 \end{bmatrix}, \quad \mathbf{r}_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{1}$$

The Cartesian lattice is then obtained for $\mathbf{R} = [\mathbf{e}_1\ \mathbf{e}_2]$, and the regular hexagonal lattice, as in Fig. 1, for $\mathbf{R} = [\mathbf{r}_1\ \mathbf{r}_2]$.

Bivariate functions are equivalently denoted as $f(x_1, x_2)$, $x_1, x_2 \in \mathbb{R}$, or $f(\mathbf{x})$, where $\mathbf{x} = [x_1\ x_2]^{\mathrm{T}}$ is interpreted as a vector in $\mathbb{R}^2$. The Fourier transform of a function $f(\mathbf{x}) \in L_2(\mathbb{R}^2)$ is defined as $\hat{f}(\boldsymbol{\omega}) = \int_{\mathbb{R}^2} f(\mathbf{x}) \exp(-j\langle\boldsymbol{\omega}, \mathbf{x}\rangle) d\mathbf{x}$, where $\langle\boldsymbol{\omega}, \mathbf{x}\rangle = \boldsymbol{\omega}^{\mathrm{T}}\mathbf{x}$ is the usual inner product of vectors.

A 2-D discrete signal is denoted as $s[\mathbf{k}] = s[k_1, k_2]$, $k_1, k_2 \in \mathbb{Z}$. Its representation in the continuous domain, associated with the lattice sites $\mathbf{Rk}$, is a weighted Dirac comb: $s(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^2} s[\mathbf{k}]\delta(\mathbf{x} - \mathbf{Rk})$. Consequently, its Fourier transform is defined accordingly as $\hat{s}(\boldsymbol{\omega}) = \sum_{\mathbf{k}\in\mathbb{Z}^2} s[\mathbf{k}] \exp(-j\langle\boldsymbol{\omega}, \mathbf{Rk}\rangle)$. For $\mathbf{z} = \exp(-j\mathbf{R}^{\mathrm{T}}\boldsymbol{\omega})$, we get the $\mathcal{Z}$-transform of s as $S(\mathbf{z}) = \sum_{\mathbf{k}\in\mathbb{Z}^2} s[\mathbf{k}]\mathbf{z}^{-\mathbf{k}}$ ($\mathbf{z}^{-\mathbf{k}}$ means $z_1^{-k_1} z_2^{-k_2}$). Continuous and discrete convolutions are denoted by $*$.

B. Definition

A 2-D box-spline model defined on a lattice $\mathbf{R}$ has the form

$$f(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^2} c[\mathbf{k}]\varphi_{\Xi}(\mathbf{x} - \mathbf{Rk}), \quad \mathbf{x} \in \mathbb{R}^2, \tag{2}$$

where $c[\mathbf{k}]$ are the box-spline coefficients that are weights for the box-spline basis functions $\varphi_{\Xi}(\mathbf{x})$, placed on every lattice site. They can be computed to ensure a desired property, typically that f interpolates a discrete available signal s (*i.e.* $f(\mathbf{Rk}) = s[\mathbf{k}]$ for every $\mathbf{k}$). The box-spline $\varphi_{\Xi}(\mathbf{x})$ depends on a concatenated matrix of N vectors $\Xi = [\mathbf{v}_1\ \cdots\ \mathbf{v}_N]$

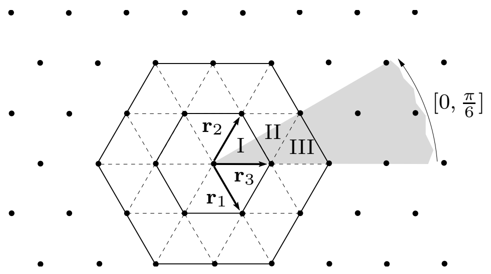

Fig. 1. The hexagonal lattice is generated using integer combinations of the vectors $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$. The hexagonal support of the first two box-splines $\chi^1(\mathbf{x})$ and $\chi^2(\mathbf{x})$ has been indicated. Box-splines are polynomial inside each triangle. Using the twelve-fold symmetry, χ^1 and χ^2 have only to be known in the triangles I, II, III, that intersect the sector $[0, \pi/6]$.

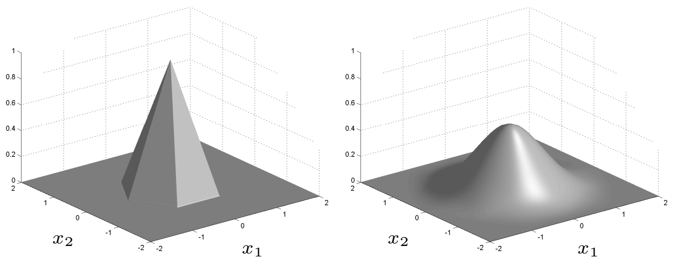

Fig. 2. First two box-splines $\chi^1(\mathbf{x})$ (left) and $\chi^2(\mathbf{x})$ (right).

($N \geq 2$), and can be defined as follows [6]: if $\Xi = [\mathbf{v}_1 \;\; \mathbf{v}_2]$, then

$$\varphi_{[\mathbf{v}_1 \;\; \mathbf{v}_2]}(\mathbf{x}) = \begin{cases} 1/|\det(\Xi)|, & \text{if } \Xi^{-1}\mathbf{x} \in [0,1)^2, \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

and inductively, $\varphi_{\Xi\cup[\mathbf{v}]}(\mathbf{x}) = \int_0^1 \varphi_\Xi(\mathbf{x} - t\mathbf{v})dt$. Therefore, we have the normalization $\int_{\mathbb{R}^2} \varphi_\Xi = 1$ and the convolution property $\varphi_{\Xi_1\cup\Xi_2} = \varphi_{\Xi_1} * \varphi_{\Xi_2}$.

On a hexagonal lattice, box-splines can be constructed using the three vectors $\mathbf{r}_1$, $\mathbf{r}_2$, $-\mathbf{r}_3$. In particular, we define the so-called Courant element [6] as $\chi^1 = \frac{\sqrt{3}}{2}\varphi_{[\mathbf{r}_1 \;\mathbf{r}_2\; -\mathbf{r}_3]}$, where we have changed the normalization towards the density of the lattice, i.e., $|\det \mathbf{R}| = \frac{\sqrt{3}}{2}$. Further on, higher orders are obtained as $\chi^n = \frac{2}{\sqrt{3}}\chi^{n-1} * \chi^1$, $n > 1$. Their expression in the Fourier domain is

$$\widehat{\chi}^n(\boldsymbol{\omega}) = \frac{\sqrt{3}}{2}\left(\frac{\exp(j\langle\boldsymbol{\omega},\mathbf{r}_3\rangle)\prod_{i=1}^3 1 - \exp(-j\langle\boldsymbol{\omega},\mathbf{r}_i\rangle)}{(j\langle\boldsymbol{\omega},\mathbf{r}_1\rangle)(j\langle\boldsymbol{\omega},\mathbf{r}_2\rangle)(j\langle\boldsymbol{\omega},\mathbf{r}_3\rangle)}\right)^n \tag{4}$$

$$= \frac{\sqrt{3}}{2}\prod_{i=1}^3 \operatorname{sinc}(\langle\boldsymbol{\omega},\mathbf{r}_i\rangle/2)^n. \tag{5}$$

where $\operatorname{sinc}(x) = \sin(x)/x$. The box-splines $\chi^n(\mathbf{x})$ have several attractive properties such as an hexagonal compact support and twelve-fold symmetry, as illustrated in Figs 1 and 2. In the next section, we provide closed analytical formulas for these box-splines in the spatial domain.

III. Differential characterization of box-splines

A. B-spline refresher

In the 1-D case, a polynomial spline $f(x)$ for uniformly sampled data can be expressed similarly to (2) as $f(x) = \sum_{k\in\mathbb{Z}} c[k]\beta^n(x-k)$. β^n is the causal B-spline of degree $n \in \mathbb{N}$, which is defined in the spatial domain as[1]

$$\beta^n(x) = \Delta^{n+1} * (x)_+^n/n!. \tag{6}$$

We identify Δ^n as the nth iterate of the finite difference filter, which is usually expressed in the $\mathcal{Z}$-domain as $\Delta^n(z) = (1-z^{-1})^n$. Further on, we have the one-sided power function $(x)_+^n = \{x^n, \text{for } x > 0; 0, \text{otherwise}\}$. The filtering process acts as a localization operator on the power function; i.e., β^n has a finite support. The term $(x)_+^n/n!$ is also called the generating function and it corresponds to the (causal) Green function of the differential operator $\mathrm{L}^n = \mathrm{d}^n/\mathrm{d}x^n$; i.e., the function $\rho(x)$ such that $\mathrm{L}^n\{\rho\}(x) = \delta(x)$. This means that a polynomial spline of degree n, when differentiated $n+1$ times, is a weighted Dirac comb.

[1]Note that the 1-D B-spline can also be obtained recursively as $\beta^n = \beta^{n-1} * \beta^0$, $n > 0$, with $\beta^0 = \mathbb{1}_{[0,1)}$ the indicator function of the unit interval.

On the 2-D Cartesian lattice, we can easily use tensor-product B-splines: $\beta^n(\mathbf{x}) = \beta^n(x_1)\beta^n(x_2)$. Then, the associated differential operator is

$$\mathrm{L}^n = \frac{\partial^{2n}}{\partial x_1^n \partial x_2^n} = \mathrm{D}^n_{\mathbf{e}_1}\mathrm{D}^n_{\mathbf{e}_2} \overset{\mathcal{F}}{\longleftrightarrow} (j\langle\boldsymbol{\omega},\mathbf{e}_1\rangle)^n(j\langle\boldsymbol{\omega},\mathbf{e}_2\rangle)^n, \tag{7}$$

where $\mathrm{D}_\mathbf{v} f(\mathbf{x}) = \lim_{t\to 0}(f(\mathbf{x}+t\mathbf{v}) - f(\mathbf{x}))/t$. In that case, the (separable) generating function is $(\mathbf{x})_+^n/(n!)^2 = (x_1)_+^n(x_2)_+^n/(n!)^2$ and the corresponding localization operator $\Delta^n(\mathbf{z}) = \Delta^n(z_1)\Delta^n(z_2)$.

B. From differential operators to generating functions

Inspired by the B-spline construction using Green functions, we propose an extension for the box-splines on the hexagonal lattice. For this purpose, we introduce the three-directional differential operator $\mathrm{L}^n = \frac{2}{\sqrt{3}}\mathrm{D}^n_{\mathbf{r}_1}\mathrm{D}^n_{\mathbf{r}_2}\mathrm{D}^n_{\mathbf{r}_3}$, $n \geq 1$. Its Fourier transform, in the sense of the distributions, is

$$\widehat{\mathrm{L}}^n(\boldsymbol{\omega}) = \frac{2}{\sqrt{3}}(j\langle\boldsymbol{\omega},\mathbf{r}_1\rangle)^n(j\langle\boldsymbol{\omega},\mathbf{r}_2\rangle)^n(j\langle\boldsymbol{\omega},\mathbf{r}_3\rangle)^n. \tag{8}$$

Proposition: A Green function $\rho^n(\mathbf{x})$ of the operator L^n, $n \geq 1$, is given by

$$\rho^n(\mathbf{x}) = \sum_{i=0}^{n-1}\binom{n-1+i}{i}\mu^{n-1-i,2n-1+i}(\mathbf{x}), \tag{9}$$

where

$$\mu^{n_1,n_2}(x_1,x_2) = \frac{1}{n_1!n_2!}\left(\frac{2|x_2|}{\sqrt{3}}\right)^{n_1}\left(x_1 - \frac{|x_2|}{\sqrt{3}}\right)_+^{n_2}. \tag{10}$$

The proof is given in Appendix I. Notice that the functions μ^{n_1,n_2} and ρ^n all have the same wedge-like support; they are causal in x_1 and symmetric in x_2, as illustrated in Fig. 3.

C. From generating functions to box-splines

In the Fourier domain, the generating function ρ^n corresponds to $\widehat{\chi}^n$ without its numerator in (4). The remaining term can be identified by introducing the discrete filter

$$\Delta(\mathbf{z}) = (1-z_1^{-1})(1-z_2^{-1})(z_1z_2 - 1). \tag{11}$$

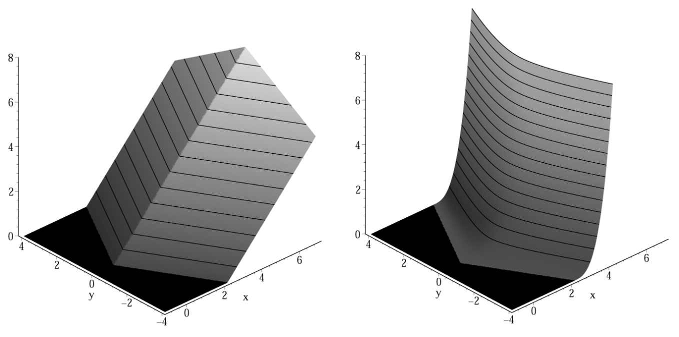

Fig. 3. The Green functions $\rho^1 = \mu^{0,1}$ in (a), and $\rho^2 = \mu^{1,3} + 2\mu^{0,4}$ in (b), which serve to generate the box-splines χ^1 and χ^2.

Using the property $\mathbf{r}_3 = \mathbf{r}_1 + \mathbf{r}_2$, we find that $\widehat{\Delta}^n(\boldsymbol{\omega}) = \Delta(\exp(j\langle\boldsymbol{\omega}, \mathbf{r}_1\rangle), \exp(j\langle\boldsymbol{\omega}, \mathbf{r}_2\rangle))^n$ is exactly the numerator of (4). We can explicitly find the filter coefficients of Δ^n by expanding the n-th power of the $\mathcal{Z}-$transform of (11). By collecting the coefficient in front of the term $z_1^{-k_1} z_2^{-k_2}$, we get for every $k_1, k_2 \in \mathbb{Z}$:

$$\Delta^n[k_1,k_2] = \sum_{i=\max(k_1,k_2,0)}^{\min(n+k_1,n+k_2,n)} (-1)^{k_1+k_2+i} \binom{n}{i-k_1}\binom{n}{i-k_2}\binom{n}{i}. \tag{12}$$

By arranging the $\Delta^n[\mathbf{k}]$ at the lattice sites $\mathbf{Rk} = k_1\mathbf{r}_1 + k_2\mathbf{r}_2$, we can represent the first two localization filters as:

$$\Delta = \begin{array}{ccccc} & -1 & & 1 & \\ 1 & & 0 & & -1 \\ & -1 & & 1 & \end{array}, \qquad \Delta^2 = \begin{array}{ccccccccc} & & 1 & & -2 & & 1 & & \\ & -2 & & 2 & & 2 & & -2 & \\ 1 & & 2 & & -6 & & 2 & & 1 \\ & -2 & & 2 & & 2 & & -2 & \\ & & 1 & & -2 & & 1 & & \end{array}. \tag{13}$$

Putting together (11) and (8) with the fact that $\widehat{\mathrm{L}}^n(\boldsymbol{\omega})\widehat{\rho}^n(\boldsymbol{\omega}) = 1$, we find that $\widehat{\chi}^n(\boldsymbol{\omega}) = \widehat{\Delta}^n(\boldsymbol{\omega})\widehat{\rho}^n(\boldsymbol{\omega})$. Therefore, we obtain the characterization:

$$\chi^n(\mathbf{x}) = \Delta^n * \rho^n(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbf{Z}^2} \Delta^n[\mathbf{k}]\rho^n(\mathbf{x} - \mathbf{Rk}). \tag{14}$$

The complete analytical expression of $\chi^n(\mathbf{x})$, $n \geq 1$, can then be written as $\chi^n(x_1, x_2) =$

$$\sum_{k_1,k_2=-n}^{n} \sum_{i=\max(k_1,k_2,0)}^{\min(n+k_1,n+k_2,n)} (-1)^{k_1+k_2+i} \binom{n}{i-k_1}\binom{n}{i-k_2}\binom{n}{i}$$
$$\sum_{d=0}^{n-1} \binom{n-1+d}{d} \frac{1}{(2n-1+d)!(n-1-d)!} \tag{15}$$
$$\left|\frac{2x_2}{\sqrt{3}} + k_1 - k_2\right|^{n-1-d} \left(x_1 - \frac{k_1+k_2}{2} - \left|\frac{x_2}{\sqrt{3}} + \frac{k_1-k_2}{2}\right|\right)_+^{2n-1+d}.$$

IV. Implementation issues

A. The generic case

Equation (15) provides us with an efficient way to evaluate at any point $\mathbf{x}$, any three-directional box-spline χ^n. Notice that the power functions grow rapidly, as shown in Fig. 3, which could lead to problems of numerical stability. A simple remedy consists of evaluating χ^n only for $x_1 \leq 0$, which exploits the causality of ρ^n in x_1 and the symmetry $\chi^n(x_1, x_2) = \chi^n(-x_1, x_2)$. The following Matlab code performs box-spline evaluations for a list of points `(x[m],y[m])`, indexed by `m`. The twelve-fold symmetry is used to fold coordinates into the sector $[5\pi/6, \pi]$, where the number of evaluations of the power functions is minimal. We use the coordinates (u, v) in the basis $(\mathbf{r}_1, \mathbf{r}_2)$, instead of the coordinates (x, y) in the canonical basis $(\mathbf{e}_1, \mathbf{e}_2)$. `nchoosek(n,k)` gives the binomial coefficient (n, k).

```
function val=boxspline(x,y,n)
x=-abs(x);  y=abs(y);
u=x-y/sqrt(3);  v=x+y/sqrt(3);
id=find(v>0);   v(id)=-v(id);  u(id)=u(id)+v(id);
id=find(v>u/2); v(id)=u(id)-v(id);
val=zeros(size(x));
for K=-n:ceil(max(max(u)))-1,
  for L=-n:ceil(max(max(v)))-1,
    for i=0:min(n+K,n+L),
      coeff=(-1)^(K+L+i)*nchoosek(n,i-K)*...
        nchoosek(n,i-L)*nchoosek(n,i);
      for d=0:n-1,
        aux=abs(v-L-u+K);
        aux2=(u-K+v-L-aux)/2;
        aux2(find(aux2<0))=0;
        val=val+coeff*nchoosek(n-1+d,d)/...
          factorial(2*n-1+d)/factorial(n-1-d)*...
          aux.^(n-1-d).*aux2.^(2*n-1+d);
end, end, end, end
```

This code was used to generate the plots in Fig. 2. The computational complexity is polynomial in n, compared to exponential for recursive methods in the literature [13]–[16]. For example, the evaluation `boxspline(1,1,3)` took $0.002s$, while $47s$ were required for the same operation using the Matlab code proposed in [15] (that can evaluate any box-spline, not just the three-directional ones).

B. Further optimization for fixed n

For evaluating a box-spline χ^n of fixed n, an attractive hybrid analytical/numerical implementation consists in determining the polynomial form $p(\mathbf{x})$ inside each triangle of the three-directional mesh. This polynomial, which is obtained by the sums of (15), can be precomputed, stored, and only evaluated at the end. The following code in C-language for χ^2 may serve as a template: coordinates are first folded in the sector $[0, \pi/2]$, then in $[0, \pi/3]$ and finally in $[0, \pi/6]$. This is done conveniently with the coordinates (u, v). The coordinates $(g = u - v/2, v)$ in the orthogonal basis $(\mathbf{r}_1, (\mathbf{r}_2 + \mathbf{r}_3)/2)$ are the most appropriate for having short polynomials with rational coefficients in each triangle.

```
 float boxspline2(float x, float y) {
  float u=fabs(x)-fabs(y)/sqrt(3.0);
  float v=fabs(x)+fabs(y)/sqrt(3.0);
  if (u<0)   { u=-u; v=v+u; }      /*symmetry % r2*/
  if (2*u<v) u=v-u;               /*symmetry % r2+r3*/
  float g=u-v/2.0;
  if (v>2.0) return 0.0;      /*outside the support*/
  if (v<1.0) return 0.5+((5/3.0-v/8.0)*v-3)*v*v/4.0+
    ((1-v/4.0)*v+g*g/6.0-1)*g*g;      /*triangle I*/
  if (u>1.0) return (v-2)*(v-2)*(v-2)*(g-1)/6.0;
  return 5/6.0+((1+(1/3.0-v/8.0)*v)*v/4.0-1)*v+
    ((1-v/4.0)*v+g*g/6.0-1)*g*g;     /*triangle II*/
}
```

V. Conclusion

We proposed a new characterization of the three-directional box-splines, based on a Green function of the differential operator adapted to the hexagonal lattice. Together with a finite difference filter that acts as a localization operator on the generating function, this provides us with new explicit analytical formulas for the three-directional box-splines. This characterization also leads to particularly easy and efficient implementations. We provided the Matlab source code for the generic case and a further optimized C-code for the case $n = 2$. The latter one could be particularly interesting for high-quality visualization of data sampled on a hexagonal lattice.

Finally, we note that these box-splines can be expressed on any lattice with matrix $\mathbf{R}'$, and not only on the hexagonal one, by the simple change of basis $\chi^n(\mathbf{R}\mathbf{R}'^{-1}\mathbf{x})$.

Appendix I
Proof of the Proposition

We verify whether ρ^n of (9) is a Green function of L^n; i.e., we need $\mathrm{L}^n\left\{\rho^n\right\}(\mathbf{x}) = \delta(\mathbf{x})$. First, we introduce the vectors

$$\mathbf{r}_1^\perp = \left[\begin{array}{c} 1 \\ 1/\sqrt{3} \end{array}\right],\ \mathbf{r}_2^\perp = \left[\begin{array}{c} 1 \\ -1/\sqrt{3} \end{array}\right],\ \mathbf{r}_3^\perp = \left[\begin{array}{c} 0 \\ 2/\sqrt{3} \end{array}\right], \tag{16}$$

which allow us to express the dual bases of $(\mathbf{r}_2, \mathbf{r}_3)$ and $(\mathbf{r}_1, \mathbf{r}_3)$ as $(\mathbf{r}_3^\perp, \mathbf{r}_2^\perp)$ and $(-\mathbf{r}_3^\perp, \mathbf{r}_1^\perp)$, respectively. For example, the coordinates of $\mathbf{x}$ in $(\mathbf{r}_2, \mathbf{r}_3)$ are $(\langle \mathbf{x}, \mathbf{r}_3^\perp\rangle, \langle \mathbf{x}, \mathbf{r}_2^\perp\rangle)$.

We now derive the Fourier expression of μ^{n_1,n_2}, which we first rewrite as

$$\begin{aligned}\mu^{n_1,n_2}(x_1,x_2) &= (x_2)_+^0\mu^{n_1,n_2}(x_1,x_2) + (-x_2)_+^0\mu^{n_1,n_2}(x_1,x_2) \\ &= (\langle \mathbf{x}, \mathbf{r}_3^\perp\rangle)_+^{n_1}(\langle \mathbf{x}, \mathbf{r}_2^\perp\rangle)_+^{n_2} + (\langle \mathbf{x}, -\mathbf{r}_3^\perp\rangle)_+^{n_1}(\langle \mathbf{x}, \mathbf{r}_1^\perp\rangle)_+^{n_2}.\end{aligned}$$

From distribution theory, we know the Fourier transform of the one-sided power function

$$(x)_+^n \xleftrightarrow{\mathcal{F}} \frac{n!}{(j\omega)^{n+1}} + \mathcal{D}(\boldsymbol{\omega}), \tag{17}$$

where $\mathcal{D}$ is essentially the n-th derivative of Dirac. This term can be omitted since it does not have any influence when applying a differential operator of order n (continuous or discrete) to $(x)_+^n$, also see [2, Appendix C].

Hence, using a tensor product and a change of basis from $(\mathbf{e}_1, \mathbf{e}_2)$ to $(\mathbf{r}_2, \mathbf{r}_3)$ (with Jacobian $|\det[\mathbf{r}_2\ \mathbf{r}_3]| = \frac{\sqrt{3}}{2}$), we get

$$(x_2)_+^0\mu^{n_1,n_2} \xleftrightarrow{\mathcal{F}} \frac{\sqrt{3}/2}{(j\langle\boldsymbol{\omega}, \mathbf{r}_2\rangle)^{n_1+1}(j\langle\boldsymbol{\omega}, \mathbf{r}_3\rangle)^{n_2+1}}. \tag{18}$$

Similarly, the Fourier transform of $(-x_2)_+^0\mu^{n_1,n_2}$ is obtained by replacing $\mathbf{r}_2$ by $\mathbf{r}_1$ in (18).

We now define the functions γ^{n_1,n_2,n_3}, for any integers n_1, n_2, n_3 as

$$\gamma^{n_1,n_2,n_3} \xleftrightarrow{\mathcal{F}} \frac{\sqrt{3}/2}{(j\langle\boldsymbol{\omega}, \mathbf{r}_1\rangle)^{n_1}(j\langle\boldsymbol{\omega}, \mathbf{r}_2\rangle)^{n_2}(j\langle\boldsymbol{\omega}, \mathbf{r}_3\rangle)^{n_3}}. \tag{19}$$

We recognize $\rho^n = \gamma^{n,n,n}$, $n \geq 1$. Using the property $\mathbf{r}_3 = \mathbf{r}_1 + \mathbf{r}_2$, we can further obtain the following recurrence relation, for $n_1 \geq 1, n_2 \geq 1, n_3 \geq 0$:

$$\gamma^{n_1,n_2,n_3} = \gamma^{n_1-1,n_2,n_3+1} + \gamma^{n_1,n_2-1,n_3+1}. \tag{20}$$

By recurrence on $n_1 + n_2$, we can also show that

$$\begin{aligned}\gamma^{n_1,n_2,n_3} &= \sum_{i=0}^{n_1-1}\binom{n_2-1+i}{i}\gamma^{n_1-i,0,n_2+n_3+i} \\ &\quad + \sum_{i=0}^{n_2-1}\binom{n_1-1+i}{i}\gamma^{0,n_2-i,n_1+n_3+i}.\end{aligned} \tag{21}$$

In the case of ρ^n, we have

$$\rho^n = \sum_{i=0}^{n-1}\binom{n-1+i}{i}\left(\gamma^{n-i,0,2n+i} + \gamma^{0,n-i,2n+i}\right). \tag{22}$$

Finally, we identify the function μ^{n_1,n_2} as

$$\mu^{n_1,n_2} = \gamma^{0,n_1+1,n_2+1} + \gamma^{n_1+1,0,n_2+1}, \tag{23}$$

which results into (9).

References

[1] M. Unser and T. Blu, "Cardinal exponential splines: Part I—Theory and filtering algorithms," *IEEE Trans. Signal Processing*, vol. 53, no. 4, pp. 1425–1438, Apr. 2005.

[2] D. Van De Ville, T. Blu, M. Unser, W. Philips, I. Lemahieu, and R. Van De Walle, "Hex-spline: a novel family for hexagonal lattices," *IEEE Trans. Image Processing*, vol. 13, no. 6, pp. 758–772, June 2004.

[3] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Signal Processing Mag.*, vol. 16, no. 6, pp. 22–38, Nov. 1999.

[4] R. M. Mersereau, "The processing of hexagonally sampled two-dimensional signals," *Proc. IEEE*, vol. 67, no. 6, pp. 930–949, June 1979.

[5] D. P. Petersen and D. Middleton, "Sampling and reconstruction of wavenumber- limited functions in N-dimensional Euclidean spaces," *Information and Control*, vol. 5, pp. 279–323, 1962.

[6] C. de Boor, K. Höllig, and S. Riemenschneider, *Box Splines*. Berlin: Springer-Verlag, 1993, vol. Applied Mathematical Sciences, vol. 98.

[7] R. DeVore and A. Ron, "Developing a computation-friendly mathematical foundation for spline functions," *SIAM News*, vol. 38, no. 4, May 2005.

[8] S. Malassiotis and M. G. Strintzis, "Optimal biorthogonal wavelet decomposition of wire-frame meshes using box splines, and its application to the hierarchical coding of 3-D surfaces," *IEEE Trans. Image Processing*, vol. 8, no. 1, pp. 41–57, Jan. 1999.

[9] H. Prautzsch and W. Boehm, "Box splines," in *Handbook of Computer Aided Geometric Design*. Berlin: Springer, 2001.

[10] E. Cohen, T. Lyche, and R. Riesenfeld, "Discrete box splines and refinement algorithms," *Comput. Aided Geom. Design*, vol. 1, pp. 131–141, 1984.

[11] W. Dahmen and C. A. Michelli, "Subdivision algorithms for the generation of box-spline surfaces," *Comput. Aided Geom. Design*, vol. 1, pp. 115–129, 1984.

[12] ——, "Line average algorithm: a method for the computer generation of smooth surfaces," *Comput. Aided Geom. Design*, vol. 2, pp. 77–85, 1985.

[13] C. K. Chui and M.-J. Lai, "Algorithms for generating B-nets and graphically displaying spline surfaces on three- and four-directional meshes," *Comput. Aided Geom. Design*, vol. 8, no. 6, pp. 479–493, 1991.

[14] M.-J. Lai, "Fortran subroutines for B-nets of box splines on three and four directional meshes," *Numerical Algorithms*, vol. 2, pp. 33–38, 1992.

[15] C. D. Boor, "On the evaluation of box splines," *Numerical Algorithms*, vol. 5, pp. 5–23, Mar. 1993.

[16] L. Kobbelt, "Stable evaluation of box-splines," *Numerical Algorithms*, vol. 14, no. 4, pp. 377–382, 1997.