B-Spline Signal Processing: Part I—Theory

Michael Unser, *Member, IEEE,* Akram Aldroubi, and Murray Eden, *Life Fellow, IEEE*

Abstract—**This paper describes a set of efficient filtering techniques for the processing and representation of signals in terms of continuous B-spline basis functions. We first consider the problem of determining the spline coefficients for an exact signal interpolation (direct B-spline transform). The reverse operation is the signal reconstruction from its spline coefficients with an optional zooming factor m (indirect B-spline transform). We derive general expressions for the z transforms and the equivalent continuous impulse responses of B-spline interpolators of order n. We present simple techniques for signal differentiation and filtering in the transformed domain. We then derive recursive filters that efficiently solve the problems of smoothing spline and least squares approximations. The smoothing spline technique approximates a signal with a complete set of coefficients subject to certain regularization or smoothness constraints. The least squares approach, on the other hand, uses a reduced number of B-spline coefficients with equally spaced nodes; this technique is in many ways analogous to the application of antialiasing low-pass filter prior to decimation in order to represent a signal correctly with a reduced number of samples.**

I. Introduction

In most image processing applications, the pictures to be manipulated are represented by a set of uniformly spaced sampled values. Although most processing algorithms are derived within a purely discrete framework [1], there is a variety of problems best formulated by considering a real-valued picture function $g(x, y)$ defined over the real plane R^2; for examples in computer vision see [2]. An obvious approach is to fit a parametrized continuous image model to the observed data points and to derive algorithms that operate on the model's parameter values directly. Piecewise bidimensional polynomial functions are frequently used in this context [3], [4]. One usually has the choice of two options. The first is to use an exact representation by which $g(x, y)$ precisely interpolates the sampled values. The second is to use an approximate representation in which the function parameters are determined by minimizing some measure of the discrepancy between pixel values and $g(x, y)$ at the grid points. This approach usually has fewer degrees of freedom than the previous one, or, at least, some built-in smoothness constraints, which may make it more robust in the presence of noise.

Manuscript received August 11, 1990; revised March 2, 1992.

M. Unser is with the Biomedical Engineering and Instrumentation Program, National Institutes of Health, Bethesda, MD 20892, on leave from INSERM U.2, Hôpital Henri-Mondor, F-94010 Créteil, France.

A. Aldroubi and M. Eden are with the Biomedical Engineering and Instrumentation Program, National Institutes of Health, Bethesda, MD 20892.

IEEE Log Number 9205111.

Edge detection is a good example in which the use of a continuous signal representation is particularly apposite. Most algorithms are based on the evaluation of spatial gradients or Laplacians [5]. Early techniques relied on finite differences to estimate these quantities [6], [7]; however, these simple operators used on noisy images perform poorly. More recent approaches often depend on the concept of fitting a continuous surface locally to the data [5], [8], [9]. Haralick used local least squares polynomial fits to determine the zero crossing of the directional second derivatives [9]. Poggio *et al.* proposed a smoothing cubic spline technique to improve the estimation of the intensity gradient in the presence of noise [10], [11]. These authors showed the approach to be more or less equivalent to smoothing the image with a Gaussian low-pass filter in a preprocessing step. In fact, an initial smoothing operation is implicit to all least squares techniques and is used in almost any modern edge detection scheme [5], [12], [13]. The surface fitting concept is also well suited to estimating multiple-order derivatives, although alternative design procedures such as the extension of difference operators or the use of an optimal Wiener filter for noise reduction may also be used [14]. There are a variety of additional problems in computer vision (optical flow, surface reconstruction, the recovery of lightness and color, shape from shading and stereo matching) that are most conveniently formulated in terms of differential equations involving continuous image models [2]. Here again, the use of surface approximation techniques appears to be an interesting alternative to more conventional finite difference methods, which are notoriously unstable in the presence of noise [15]. Finally, with the recent development of multiresolution techniques [16], [17], there is a strong need for continuous image representations compatible with varying levels of resolutions, and which facilitate the transition from one scale to another.

When contrasted with local or running polynomial fits, the use of B-splines functions [18]–[21], which are piecewise polynomials as well, seems to have a number of advantages. First, higher order polynomials tend to oscillate while spline functions are usually smooth and well behaved. Second, the juxtaposition of local polynomial approximations may produce strong discontinuities in the connecting regions. B-spline surfaces, by contrast, are continuous everywhere. The polynomial segments are patched together so that the interpolating function and all derivatives up to order $n - 1$ are continuous at all joining intersections. Third, there is exactly one B-spline coefficient associated with each grid point and the range of these coefficients is of the same order of magnitude as that of the initial gray level values [20]. This property greatly facilitates the storage of B-spline representations and allows the use of standard picture arrays. Finally, as will be shown here, either exact or approximate B-spline signal representations can be evaluated quite efficiently.

Since the early paper of Hou and Andrews, which provides a detailed analysis of cubic spline interpolation [22], the use of B-spline representations has had limited application in signal processing. It would appear that the main reason for this lack of acceptance is because the conventional approach to B-spline interpolation or approximation is computationally quite expensive for it involves explicit matrix inversions and multiplications [22]. In principle, this problem can be alleviated by using more efficient computational techniques for the fast solution of banded systems of equations; such algorithms are to be found in the approximation theory or numerical methods literature [20], [23]. In the context of signal processing where the spacing between the data points is constant, there is yet another simpler approach. In recent years, some authors have come to realize that the operations involved are translation invariant and that B-spline coefficients could be determined efficiently through linear filtering. Toraichi *et al.* [24] have studied quadratic spline interpolation and have derived a finite impulse response filter approximating this operator. Poggio *et al.* [11], [25] have suggested using smoothing cubic splines to regularize differentiation and have shown that the variational formulation of a Tikhonov regularization leads to a Gaussian-like convolution filter. Elsewhere, we have considered the general case of B-spline interpolation of any order and have provided simple mechanisms for the design of filters to evaluate the direct or indirect B-spline transforms [26]. We have also brought out the recursive structure of these operators, which is the key to the design of fast computational algorithms.

The main purpose of this paper is to extend these recent results to signal approximation and to present a new class of processing techniques based on the representation of a signal in terms of continuous B-spline basis functions. In Section II, we introduce some preliminary definitions and review some essential properties of continuous B-spline functions. The basic approach for B-spline interpolation is summarized in Section III, which also includes some extensions of our previous results [26]. Finally, in Section IV, we describe new signal processing techniques for efficient signal differentiation, filtering, smoothing spline, and least squares approximations. The major result is that all these operations can be performed using space invariant linear operators completely characterized in terms of their z transforms.

The present paper is concerned primarily with the theoretical aspects of B-spline processing. Efficient implementation techniques, examples of applications, as well as the interpretation of these results in the context of image processing will be discussed in a companion paper [27].

II. Preliminaries

A. Discrete Signals and Operators

l_2 in the space of square-summable real-valued sequences $\{a(k)\}_{k\in Z}$. l_2 is a Hilbert space whose metric $\|\cdot\|$ (the l_2-norm) is derived from the standard inner product

$$\langle a, b\rangle = \sum_{k=-\infty}^{+\infty} a(k)b(k). \tag{2.1}$$

Some of the later derivations will involve the differentiation of such inner products with respect to a given signal. In Appendix A, we give the rules of this calculus useful for our purpose.

The convolution between two sequences $a \in l_2$ and $b \in l_2$ is denoted by $b*a(k)$. The sequence b may be viewed as a discrete convolution operator (or digital filter) that is applied to a; it is entirely characterized by its z transform (or transfer function) $B(z)$. If $B(z)$ has no zeros on the unit circle, then the inverse operator $(b)^{-1}$ exists and is uniquely defined by the equation

$$(b)^{-1}(k) \overset{z}{\leftrightarrow} 1/B(z). \tag{2.2}$$

We also use the symbol $'$ to denote the adjoint operator that reverses a signal:

$$b'(k) = b(-k) \overset{z}{\leftrightarrow} B(1/z).$$

Another useful operator is the up-sampling by an integer multiple m, which is defined as

$$[b]_{\uparrow m}(k) := \begin{cases} b(k') & \text{for } k = mk' \\ 0 & \text{otherwise} \end{cases} \overset{z}{\leftrightarrow} B(z^m). \tag{2.3}$$

The dual operation is the down-sampling (or decimation) by an integer m:

$$[b]_{\downarrow m}(k) := b(mk) \overset{z}{\leftrightarrow} \frac{1}{m}\sum_{k=0}^{m-1} B([ze^{j2\pi k}]^{1/m}) \tag{2.4}$$

where $j = \sqrt{-1}$.

B. Continuous Polynomial Splines and B-Spline Functions

In this paper, we are concerned with the problem of constructing polynomial splines that interpolate or approximate a given sequence $g(k) \in l_2$; these functions are piecewise polynomials that satisfy some specific continuity constraints. The generic space of polynomial splines of order n is denoted by S_1^n, where the superscript n refers to the degree of the piecewise polynomial segments, and where the subscript represents the spacing between the knots (i.e., the joining points of the polynomial segments). More precisely, S_1^n is the subset of functions of L_2 (the space of square integrable functions) that are of class C^{n-1} (i.e., continuous functions with continuous derivatives up to order $n - 1$) and are equal to a polynomial of degree n on each interval $[k, k+1)$, $k \in Z$ when n is odd, and $[k - \frac{1}{2}, k + \frac{1}{2})$, $k \in Z$ when n is even. In the present context, it is especially convenient to use an equivalent definition of S_1^n proposed by Schoenberg (cf. [19, p. 199, theorem 12]):

$$S_1^n = \left\{ g^n(x) = \sum_{k=-\infty}^{+\infty} y(k)\beta^n(x-k), \ (x \in R, \ y \in l_2) \right\} \tag{2.5}$$

where $\beta^n(x)$ is the symmetrical B-spline of order n

$$\beta^n(x) := \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \left(x + \frac{n+1}{2} - j \right)^n \cdot \mu\left(x + \frac{n+1}{2} - j \right), \quad (x \in R) \tag{2.6}$$

and where $\mu(x)$ is the unit step function

$$\mu(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0. \end{cases}$$

This definition essentially states that any polynomial spline $g^n(x) \in S_1^n$ can be constructed from a weighted sum of shifted B-splines and is uniquely characterized by the discrete sequence of its spline coefficients $y(k)$.

Since the set of shifted B-splines $\{\beta^n(x-k), k \in Z\}$ constitutes a basis of S_1^n, the study of these functions will uncover most of the properties of polynomial splines. For instance, it is clear from (2.6), that the $\beta^n(x)$'s are made of piecewise polynomials of degree n that are connected at the knot points. Interestingly enough, these functions can be constructed recursively:

$$\beta^n(x) = \frac{\left(\frac{n+1}{2} + x\right)\beta^{n-1}\left(x + \frac{1}{2}\right) + \left(\frac{n+1}{2} - x\right)\beta^{n-1}\left(x - \frac{1}{2}\right)}{n}. \tag{2.7}$$

Their derivatives can also be obtained in a recursive fashion based on the following property

$$\frac{\partial \beta^n(x)}{\partial x} = \beta^{n-1}\left(x + \frac{1}{2}\right) - \beta^{n-1}\left(x - \frac{1}{2}\right). \tag{2.8}$$

Another fundamental result is the well-known convolution property:

$$\beta^n(x) = \beta^{n-1} * \beta^0(x) = \underbrace{\beta^0 * \beta^0 * \cdots * \beta^0}_{n+1 \text{ times}}(x) \tag{2.9}$$

which states that a B-spline of order n can be generated by convolving β^0 $(n + 1)$ times with itself; the function $\beta^0(x)$ is a centered normalized rectangular pulse. Based on (2.9), it is rather straightforward to show that all B-splines are positive and have an integral that is equal to one. For a more detailed discussion of these properties, refer to [18], [20], [21].

III. Discrete B-Splines

Studying the properties of discrete (or sampled) B-splines is essential to the design of efficient algorithms for representing discrete signals in terms of such basis functions. In this section, we summarize the most important results reported initially in [26] with some additional formulas for the efficient evaluation of discrete B-splines. We also emphasize an interpretation of polynomial spline interpolators in terms of their continuous impulse response (cardinal spline) and provide a simple expression for the frequency response of such systems.

A. Definition and Properties

We define the discrete centered and shifted B-splines by sampling the corresponding continuous functions expanded by an integer factor of m:

$$b_m^n(k) := \beta^n(k/m) \tag{3.1}$$

$$c_m^n(k) := \beta^n(k/m + \tfrac{1}{2}). \tag{3.2}$$

According to our convention, the superscript n refers to the order of the splines and the subscript m to the spacing between the nodes (or step size). This latter parameter may also be interpreted as an expansion factor. Using (2.7), we derive the recursive equations

$$b_m^n(k) = \frac{\left(\frac{k}{m} + \frac{n+1}{2}\right) c_m^{n-1}(k) + \left(\frac{n+1}{2} - \frac{k}{m}\right) c_m^{n-1}(k-m)}{n} \tag{3.3}$$

$$c_m^n(k) = \frac{\left(\frac{k}{m} + \frac{n+2}{2}\right) b_m^{n-1}(k+m) + \left(\frac{n}{2} - \frac{k}{m}\right) b_m^{n-1}(k)}{n} \tag{3.4}$$

TABLE I
Z TRANSFORMS OF BASIC SYMMETRIC AND SHIFTED B-SPLINE KERNELS FOR $n = 0$ TO 5

n	$B_1^n(z)$	$C_1^n(z)$
0	1	1
1	1	$\frac{z+1}{2}$
2	$\frac{z+6+z^{-1}}{8}$	$\frac{z+1}{2}$
3	$\frac{z+4+z^{-1}}{6}$	$\left(\frac{z+1}{2}\right)\left(\frac{z+22+z^{-1}}{24}\right)$
4	$\frac{z^2+76z+230+76z^{-1}+z^{-2}}{384}$	$\left(\frac{z+1}{2}\right)\left(\frac{z+10+z^{-1}}{12}\right)$
5	$\frac{z^2+26z+66+26z^{-1}+z^{-2}}{120}$	$\left(\frac{z+1}{2}\right)\left(\frac{z^2+236z+1446+236z^{-1}+z^{-2}}{1920}\right)$

with the following starting conditions:

$$b_m^0(k) = \begin{cases} 1 & \text{for } -m/2 \le k \le m/2 \\ 0 & \text{otherwise} \end{cases}$$

$$c_m^0(k) = \begin{cases} 1 & \text{for } 1 - m \le k \le 0 \\ 0 & \text{otherwise.} \end{cases}$$

The discrete spline of order 0, b_m^0 (respectively, c_m^0), is a rectangular window of width m that is centered (respectively, shifted to the left) with respect to the origin when m is odd. This operator corresponds to a moving average filter of size m that can be implemented recursively using a standard update procedure (two operations per sample value) [28]. The values of $b_1^n(k)$ and $c_1^n(k)$ for $n = 0$ to 5 were determined iteratively from (3.3) and (3.4) and correspond to the coefficients of the z transforms that are given in Table I.

For discrete B-splines with upsampling integer m greater than 1, a convolution property that is somewhat similar to (2.9) can be established (cf. Appendix B):

a) m odd:

$$b_m^n(k) = \frac{1}{m^n} \underbrace{(b_m^0 * b_m^0 * \cdots * b_m^0)}_{n+1 \text{ times}} * b_1^n(k). \quad (3.5)$$

b) n odd and m even:

$$b_m^n(k) = \frac{1}{m^n} \delta_{(n+1)/2} * \underbrace{(b_m^0 * b_m^0 * \cdots * b_m^0)}_{n+1 \text{ times}} * b_1^n(k) \quad (3.6)$$

where $\delta_i(k)$ is the shift operator (e.g., $\delta_i * a(k) = a(k - i)$).

c) n even and m even:

$$b_m^n(k) = \frac{1}{m^n} \delta_{(n+2)/2} * \underbrace{(b_m^0 * b_m^0 * \cdots * b_m^0)}_{n+1 \text{ times}} * c_1^n(k). \quad (3.7)$$

These equations demonstrate that discrete B-splines of various widths can be constructed from the repeated convolution of simple moving average filters (b_m^0) and a correction kernel (b_1^n) or c_1^n (depending on the parity of n and m). Equations (3.5) and (3.6) are essentially the same and the only addition to our initial results [26] is the use of c_1^n instead of b_1^n when m and n are both even.

B. Transform Domain Characterization

Z-transform representations are especially suited for the design of recursive filtering algorithms for the direct and indirect B-spline transforms [26]. The up-sampled B-spline of order 0 is a rectangular window of length m and its z transform is given by

$$B_m^0(z) = \sum_{k=-[m/2]}^{[(m-1)/2]} = z^{[m/2]} \left(\frac{1 - z^{-m}}{1 - z^{-1}}\right) \quad (3.8)$$

where $[x]$ denotes the truncation of the variable x to the smaller integer. The basic ($m = 1$) sampled discrete B-spline is a symmetric function characterized by

$$B_1^n(z) = \sum_{k=-[n/2]}^{+[n/2]} b_1^n(k) z^{-k}. \quad (3.9)$$

As shown in Appendix B, the z transform of b_m^n when n and m are not both even is

$$B_m^n(z) = \frac{z^{i_0}}{m^n} \left(\frac{1 - z^{-m}}{1 - z^{-1}}\right)^{n+1} \sum_{k=-[n/2]}^{+[n/2]} b_1^n(k) z^{-k} \quad (3.10)$$

where $i_0 = (m - 1)(n + 1)/2$. Clearly, this result is in concordance with convolution properties (3.5) and (3.6). Furthermore, it is easily adapted to the condition n and m both even, by replacing b_1^n by c_1^n (cf. Appendix B).

From (3.10), we derive the Fourier transform of b_m^n for $(n + 1) \times (m - 1)$ even by replacing z by $e^{j2\pi f}$:

$$B_m^n(f) = \frac{1}{m^n} \left(b_1^n(0) + \sum_{k=1}^{[n/2]} 2b_1^n(k) \cos(2\pi f k)\right) \cdot \left(\frac{\sin(\pi m f)}{\sin(\pi f)}\right)^{n+1}. \quad (3.11)$$

C. Direct and Indirect Spline Transform [26]

The exact (or reversible) representation of a discrete signal $g(k)$ in the space of B-splines is obtained by imposing the interpolation condition:

$$\forall k \in Z, \qquad g(k) = g^n(x)|_{x=k} \tag{3.12}$$

where $g^n(x)$ is as in (2.5). As stated before, $g^n(x)$ is entirely specified by its expansion coefficients $\{y(k)\}$. The problem of finding these coefficients is sometimes referred to as the cardinal spline interpolation problem, and has been thoroughly investigated by Schoenberg in a very general mathematical framework [18], [19]. The solution of the system of equations (3.12) and (2.5) can be determined through the use of a linear operator which we refer to as the direct B-spline transform. We have shown that this transform can be obtained by convolution [26]:

$$\forall k \in Z, \qquad y(k) = (b_1^n)^{-1} * g(k) \tag{3.13}$$

where $(b_1^n)^{-1}$ is the impulse response of the direct B-spline filter of order n. The transfer function of this operator is

$$(b_1^n)^{-1}(k) \overset{z}{\leftrightarrow} B_1^n(z)^{-1} = \frac{1}{b_1^n(0) + \sum_{k=1}^{+[n/2]} b_1^n(k)[z^k + z^{-k}]}. \tag{3.14}$$

It may be decomposed as

$$B_1^n(z)^{-1} = \frac{z^{[n/2]}}{b_1^n([n/2]) \prod_{i=1}^{[n/2]} (z - z_i)(z - z_i^{-1})} \tag{3.15}$$

where $\{(z_i, z_i^{-1}): |z_i| \le 1, i = 1, \cdots, [n/2]\}$ are the roots of $B_1^n(z)$ which can be grouped in reciprocal pairs due to the symmetry of this kernel. The basic inverse transformation (indirect B-spline transform with $m = 1$) is the convolution between the coefficient sequence and the discrete B-spline kernel b_1^n:

$$\forall k \in Z, \qquad g(k) = b_1^n * y(k). \tag{3.16}$$

The indirect B-spline filter b_1^n is a symmetric finite impulse response (FIR) operator (cf. Table I). The direct B-spline filter is also symmetric but has an infinite impulse response (IIR). We also have demonstrated [29] that all direct B-spline filters are stable (i.e., that the poles of $B_1^n(z)^{-1}$ do not lie on the unit circle for any value of n). Computationally efficient implementations of these operators are considered in [26].

B-splines are primarily useful for interpolation. A signal may be reconstructed from its B-spline coefficients at a higher sampling rate through the use of an indirect transform with an up-sampling factor m:

$$\begin{aligned} g_m(k) &:= g^n(x)|_{x=k/m} = b_m^n * [y]_{\uparrow m}(k) \\ &= b_m^n * [(b_1^n)^{-1} * g]_{\uparrow m}(k). \end{aligned} \tag{3.17}$$

We do recall that b_m^n can be implemented from a cascade of simple operators by taking advantage of the convolution properties (3.5), (3.6), or (3.7). These operations are summarized in the block diagram in Fig. 1.

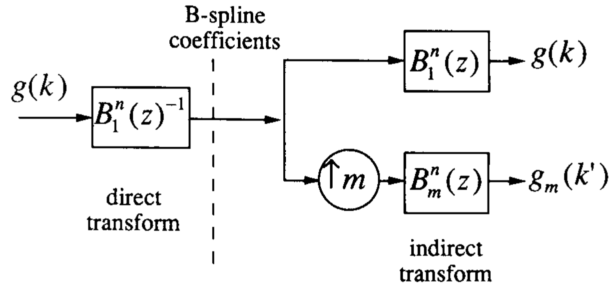

Fig. 1. Block diagram of a general B-spline interpolator with an optional expansion factor m.

D. Cardinal Spline Representation

An interesting way to look at B-spline interpolation is to express the interpolating function $g^n(x)$ in terms of the discrete function values themselves:

$$g^n(x) = \sum_{k=-\infty}^{+\infty} g(k)\eta^n(x - i) \tag{3.18}$$

where $\eta^n(x)$ is the *cardinal* spline of order n and represents the continuous impulse response of the polynomial spline interpolator; this function is also sometimes referred to as the *fundamental* spline of order n in the approximation theory literature. By using (2.5) and (3.12), we can show that

$$\eta^n(x) = \sum_{k=-\infty}^{+\infty} (b_1^n)^{-1}(k)\beta^n(x - k). \tag{3.19}$$

A standard decomposition of (3.15) into simple partial fractions allows us to express $(b_1^n)^{-1}$ in terms of simple symmetrical exponential responses. By making these calculations and substituting the results in (3.19), we find the new expression

$$\eta^n(x) = \sum_{k=-\infty}^{+\infty} \sum_{j=1}^{[n/2]} \alpha_j z_j^{|k|} \beta^n(x - k) \tag{3.20}$$

where the z_j are the $[n/2]$ smallest roots ($|z_i| < 1$) of $B_1^n(z)$ and where the weighting coefficients are

$$\alpha_j = \frac{1}{b_1^n(n/2)(z_j - z_j^{-1}) \prod_{\substack{i=1 \\ i \ne j}}^{[n/2]} (z_j + z_j^{-1} - z_i - z_i^{-1})}. \tag{3.21}$$

The Fourier transform of $\eta^n(x)$ is given by

$$H^n(f) = \int_{-\infty}^{+\infty} \eta^n(x)e^{-j2\pi xf}\,dx = \frac{(\sin(\pi f)/\pi f)^{n+1}}{b_1^n(0) + \sum_{k=1}^{[n/2]} 2b_1^n(k)\cos(2\pi fk)}. \tag{3.22}$$

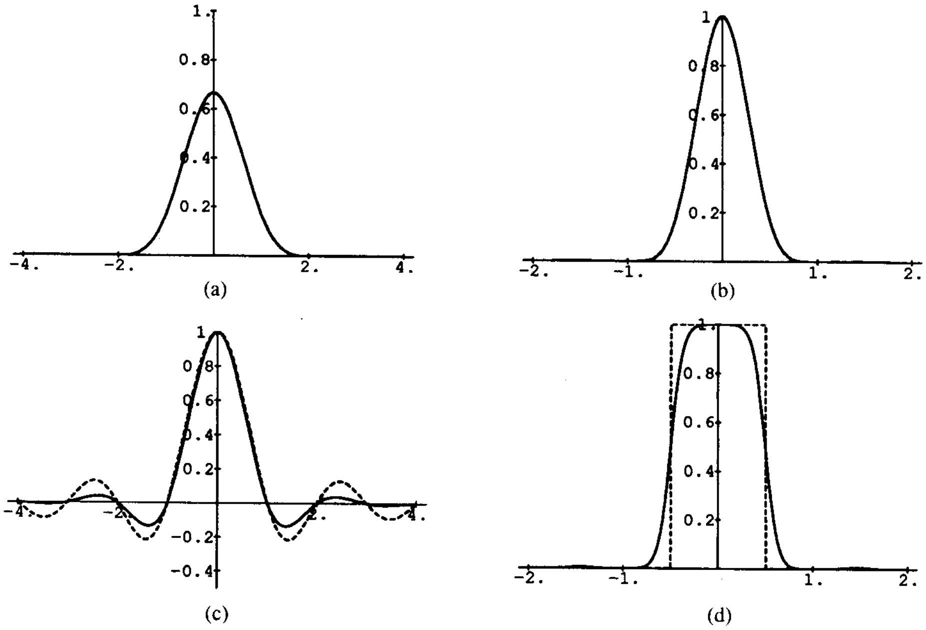

Fig. 2. Signals associated with the cubic spline interpolator. (a) Cubic B-spline, (b) Fourier transform of the cubic B-spline, (c) cardinal cubic spline, (d) Fourier transform of the cardinal cubic spline. The responses in dotted lines correspond to the ideal interpolator for band-limited signals.

As an example, the cubic cardinal spline and its Fourier transform are shown in Figs. 2(c) and (d); we used the coefficient values in Table I together with (3.20) and (3.22) to compute these graphs. These functions should not be confused with the cubic B-spline and its Fourier transform (Figs. 2(a) and (b)), which have been used incorrectly to characterize a cubic B-spline interpolator [1], [30], [31]; for a discussion see [26]. The essential interpolation property of the cardinal spline functions stems from the fact that they are equal to zero at all the nodes except the origin. As n increases, the cardinal spline becomes more and more nearly similar to a sinc function that corresponds to the ideal interpolator for a bandlimited function [29]. The B-spline functions, on the other hand, become more and more Gaussian-like, as a consequence of the central limit theorem.

IV. B-Spline Processing

The use of B-splines can go beyond simple interpolation. Their main advantage is to provide a convenient bridge between the discrete and continuous signal domains. It is thus possible to use concepts and mathematical techniques available for the study of continuous functions and to derive equivalent procedures for discrete signals, which may suggest new processing techniques. Some algorithms can be designed to operate directly onto the B-spline representation of a signal.

In this section, we first consider the problem of signal differentiation which is particularly relevant in the context of edge detection. A related topic is the design of discrete algorithms for the convolution of continuous signals. We also study the issue of obtaining B-spline approximations, which are useful for noise reduction and data compression. Such approximations can be obtained by imposing some smoothness constraints on the solution (smoothing splines), or by reducing the number of coefficients (least squares approximation).

Although most of the methods discussed here are standard in the spline literature [20], [21], the present derivations and the principles of computing these solutions using digital filters are new. When compared to the conventional procedure which is to formulate these tasks in terms of matrix equations and to solve these equations explicitly, the present filtering approach has a number of advantages. The first is a reduction of the number of operations, particularly when using recursive filters [26], [27], which have a complexity $O(N)$ where N is the number of data points. In contrast, the approach described in [22], which uses explicit matrix multiplications and inversions has a complexity of at least $O(N^2)$. The second is the simplicity of realization since all that is required is the implementation of a few digital filters.

These simplifications are only possible because we are dealing with the special case of equidistant data points. We are also intentionally avoiding the problem of boundary conditions by considering sequences of infinite length. Fortunately, this is not a major problem in practice since it is relatively easy to select boundary conditions so as to suppress border artifacts [27].

A. *Differentiation*

One of the simplest forms of B-spline processing is differentiation, a notion that is usually not well defined for discrete signals. The derivative of a signal is formally obtained by differentiating its continuous B-spline representation (2.5):

$$\frac{\partial g^n(x)}{\partial x} = \sum_{k=-\infty}^{+\infty} y(k) \frac{\partial \beta^n(x-k)}{\partial x}. \tag{4.1}$$

Substituting (2.8) in (4.1), we find that

$$\frac{\partial g^n(x)}{\partial x} = \sum_{k=-\infty}^{+\infty} (y(k) - y(k-1))\, \beta^{n-1}\left(x - k + \frac{1}{2}\right)$$
$$= \sum_{k=-\infty}^{+\infty} d^{(1)} * y(k) \beta^{n-1}\left(x - k + \frac{1}{2}\right) \quad (4.2)$$

where $d^{(1)}(k) = \delta_0(k) - \delta_0(k-1)$ is the first-order finite difference operator. By using the property

$$\frac{\partial \beta^n(x + \frac{1}{2})}{\partial x} = \beta^{n-1}(x+1) - \beta^{n-1}(x)$$

this formula is easily extended for higher order derivatives. In particular, the second derivative of the interpolating function $g^n(x)$ is given by

$$\frac{\partial^2 g^n(x)}{\partial x^2} = \sum_{k=-\infty}^{+\infty} (y(k+1) - 2y(k) + y(k-1))\beta^{n-2}(x-k)$$
$$= \sum_{k=-\infty}^{+\infty} d^{(2)} * y(k) \beta^{n-2}(x-k) \quad (4.3)$$

where $d^{(2)}(k) = \delta_0(k+1) - 2\delta_0(k) + \delta_0(k-1)$ is the second-order difference (or Laplacian) operator. It follows that differentiation in the B-spline domain is simply achieved by convolution with the appropriate finite difference operator. It is important to keep in mind that the resulting coefficients are the weights of B-splines of lower order. Therefore, the order of the indirect B-spline transform has to be decreased accordingly to map these results back into the initial discrete signal space. This principle is illustrated in Fig. 3 with the block diagram of a B-spline differentiator.

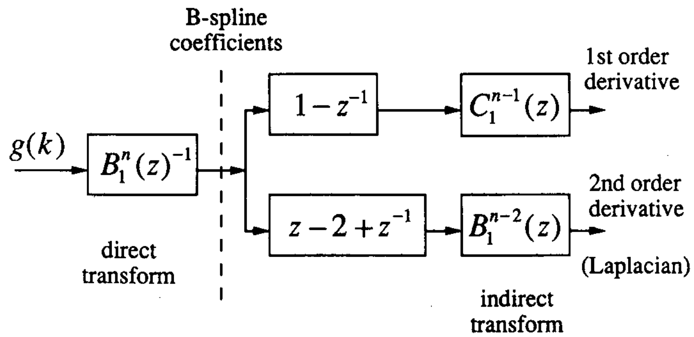

Fig. 3. Block diagram of a general B-spline differentiator (first- and second-order derivatives).

B. B-Spline Filtering

We propose the concept of B-spline filtering which is the process of applying a filtering operator to the continuous B-spline representation of a signal. When the operator is discrete, this procedure is rather trivial and does not seem to have any specific advantages: due to the linearity of all operations, one may as well apply the filter onto the discrete signal and avoid the unnecessary transformation step. Of greater interest is the case when the impulse response of the filter itself is represented by a B-spline of order p:

$$h^p(x) = \sum_{i=-\infty}^{+\infty} z(i) \beta^p(x - i). \quad (4.4)$$

The continuous convolution between $h^p(x)$ and $g^n(x)$ is

$$h^p * g^n(x) = \int_{-\infty}^{+\infty} h^p(x') g^n(x - x')\, dx'. \quad (4.5)$$

By substituting (4.4) and (2.5) in (4.5) and making the appropriate change of variables, we find that

$$h^p * g^n(x) = \sum_{k=-\infty}^{+\infty} z * y(k) \cdot \int_{-\infty}^{+\infty} \beta^p(x')\beta^n(x - k - x')\, dx'$$

which, due to the convolution properties of continuous B-splines, is also equivalent to

$$h^p * g^n(x) = \sum_{k=-\infty}^{+\infty} z * y(k) \beta^{p+n+1}(x-k). \quad (4.6)$$

It simply follows that $h^p * g^n(x)$ can be determined from the discrete convolution between the B-spline coefficients of the underlying signals. The only adjustment is an increase of the order of the B-spline representation of the filtered result, as illustrated by the block diagram in Fig. 4.

An interesting application of this result is the evaluation of the L_2-norm of the polynomial spline $g^n(x)$. Using (4.6), it is not difficult to show that

$$\| g^n(x) \|^2 = \langle y, b_1^{2n+1} * y \rangle = \sum_{k=-\infty}^{+\infty} \left(b_1^{2n+1} * y(k)\right) y(k) \quad (4.7)$$

which indicates that $\| g^n(x) \|^2$ can be computed from the inner product between the B-spline sequence $y(k)$ and the low-pass filtered signal $b_1^{2n+1} * y(k)$.

C. Smoothing Splines

For signals that are corrupted by noise, an exact B-spline interpolation is not necessarily the most adequate continuous signal approximation. Reinsh [33] and Schoenberg [32] have proposed the use of smoothing splines. Given a set of discrete signal values $\{g(k)\}$, the smoothing spline $\hat{g}(x)$ of order $2r-1$ is defined as the function that minimizes

$$\epsilon_S^2 = \sum_{k=-\infty}^{+\infty} (g(k) - \hat{g}(k))^2 + \lambda \int_{-\infty}^{+\infty} \left(\frac{\partial^r \hat{g}(x)}{\partial x^r}\right)^2 dx = \epsilon_A^2 + \lambda \epsilon_r^2 \quad (4.8)$$

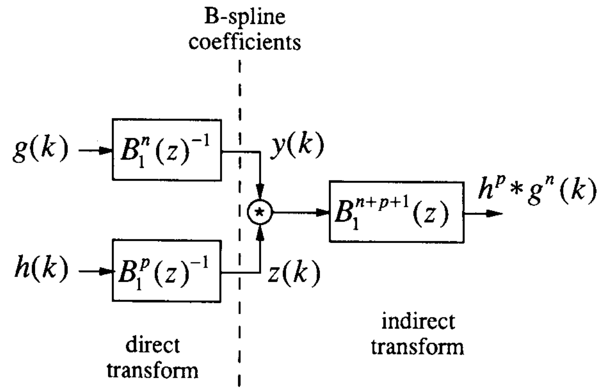

Fig. 4. Block diagram of a general B-spline convolver.

where λ is a given positive parameter. This method establishes a sort of compromise between the desire for an approximation that is reasonably close to the data and the requirement of a function that is sufficiently smooth. The choice of λ depends on which of these two conflicting goals is accorded the greater importance.

Schoenberg has considered the general case of nonequally spaced nodes and a finite number of data points [32]. He has demonstrated the important result that the function that minimizes (4.8) turns out to be a spline of order $n = 2r - 1$ with simple knots at the data points and some natural end conditions. Reinsh has worked out the explicit solution for the smoothing cubic B-spline [33].

Here, we will derive the general solution for an infinite sequence of data points with equally spaced nodes. This task first involves finding a simpler expression for the smoothing term in the criterion to be minimized. By generalizing (4.3) and assuming that r is even, we have

$$\epsilon_r^2 = \int_{-\infty}^{+\infty} \left(\frac{\partial^r \hat{g}(x)}{\partial x^r} \right)^2 dx$$
$$= \int_{-\infty}^{+\infty} \left(\sum_{i=-\infty}^{+\infty} d^{(r)} * y(i) \beta^{r-1}(x - i) \right)$$
$$\cdot \left(\sum_{j=-\infty}^{+\infty} d^{(r)} * y(j) \beta^{r-1}(x - j) \right) dx$$

where $d^{(r)}$ is the symmetric rth order difference operator (e.g., $D^{(r)}(z) = (z - 2 + z^{-1})^{r/2}$). We note that this expression is easily adapted for r odd by replacing $\beta^{r-1}(x)$ by $\beta^{r-1}(x + \frac{1}{2})$. We then make the change of variable $j = i - k$ and rearrange the sums and the integral:

$$\epsilon_r^2 = \sum_{i=-\infty}^{+\infty} d^{(r)} * y(i) \sum_{k=-\infty}^{+\infty} d^{(r)} * y(i - k)$$
$$\cdot \int_{-\infty}^{+\infty} \beta^{r-1}(x - i) \beta^{r-1}(x - i + k)\, dx.$$

The convolution property of the B-splines (2.9) implies that the result of the integral is simply $\beta^{2r-1}(-k) = \beta^{2r-1}(k)$. A simpler form of ϵ_r^2 is therefore

$$\epsilon_r^2 = \sum_{i=-\infty}^{+\infty} d^{(r)} * y(i)$$
$$\cdot \sum_{k=-\infty}^{+\infty} d^{(r)} * y(i - k) b_1^{2r-1}(k)$$
$$= \sum_{i=-\infty}^{+\infty} (d^{(r)} * y(i))(d^{(r)} * y * b_1^{2r-1}(i)). \quad (4.9)$$

Using a similar procedure, we can derive an identical formula for the case when n is odd. By using (3.16) and (4.9), the criterion to be minimized is expressed in terms of discrete convolutions

$$\epsilon_S^2 = \sum_{k=-\infty}^{+\infty} (g(k) - (b_1^{2r-1} * y(k)))^2$$
$$+ \lambda \sum_{i=-\infty}^{+\infty} (d^{(r)} * y(i))(d^{(r)} * y * b_1^{2r-1}(i)) \quad (4.10)$$

which, using our inner product notation, is also equivalent to

$$\epsilon_S^2 = \langle g, g \rangle - 2 \langle g, b_1^{2r-1} * y \rangle$$
$$+ \langle b_1^{2r-1} * y, b_1^{2r-1} * y \rangle$$
$$+ \lambda \langle d^{(r)} * y, d^{(r)} * y * b_1^{2r-1} \rangle. \quad (4.11)$$

The smoothing spline coefficients are found by setting to zero the derivative of this expression with respect to $y(k)$. By using rules (A-4) and (A-5) of the inner product calculus (Appendix A), we find that

$$(b_1^{2r-1})' * g(k) = (b_1^{2r-1})' * b_1^{2r-1} * y(k)$$
$$+ \lambda (d^{(r)\prime} * d^{(r)} * (b_1^{2r-1})' * y(k)) \quad (4.12)$$

which, in the z transform domain, is equivalent to

$$B_1^{2r-1}(z^{-1}) G(z) = B_1^{2r-1}(z^{-1}) B_1^{2r-1}(z) Y(z)$$
$$+ \lambda D^{(r)}(z) D^{(r)}(z^{-1}) B_1^{2r-1}(z^{-1}) Y(z).$$

Finally, by solving for $Y(z)$ and simplifying by $B_1^{2r-1}(z^{-1})$, we find that

$$Y(z) = S_\lambda^{2r-1}(z) G(z)$$
$$= \frac{1}{B_1^{2r-1}(z) + \lambda(-z + 2 - z^{-1})^r} G(z). \quad (4.13)$$

This expression clearly shows that the coefficients of the smoothing spline can be determined by digital filtering, as illustrated in Fig. 5. The transfer function of the smoothing spline filter $S_\lambda^{2r-1}(z)$ corresponds to a IIR filter, which is most efficiently implemented recursively as shown in [27]. We note that these operators are very similar to some of the R-filters derived by us earlier using regularization theory [34].

D. Least Squares Splines

The smoothing spline has as many coefficients (or degrees of freedom) as the initial signal. The noise reduction is achieved by imposing some smoothness constraints on the solution. De Boor has suggested instead using an approximation with fewer degrees of freedom and has introduced the use of least squares splines. He describes a general method for determining such solutions in the case of arbitrarily spaced data points that relies on the use of standard least squares approximation techniques [20].

Our approach which considers equally spaced nodes is somewhat less general but leads to some substantial computational simplifications. When dealing with discrete signals, this approximation method involves some form of decimation of the spline coefficients. This technique is conceptually similar to resampling a signal at a lower rate which requires the use of an antialiasing filter for the band-limited approximation of a signal with minimum error. In this sense, the present theory of least square B-spline fitting is an extension of the conventional sampling theorem for the subspace of piecewise polynomial functions of class $C^{n-1}(-\infty, +\infty)$ with equally spaced nodes (i.e., S_m^n). Clearly, this technique is a data reduction method but can also be considered as a noise reduction procedure.

The general form of a spline approximation $g_m^n(x) \in S_m^n$ with an up-sampling integer m is

$$g_m^n(x) = \sum_{i=-\infty}^{+\infty} y(i)\beta^n(x/m - i) \quad (4.14)$$

where the basis functions are enlarged by a factor of m and the number of B-spline coefficients is reduced in the same proportion. We wish to determine the least squares spline coefficients that minimize the approximation error:

$$\epsilon_m^2 = \sum_{k=-\infty}^{+\infty} (g(k) - [y]_{\uparrow m} * b_m^n(k))^2 \quad (4.15)$$

which, using our inner product notation, is also equivalent to

$$\epsilon_m^2 = \langle g, g\rangle - 2\langle g, [y]_{\uparrow m} * b_m^n\rangle + \langle [y]_{\uparrow m} * b_m^n, [y]_{\uparrow m} * b_m^n\rangle. \quad (4.16)$$

By setting to zero the derivative of (4.16) with respect to $y(k)$ (rules (A-6) and (A-7), Appendix A), we get the system of equations

$$[b_m^{n\prime} * b_m^n]_{\downarrow m} * y(k) = [b_m^n * g]_{\downarrow m}(k), \quad k \in Z. \quad (4.17)$$

Provided that the inverse of the operator $[b_m^n * b_m^n]_{\downarrow m}$ exists, the expression can be solved by inverse filtering:

$$y(k) = s_m^n * [b_m^n * g]_{\downarrow m}(k), \quad k \in Z \quad (4.18)$$

where the postfilter $s_m^n(k)$ is defined by

$$s_m^n(k) := \left([b_m^n * b_m^n]_{\downarrow m}\right)^{-1}(k). \quad (4.19)$$

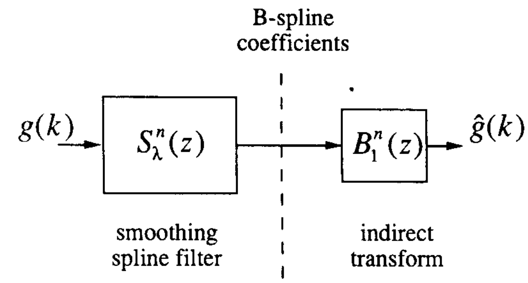

Fig. 5. Block diagram of a filter-based system for signal approximation using smoothing splines.

Using (2.4), we get an expression for the transfer function

$$S_m^n(z) = \frac{1}{\frac{1}{m}\sum_{k=0}^{m-1} B_m^n([ze^{j2\pi k}]^{1/m})^2}. \quad (4.20)$$

These results suggest a simple three step procedure for the determination of the least squares B-spline coefficients: i) a prefiltering with a B-spline kernel of width $m(b_m^n)$, ii) a decimation by a factor of m, and iii) a postfiltering with s_m^n; this algorithm is illustrated in Fig. 6. Clearly, $[b_m^n * b_m^n]_{\downarrow m}$ is a FIR filter which implies that the inverse filter has an infinite impulse response. The issue of the stability of these filters as well as their efficient implementation in some cases of interest is treated in the companion paper [27].

It is also conceivable to perform the decimation at the very end of the procedure. To achieve this, we have to up-sample the impulse response of the postfilter by a factor of m. By defining

$$\mathring{b}_m^n = [s_m^n]_{\uparrow m} * b_m^n$$

we can rewrite (4.18) as

$$y(k) = [\mathring{b}_m^n * g]_{\downarrow m}(k), \quad k \in Z \quad (4.21)$$

where the global transfer function of the least squares spline prefilter is given by

$$\mathring{B}_m^n(z) = \frac{B_m^n(z)}{\frac{1}{m}\sum_{k=0}^{m-1} B_m^n(ze^{j2\pi k/m})^2}. \quad (4.22)$$

The prefilter $\mathring{B}_m^n(z)$ that is applied prior to decimation can be interpreted as a pseudoinverse of the interpolation operator $B_m^n(z)$. Its role is in all point similar to that of an antialiasing filter used in conventional sampling theory.

E. Extensions to Higher Dimensions

Although all our results were derived for the one-dimensional case, they are directly applicable to higher dimensions through the use of tensor product splines [21]. The corresponding basis functions are obtained from the product of one-dimensional splines defined for each index variable. Since all basis functions are separable, the corresponding linear direct and indirect transformations are

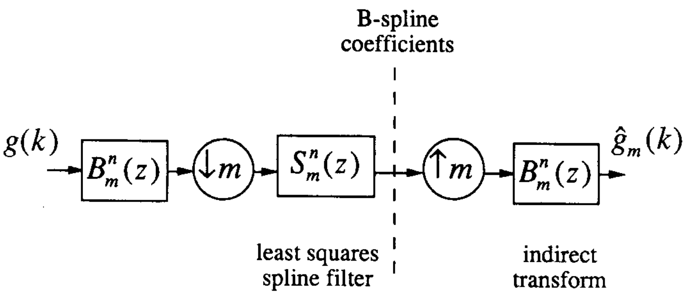

Fig. 6. Block diagram of a filter-based system for signal approximation using least squares B-splines.

also separable [1]. This implies that the spline coefficients can be determined by successive one-dimensional direct B-spline filtering along the coordinates. The same strategy is also applicable for signal reconstruction or interpolation by indirect spline filtering.

The results on B-spline differentiation are straightforward to extend for the evaluation of partial derivatives, gradients or Laplacians in higher dimensions. The corresponding convolution masks, which are separable, are simply obtained through the tensor product of basic one-dimensional difference operators. The only delicate step is to apply the proper order reduction for the indirect B-spline transforms along the different index variables when these results are mapped back into the initial signal domain. The difference operators and indirect B-spline kernels can be combined to obtain templates for image gradients and Laplacians [27].

The least squares B-spline approximation techniques are also directly applicable by sequentially processing the various dimensions of the data with one-dimensional operators such as the one derived in Section IV-D. For a simple proof that separability also applies for linear least square approximation in two dimensions, we refer to [4, appendix B].

The extension of smoothing B-spline approximation to higher dimensional signals is not quite as straightforward. In order to preserve separability, it is necessary to extend the smoothing functional in (4.8) by introducing some appropriate cross terms. In particular, by applying a technique similar to the one used in Section IV-C, we can show that the problem of finding a bidimensional spline that minimizes the criterion

$$\epsilon_S^2 = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} (g(k, l) - \hat{g}(k, l))^2 + \lambda \int_{-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} \left(\frac{\partial^r \hat{g}(x, l)}{\partial x^r}\right)^2 dx + \mu \sum_{k=-\infty}^{+\infty} \int_{-\infty}^{+\infty} \left(\frac{\partial^r \hat{g}(k, y)}{\partial y^r}\right)^2 dy + \lambda\mu \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \left(\frac{\partial^r \hat{g}(x, y)}{\partial x^r \partial y^r}\right)^2 dx\, dy \quad (4.23)$$

can be determined through the successive application along the row and columns of one-dimensional operators of the type defined by (4.13) with horizontal and vertical regularization parameters λ and μ, respectively. If the last term of this expression is not included, the corresponding operator is not separable anymore. In addition, it is not clear to us at this stage that we have the same fundamental result as for the one dimensional case [32]. In other words, we do not know if among all possible functions, the one that minimizes (4.23) is a separable two-dimensional spline of order $2r - 1$ with knots at the grid points.

V. Conclusion

In this paper, we have considered the use of continuous B-spline representations for signal processing applications such as interpolation, differentiation, filtering, noise reduction, and data compression. B-spline representations can be useful in a variety of problems that are best formulated in a continuous rather than a discrete framework. In this respect, it appears that computational tasks such as differentiation, integration, or the search for extrema are especially simple to perform in the transformed B-spline domain. Some of the most obvious applications are the problem of estimating higher order derivatives from a noisy signal and edge detection in image processing.

The B-spline coefficients are obtained through a linear transformation, which unlike other commonly used transforms (Fourier, Karhunen–Loève, sine, cosine, etc.), is translation invariant and can be implemented efficiently by linear filtering. The same property also applies for the indirect B-spline transform as well as for the evaluation of approximating representations using smoothing or least squares splines. In this study, we have fully characterized the filters associated with these operations by explicitly evaluating their transfer functions for splines of any order. We have also considered the extension of such operators for higher dimensional signals such as digital images.

Image processing applications of all these techniques will be presented and discussed in the companion paper [27]. This report will also focus on the issue of efficient implementation using recursive filters.

Appendix A
Inner Product Calculus

Let $y(k)$, $z(k)$, and $h(k)$ be discrete signals in l_2. The inner product between such sequences is defined by (2.1). Using standard calculus, it is relatively straightforward to establish the basic differentiation rules (A-1) and (A-2). The additional equations that we are giving here are all derived from those two fundamental rules.

a) *Linear and quadratic forms:*

$$\frac{\partial \langle y, z \rangle}{\partial y(k)} = z(k) \quad \text{(A-1)}$$

$$\frac{\partial \langle h * y, y \rangle}{\partial y(k)} = h * y(k) + h' * y(k) \quad \text{(A-2)}$$

$$\frac{\partial \langle y, y \rangle}{\partial y(k)} = 2y(k) \quad \text{(A-3)}$$

Proof of (A-3): By setting $h(k) = \delta(k)$ in (A-2), we directly get (A-3).

b) *Convolutions:*

$$\frac{\partial \langle h * y, z \rangle}{\partial y(h)} = h' * z(k) \quad \text{(A-4)}$$

$$\frac{\partial \langle h * y, h * y \rangle}{\partial y(k)} = 2h' * h * y(k) = 2\varphi_{hh} * y(k) \quad \text{(A-5)}$$

where $\varphi_{hh}(k) = h' * h(k)$ is the autocorrelation function of $h(k)$.

Proof of (A-4): $\langle h * y, z \rangle = \langle y, h' * z \rangle$. Equation (A-4) then directly follows from (A-1).

Proof of (A-5): $\langle h * y, h * y \rangle = \langle h' * h * y, y \rangle$. Equation (A-5) then follows from (A-2).

c) *Up-sampling and decimation:*

$$\frac{\partial \langle h * [y]_{\uparrow m}, z \rangle}{\partial y(k')} = [h' * z]_{\downarrow m}(k') \quad \text{(A-6)}$$

$$\frac{\partial \langle h * [y]_{\uparrow m}, h * [y]_{\uparrow m} \rangle}{\partial y(k')} = 2[h' * h]_{\downarrow m} * y(k') = 2[\varphi_{hh}]_{\downarrow m} * y(k'). \quad \text{(A-7)}$$

Proof of (A-6): $\langle h * [y]_{\uparrow m}, z \rangle = \langle h' * z, [y]_{\uparrow m} \rangle$. We then apply a restriction on $y(k')$ which is obtained by decimation: $\langle [h' * z]_{\uparrow m}, y \rangle$. Equation (A-6) then directly follows from (A-1).

Proof of (A-7): $\langle h * [y]_{\uparrow m}, h * [y]_{\uparrow m} \rangle = \langle h' * h * [y]_{\uparrow m}, [y]_{\uparrow m} \rangle$. We then apply a restriction on $y(k')$: $\langle [(h' * h * [y]_{\uparrow m}]_{\downarrow m}, y \rangle = \langle [h' * h]_{\downarrow m} * y, y \rangle$ and use (A-2).

Appendix B
Discrete B-Spline Convolution Properties

Our derivation of the convolution properties of discrete B-splines is based on the evaluation of the z-transforms of the two following sequences:

$$u_m^n(k) = \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \left(\frac{k}{m} - j\right)^n \mu\left(\frac{k}{m} - j\right) = \beta^n\left(\frac{k}{m} - \frac{n+1}{2}\right) \quad \text{(B-1)}$$

$$v_m^n(k) = \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \left(\frac{k+\frac{1}{2}}{m} - j\right)^n \cdot \mu\left(\frac{k+\frac{1}{2}}{m} - j\right) = \beta^n\left(\frac{k+\frac{1}{2}}{m} - \frac{n+1}{2}\right). \quad \text{(B-2)}$$

We have shown previously that the z transform of u_m^n is [26]

$$U_m^n(z) = \frac{1}{m^n} U_1^n(z)(U_m^0(z))^{n+1}. \quad \text{(B-3)}$$

We will use a similar technique to show that the same result applies for v_m^n. We first need to determine the transform of power series of the form: $\{(k + \frac{1}{2})^n; k = 0, \cdots, +\infty\}$. Let

$$P_\delta^n(z) = \sum_{k=-\infty}^{+\infty} (k - \delta)^n z^{-k} \mu(k - \delta) \quad \text{(B-4)}$$

where $\delta > -1$ is an offset parameter. Since

$$\frac{\partial(z^\delta P_\delta^n(z))}{\partial z} = \sum_{k=-\infty}^{+\infty} -(k-\delta)^{n+1} z^{-k+\delta-1} \mu(k-\delta) = -z^{\delta-1} \sum_{k=-\infty}^{+\infty} (k-\delta)^{n+1} z^{-k} \mu(k-\delta)$$

we have the recurrence equation

$$P_\delta^{n+1}(z) = -z^{-\delta+1} \frac{\partial(z^\delta P_\delta^n(z))}{\partial z}. \quad \text{(B-5)}$$

By using the fact that the z transform of $\mu(k)$ is

$$P_{-1/2}^0(z) = \frac{1}{1 - z^{-1}} \quad \text{(B-6)}$$

we are able to evaluate the transforms of all subsequent power series recursively. More importantly, we can show that the general term has the form:

$$P_{-1/2}^n(z) = \frac{A^n(z)}{(1 - z^{-1})^{n+1}} \quad \text{(B-7)}$$

where the numerator $A^n(z)$ is some polynomial in z^{-1}. The z transform of (B-2) is found by making use of the shift theorem and substituting the expression for $P_{-1/2}^n(z)$:

$$V_m^n(z) = \frac{1}{m^n} \sum_{j=0}^{n+1} \frac{1}{n!} \binom{n+1}{j} (-1)^j z^{-jm} \frac{A^n(z)}{(1 - z^{-1})^{n+1}}.$$

By noticing that $(-1)^j z^{-jm}$ is also equal to $(-z^{-m})^j$ and recalling that

$$(x + 1)^m = \sum_{j=0}^{m} \binom{m}{j} x^j$$

we finally get

$$V_m^n(z) = \frac{1}{m^n} \frac{A^n(z)}{n!} \left(\frac{1 - z^{-m}}{1 - z^{-1}}\right)^{n+1}. \quad \text{(B-8)}$$

By evaluating (B-8) for $m = 1$ and solving for $A^n(z)$, we find that

$$A^n(z) = V_1^n(z) n!. \quad \text{(B-9)}$$

We also note that v_m^0 is a rectangular window of size m and that its z transform is

$$V_m^0 = \frac{1 - z^{-m}}{1 - z^{-1}}. \quad \text{(B-10)}$$

By substituting (B-9) and (B-10) in (B-8), we finally get

$$V_m^n(z) = \frac{1}{m^n} V_1^n(z)(V_m^0(z))^{n+1}. \qquad \text{(B-11)}$$

The convolution equations (3.5)–(3.7) follow directly from (B-3) and (B-11) by noticing that we have the following equivalences:

a) *m even*

$$b_m^n(k) = u_m^n\left(k + \frac{(n+1)m}{2}\right)$$

$$c_m^n(k) = u_m^n\left(k + \frac{(n+2)m}{2}\right).$$

b) *n odd and m odd*

$$b_m^n(k) = u_m^n\left(k + \frac{(n+1)m}{2}\right)$$

$$c_m^n(k) = v_m^n\left(k + \frac{(n+2)m - 1}{2}\right).$$

c) *n even and m odd*

$$b_m^n(k) = v_m^n\left(k + \frac{(n+1)m - 1}{2}\right)$$

$$c_m^n(k) = u_m^n\left(k + \frac{(n+2)m}{2}\right).$$

References

[1] W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.
[2] B. K. P. Horn, *Robot Vision*. New York: McGraw-Hill, 1986.
[3] R. M. Haralick and L. Watson, "A facet model for image data," *Comput. Graphics Image Processing*, vol. 15, pp. 113–129, 1981.
[4] M. Eden, M. Unser, and R. Leonardi, "Polynomial representation of pictures," *Signal Processing*, vol. 10, pp. 385–393, July 1986.
[5] V. Torre and T. A. Poggio, "On edge detection," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp. 147–163, Mar. 1986.
[6] L. G. Roberts, "Machine perception of three-dimensional solids," in *Optical and Electrooptical Information Processing*, J. T. Tripett *et al.*, Ed. Cambridge, MA: M.I.T. Press, 1965, pp. 159–197.
[7] J. Kittler, "On the accuracy of the Sobel edge detector," *Image Vision Comput.*, vol. 1, pp. 37–42, Feb. 1983.
[8] M. Hueckel, "A local visual operator which recognizes edges and lines," *J. Ass. Comput. Mach.*, vol. 20, pp. 634–647, 1973.
[9] R. M. Haralick, "Digital step edges from zero crossing of second directional derivatives," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-6, pp. 58–68, Jan. 1984.
[10] T. Poggio, H. Voorhees, and A. Yuille, "Regularizing edge detection," Artificial Intelligence Lab. Memo 776, Massachusetts Institute Technol., Cambridge, MA, 1984.
[11] T. Poggio, H. Voorhees, and A. Yuille, "A regularized solution to edge detection," *J. Complexity*, vol. 4, pp. 106–123, 1988.
[12] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Roy. Soc. London*, vol. B 207, pp. 187–217, 1980.
[13] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp. 679–697, 1986.
[14] M. Hashimoto and J. Sklansky, "Multiple-order derivatives for detecting local image characteristics," *Comput. Vision, Graph., Image Processing*, vol. 39, pp. 28–55, 1987.
[15] T. Poggio, V. Torre, and C. Koch, "Computational vision and regularization theory," *Nature*, vol. 317, pp. 314–319, 1985.
[16] A. P. Witkin, "Scale-space filtering," in *Proc. 4th Int. Joint Conf. Artificial Intelligence*, 1983, pp. 1019–1022.
[17] D. Terzopoulos, "Multilevel computational processes for visual surface reconstruction," *Comput. Vision, Graph., Image Processing*, vol. 24, pp. 52–96, 1983.
[18] I. J. Schoenberg, "Contribution to the problem of approximation of equidistant data by analytic functions," *Quart. Appl. Math.*, vol. 4, pp. 45–99, 112–141, 1946.
[19] I. J. Schoenberg, "Cardinal interpolation and spline functions," *J. Approximation Theory*, vol. 2, pp. 167–206, 1969.
[20] C. de Boor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.
[21] L. L. Schumaker, *Spline Functions: Basic Theory*. New York: Wiley, 1981.
[22] H. S. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 508–517, 1978.
[23] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, 1989.
[24] K. Toraichi, S. Yang, M. Kamada, and R. Mori, "Two-dimensional spline interpolation for image reconstruction," *Patt. Recognition*, vol. 21, pp. 275–284, 1988.
[25] M. Bertero, T. A. Poggio, and V. Torre, "Ill-posed problems in early vision," *Proc. IEEE*, vol. 76, pp. 869–889, Aug. 1988.
[26] M. Unser, A. Aldroubi, and M. Eden, "Fast B-spline transforms for continuous image representation and interpolation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, pp. 277–285, Mar. 1991.
[27] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part II–Efficient design and applications," *IEEE Trans. Signal Processing*, this issue, pp. 834–848.
[28] M. Unser, "Recursion in short time signal analysis," *Signal Processing*, vol. 5, pp. 229–240, May 1983.
[29] A. Aldroubi, M. Unser, and M. Eden, "Cardinal spline filters: Stability and convergence to the ideal sinc interpolator," *Signal Processing*, vol. 28, no. 2, pp. 127–138, Aug. 1992.
[30] J. A. Parker, R. V. Kenyon, and D. E. Troxel, "Comparison of interpolating methods for image resampling," *IEEE Trans. Med. Imaging*, vol. MI-2, pp. 31–39, 1983.
[31] P. V. Sankar and L. A. Ferrari, "Simple algorithms and architectures for B-spline interpolation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, pp. 271–276, 1988.
[32] I. J. Schoenberg, "Spline functions and the problem of graduation," *Proc. Nat. Acad. Sci.*, vol. 52, pp. 947–950, 1964.
[33] C. H. Reinsh, "Smoothing by spline functions," *Numer. Math.*, vol. 10, pp. 177–183, 1967.
[34] M. Unser, A. Aldroubi, and M. Eden, "Recursive regularization filters: Design, properties and applications," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, pp. 272–277, Mar. 1991.

Michael Unser (M'88) was born in Zug, Switzerland, on April 9, 1958. He received the M.S. (with honors) and Ph.D. degrees in electrical engineering in 1981 and 1984, respectively, from the Swiss Federal Institute of Technology, Lausanne, Switzerland.

He is currently a Visiting Scientist with the Biomedical Engineering and Instrumentation Program, National Institutes of Health, Bethesda, MD, which he joined in 1985. He has also been affiliated with the French National Institutes of Health and Biomedical Research (INSERM) since April 1988. His research interests include the application of image processing and pattern recognition techniques to various biomedical problems, multiresolution algorithms, wavelet transforms, and the use of splines in signal processing.

Dr. Unser is an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING.