CHAPTER 1

Local B-spline Multiresolution with Examples in Iris Synthesis and Volumetric Rendering

Faramarz F. Samavati[1], Richard H. Bartels[2] and Luke Olsen [1]

[1]*Department of Computer Science, University of Calgary, Calgary, Alberta T2N 1N4, Canada. email:samavati@cpsc.ucalgary.ca*
[2]*School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. email: rhbartel@uwaterloo.ca*

Multiresolution has been extensively used in many areas of computer science, including biometrics. We introduce local multiresolution filters for quadratic and cubic B-splines that satisfy the first and the second level of smoothness respectively. For constructing these filters, we use a reverse subdivision method. We also show how to use and extend these filters for tensor-product surfaces, and 2D/3D images. For some types of data, such as curves and surfaces, boundary interpolation is strongly desired. To maintain this condition, we introduce extraordinary filters for boundaries. For images and other cases in which interpolating the boundaries is not required or even desired, we need a particular arrangement to be able to apply regular filters. As a solution, we propose a technique based on symmetric extension. Practical issues for efficient implementation of multiresolution are discussed. Finally, we discuss some example applications in biometrics, including iris synthesis and volumetric data visualization.

1. Introduction

Multiresolution provides a tool for decomposing data into a hierarchy of components with different scales or resolutions. This hierarchy can be used for noise removal, compression, synthesizing and recognition of the objects. An efficient and economical multiresolution is associated with wavelets. Therefore, this kind of multiresolution has been employed in many areas such as image processing[13], biomedical[21] and computer graphics[18].

Noise removal, data synthesis and feature recognition are common prob-

lems in biometrics. Multiresolution techniques have been employed in iris identification[7,9,23], iris synthesis[22], feature extraction from fingerprints and irises[3,10,14], and for detecting faces in images[24].

To date, Haar wavelets have primarily been used in biometric applications due to their simplicity. Haar wavelets are based on zero-degree B-splines. Unfortunately, zero-degree B-spline functions and their wavelet functions are not even continuous and consequently they are not well suited when we need a "smooth multiresolution representation" for data.

In this chapter, we introduce multiresolution representations for quadratic and cubic B-spline that satisfy the first and the second level of smoothness respectively. To keep the simplicity and efficiency of the resulting techniques, we build and describe *local multiresolution filters* from a condition for *biorthogonal wavelets* using the general approach of *reverse subdivision* . Consequently, the purpose of this work is not only describing the construction method (Secs. 3 –5), but also discussing practical implementation issues (Secs. 6–8).

2. Wavelets and Multiresolution Background

Multiresolution operations are specified by a set of filter matrices $\mathbf{A}^k$, $\mathbf{B}^k$, $\mathbf{P}^k$ and $\mathbf{Q}^k$. Consider a given discrete signal C^k, expressed as a column vector of samples. A lower-resolution sample vector C^{k-1} is created by a down-sampling filter on C^k. This process can be expressed as a matrix equation

$$C^{k-1} = \mathbf{A}^k C^k \ .$$

The *details* D^{k-1} lost through the down-sampling are captured using $\mathbf{B}^k$

$$D^{k-1} = \mathbf{B}^k C^k \ .$$

The pair of matrices $\mathbf{A}^k$ and $\mathbf{B}^k$ are called *analysis filters* and the process of splitting a signal C^k into C^{k-1} and D^{k-1} is called *decomposition*. Recovering the original signal C^k is called *reconstruction*. It involves refinement of the low-resolution sample C^{k-1} and details D^{k-1} using the *synthesis filters* $\mathbf{P}^k$ and $\mathbf{Q}^k$, which reverse the operations of $\mathbf{A}^k$ and $\mathbf{B}^k$

$$C^k = \mathbf{P}^k C^{k-1} + \mathbf{Q}^k D^{k-1} \ .$$

The matrices $\mathbf{A}^k$, $\mathbf{B}^k$, $\mathbf{P}^k$ and $\mathbf{Q}^k$ form the core of the multiresolution approach, and the efficiency of the resulting techniques depends on the structure of these matrices. For an efficient and useful representation, the following properties are desired:

- All matrices should be banded, with repetitive row/column entries.
- C^{k-1} is a good approximation for C^k.
- The storage requirement for storing C^{k-1} and D^{k-1} is not more than that of C^k.
- The time required to decompose and reconstruct the signal is linearly dependent on the size of C^k.

Decomposition and reconstruction operations take place in some underlying function spaces $\mathcal{V}^{k-1} \subset \mathcal{V}^k$ wherein C^k defines some function $f^k = \sum_i c_i^k \phi_i^k$ in the large space, C^{k-1} defines an approximation $f^{k-1} = \sum_i j c_j^{k-1} \phi_j^{k-1}$ to that function in the smaller space, and D^{k-1} defines the difference $g^{k-1} = \sum_i j d_j^{k-1} \psi_j^{k-1}$ in the complement space $\mathcal{V}^k \setminus \mathcal{V}^{k-1}$. The basis functions ψ_i^{k-1} are conventionally called *wavelets* and the ϕ_j are called *scale functions*.

Wavelet systems are usually classified according to the relationship between the wavelets and the scaling functions. Stollnitz et al. provide an excellent overview of wavelet classifications[18], which we summarize here.

Orthogonal wavelets. An orthogonal wavelet system is one in which "the scaling functions are orthogonal to one another, the wavelets are orthogonal to one another, and each of the wavelets is orthogonal to every coarser scaling function." In such a setting, the determination of the multiresolution filters is quite easy. Unfortunately, orthogonality is difficult to satisfy for all but the most trivial scaling functions.

Semiorthogonal wavelets. Semiorthogonal wavelets relax the orthogonality conditions, only requiring that each wavelet function is orthogonal to all coarser scaling functions. By relaxing the constraints on the wavelets, it is easier to derive a $\mathbf{Q}^k$ filter (note that there is no unique choice of $\mathbf{Q}^k$, but there are some choices that are better than others). The drawback of semiorthogonal wavelets is that while $\mathbf{P}^k$ and $\mathbf{Q}^k$ will be sparse matrices (meaning that reconstruction can be done in linear time), the decomposition filters $\mathbf{A}^k$ and $\mathbf{B}^k$ offer no such guarantee. It often turns out that the decomposition filters are full matrices while these matrices are very simple and banded in the case of Haar wavelets[18].

Biorthogonal wavelets. Finally there are biorthogonal wavelets, which have many of the properties of semiorthogonal wavelets but enforce no orthogonality conditions. The only condition in a biorthogonal setting is that $\left[\mathbf{P}^k | \mathbf{Q}^k\right]$ is invertible, which implies that the decomposition filters $\mathbf{A}^k$

and $\mathbf{B}^k$ exist such that

$$\begin{bmatrix} \mathbf{A}^k \\ \mathbf{B}^k \end{bmatrix} \begin{bmatrix} \mathbf{P}^k & \mathbf{Q}^k \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} . \tag{2.1}$$

Simply having a matrix $\mathbf{A}^k$ that satisfies (2.1) does not necessarily produce coarse data C^{k-1} that is a good approximation of C^k. Consequently, this condition should also be taken to account in our construction of $\mathbf{A^k}$.

Wavelet transform. We can repeatedly decompose a signal C^k to $C^\ell, C^{\ell+1}, \ldots, C^{k-1}$ and details $D^\ell, D^{\ell+1}, \ldots, D^{k-1}$ where $\ell < k$. The original signal C^k can be recovered from the sequence $C^\ell, D^\ell, D^{\ell+1}, \ldots, D^{k-1}$; this sequence is known as a *wavelet transform*. Based on the properties mentioned above the total size of the transform $C^\ell, D^\ell, D^{\ell+1}, \ldots, D^{k-1}$ is the same as that of the original signal C^k. In addition, the time required to transform C^k to $C^\ell, D^\ell, D^{\ell+1}, \ldots, D^{k-1}$, and vice versa, is a linear function of the size C^k.

Details interpretation. If C^k represents a high-resolution approximation of a curve, then C^ℓ is a very coarse approximation of the curve showing the main outline, and D^i consist of vectors which perturb the curve into its original path. As Fig. 8(b) demonstrates, if we eliminate D^i, the reconstructed curve becomes much smoother but without any of the curve's individual finer structure. In fact, D^i can be considered as *characteristic* of the curves. It is possible to apply D^i to a new coarse curve to obtain a new curve but with the same character (see Fig. 8(d)). Consequently, D^i at different levels are important features for synthesizing techniques.

B-spline multiresolution. B-splines are often chosen as scaling functions[6]. The first order (zero degree) B-splines form a set of step functions and Haar functions are their associated wavelets[18,19]. The resulting matrix filters are very simple and efficient. However, these scaling functions and wavelets are non-continuous. This is a problem when we have discrete data that is a sample of smooth signals and objects. Higher order B-splines and their wavelets can be considered for smooth signals[6,8,16].

A common knot arrangement, the standard arrangement, for B-splines of order k is to have knots of single multiplicity uniformly spaced everywhere except at the ends of the domain where knots have multiplicity k[1,15]; this arrangement produces endpoint-interpolation. Conventionally, B-spline wavelets are constructed with a goal of semiorthogonality, which results in full analysis matrices.

An alternative approach to generating multiresolution matrices is *reverse subdivision*, originally introduced by Bartels and Samavati[2]. Based

on this approach, it is possible to obtain banded matrices for biorthogonal B-spline wavelets whose bands are narrower than the ones conventionally produced. In this work, we construct and report multiresolution filter matrices for quadratic and cubic B-splines, which are important practical cases. Because of the similarity in the constructions, we just describe the process in detail for cubic B-splines.

Notation. For clarity of notation, the remainder of the chapter will forgo the superscript k for denoting the k-th level of subdivision. Let $C = C^k$ and $F = C^{k+1}$, such that

$$C = \{c_1, \ldots, c_n\} \ ,$$
$$F = \{f_1, \ldots, f_m\} \ .$$

Further, let $\mathbf{P} = \mathbf{P}^k$, $\mathbf{Q} = \mathbf{Q}^k$, $\mathbf{A} = \mathbf{A}^k$, and $\mathbf{A} = \mathbf{A}^k$. These matrices are assumed to be of the proper size so that the following equations hold

$$C = \mathbf{A}F \tag{2.2}$$
$$D = \mathbf{B}F \tag{2.3}$$
$$F = \mathbf{P}C + \mathbf{Q}D \ . \tag{2.4}$$

3. Review of Construction

We construct multiresolution of B-splines by reversing their *subdivision* schemes. In general, a subdivision process takes some coarse data as input. To this is applied a set of rules that replace the coarse data with a finer (smoother) representation. The set of rules could again be applied to this finer data. In the limit of repeated application, the rules yield data with provable continuity. The standard midpoint knot insertion process results in a subdivision scheme for B-splines[1,15].

Though subdivision is usually discussed in the context of curves and surfaces, it is a general process that can operate on any data type upon which linear combinations are defined. Thus we will consider subdivision to operate on some "coarse set" C of samples, and the process of subdivision is expressed in matrix form as $F = \mathbf{P}C$, whereby C is converted into a larger "fine set" F by the subdivision matrix $\mathbf{P}$.

The construction of multiresolution assumes that F is not the result of subdivision; that is, $F \neq \mathbf{P}C$ for any vector C. In this case we wish to find a vector C so that $F \approx \widetilde{F} \equiv \mathbf{P}\ C$, so that the residuals $F - \widetilde{F}$ are small, and so that complete information about these residuals can be stored in the space used for $\{f\} \setminus \{c\}$ (or one of an equivalent size). Informally, this

describes the features of a biorthogonal multiresolution built upon $\mathbf{P}$, which is the goal of the construction.

If the components of C and F are arranged in sequence, the subdivision matrices will be banded, repetitive, and slanted. That is, each column j of $\mathbf{P}$ has only a finite number of nonzero entries, located from some row r_j through a lower row $r_j + \ell$; these nonzero numbers appear in all columns save for a few exceptions (corresponding to the boundaries of the data $\{c\}$), and the entries of each succeeding or preceding column are shifted down or up by some fixed number of rows (which is determined by the *dilation scale* of the nested function spaces underlying the subdivision).

$$\mathbf{P} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{matrix} \\ \\ c_{i-1} \\ c_i \\ c_{i+1} \\ \\ \\ \end{matrix} \qquad (3.1)$$

$$c_{j-1}\ c_j\ \ c_{j+1}$$

We demonstrate the construction of multiresolution by reversing cubic B-spline subdivision. For this construction, we take the subdivision provided by midpoint knot insertion for uniform cubic B-splines (which provides a 2-scale dilation for a nesting of uniform-knot spline spaces). A finite (7-row, 5-column) portion of the interior of the $\mathbf{P}$ matrix for this subdivision is given in (3.1).

A biorthogonal multiresolution based upon $\mathbf{P}$ consists of the matrix $\mathbf{P}$ together with matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{Q}$ that satisfy (2.1). The construction method of Bartels and Samavati[2] is directed toward finding examples of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{Q}$ that are also banded, repetitive, and slanted; specifically, these characteristics should be true of the columns of $\mathbf{Q}$ and of the rows of $\mathbf{A}$ and $\mathbf{B}$.

The construction is staged as follows:

(1) a matrix $\mathbf{A}$ is produced that satisfies $\mathbf{AP} = \mathbf{I}$;
(2) trial versions of $\mathbf{B}$ and $\mathbf{Q}$ are produced, containing partially constrained symbolic entries, that satisfy $\mathbf{BP} = \mathbf{0}$ and $\mathbf{AQ} = \mathbf{0}$;
(3) the final step to fix $\mathbf{B}$ and $\mathbf{Q}$ by solving $\mathbf{BQ} = \mathbf{I}$.

In each stage, we can take advantage of the fact that the matrices are banded, repetitive, and slanted. This means that any scalar equation that forms a part of the matrix equation (2.1) is entirely characterized by the interaction of a row of the left-hand matrix with only one of a small number of adjacent columns of the right-hand matrix. (Alternatively, the scalar equations can be studied by looking at the interaction of a column of the right-hand matrix with only a small number of adjacent rows in the left-hand matrix.) The repetitiveness offers us the benefit of being able to characterize the entire matrix-matrix product (or at least, all of it except for a few special cases at the boundary) by studying how one representative row (or column) interacts with a small number of columns (or rows).

The construction is, of course, carried out only once for each choice of regular subdivision and connectivity. The rows of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{P}$, and $\mathbf{Q}$ are treated as *filters* that *decompose* the fine data F as in (2.2) and (2.3)

$$c_j = \sum_{\lambda} a_\lambda f_\lambda \qquad d_\ell = \sum_{\mu} b_\mu f_\lambda$$

and to *reconstruct* it as in (2.4)

$$f_i = \sum_{\rho} p_\rho c_\rho + \sum_{\sigma} q_\sigma d_\sigma \ .$$

As an example, the following illustrates the complete setup of equations to specify the elements of any general, interior (regular) row of $\mathbf{A}$ for cubic B-spline subdivision under the assumption that there are 7 nonzero elements in the row, and that in the row defining the value of c_j they are

centered on the position corresponding to f_i

$$\begin{bmatrix} a_{i-3} & a_{i-2} & a_{i-1} & a_i & a_{i+1} & a_{i+2} & a_{i+3} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad (3.2)$$

These are the only nontrivial scalar equations obtainable from the interior rows of $\mathbf{A}$ and interior columns of $\mathbf{P}$, assuming this width and positioning for the elements in each row of $\mathbf{A}$. The interaction of this row of $\mathbf{A}$ with any other interior column of $\mathbf{P}$ involves only sums of products with one factor in each product equal to zero. Interactions coming from the boundary will produce a small number of scalar equations distinct from the ones in (3.2). These distinct equations have no effect on the ones shown in (3.2). They will be solved separately to yield $\mathbf{A}$ values that are to be applied only to specific samples at the boundary. An example of this will be given in Sec. 5.

By solving the equations implied by (3.2) for the elements a_i which yield a minimum Euclidean norm(to have a good coarse approximation), we find that $[a_{i-3} \ \cdots \ a_{i+3}]$ is

$$\begin{bmatrix} \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} \end{bmatrix}$$

The sample $c_j = a_{i-3}f_{i-3} + \cdots + a_{i+3}f_{i+3}$ represents a local least squares estimate based upon the 7 consecutive fine samples $f_{i-3}, \ldots, f_{i+3}$ [2]. Figure 1 illustrates that in a curve, these 7 consecutive fine samples are those that are physically nearest to and symmetrically placed about c_j. This is arguably the configuration of choice for estimating c_j in a least squares sense from a local neighborhood about f_i. With the same motivation, 1, 3, 5, 9, or more consecutive samples $f_{i\pm\lambda}$ could be chosen for the estimate, producing other options for $\mathbf{A}$, and then correspondingly for $\mathbf{B}$ and $\mathbf{Q}$.

To handle the second matrix equation, $\mathbf{BP} = \mathbf{0}$, scalar equations corresponding to the nontrivial interactions of one row of $\mathbf{B}$ with the columns of $\mathbf{P}$ are set up in a similar way, assuming a number of nonzeros in a row of $\mathbf{B}$ and a position for those nonzeros in the row defining the generic element d_λ. These scalar equations (along with any additional ones desired to enforce,

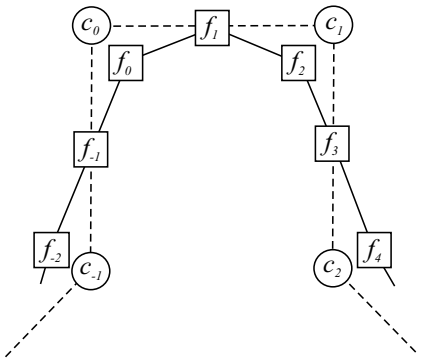

Fig. 1. The filters are based on a local indexing centered about a representative sample c_0.

for example, symmetry in the values of the row elements) are solved using a symbolic algebra system. Enough elements should be assumed in a row of **B** so that the solution is not fully defined and has free variables.

To handle the third matrix equation, $\mathbf{QA} = \mathbf{0}$, scalar equations corresponding to the nontrivial interactions of one column of **Q** with the rows of **A** are set up in a similar way, with assumptions about number and position of nonzeros being made. Additional conditions of symmetry are also possible. The equations are solved in symbolic algebra, and the result must also contain free variables.

The final matrix equation, $\mathbf{BQ} = \mathbf{I}$, is handled by using the symbolic results of the preceding two steps to generate the scalar equations representing the nontrivial interactions of a single row of **B** with the columns of **Q** (or a single column of **Q** with the rows of **B**), and the resulting (bilinear) equations are solved. Any remaining free variables may be fixed at will (our preference being to establish a norm of approximately unity for any column of **Q**).

A consistent set of solutions for cubic B-spline subdivision yields **A** as

follows (a 5×7 slice)

$$\mathbf{A} = \begin{bmatrix} \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} & 0 & 0 & 0 & 0 \\ \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} & 0 & 0 \\ \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} \\ 0 & 0 & \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} \\ 0 & 0 & 0 & 0 & \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} \end{bmatrix} \begin{matrix} \\ \\ c_{j-1}^k \\ c_j^k \\ c_{j+1}^k \\ \\ \end{matrix}$$

$$\begin{matrix} c_{i-1}^{k+1} & c_i^{k+1} & c_{i+1}^{k+1} \end{matrix}$$

A corresponding (5×7) slice of one possible $\mathbf{B}$ matrix is

$$\mathbf{B} = \begin{bmatrix} \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} & 0 & 0 & 0 & 0 \\ \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} & 0 & 0 \\ 0 & 0 & \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} \\ 0 & 0 & 0 & 0 & \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{13}{98} \end{bmatrix} \begin{matrix} \\ \\ d_{\ell-1}^k \\ d_\ell^k \\ d_{\ell+1}^k \\ \\ \end{matrix}$$

$$\begin{matrix} c_{i-1}^{k+1} & c_i^{k+1} & c_{i+1}^{k+1} \end{matrix}$$

And a corresponding (7×5) portion of a possible $\mathbf{Q}$ matrix is

$$\mathbf{Q} = \begin{bmatrix} 1 & -\frac{23}{52} & 0 & 0 & 0 \\ -\frac{63}{208} & -\frac{63}{208} & -\frac{23}{208} & 0 & 0 \\ -\frac{23}{52} & 1 & -\frac{23}{52} & 0 & 0 \\ -\frac{23}{208} & -\frac{63}{208} & -\frac{63}{208} & -\frac{23}{208} & 0 \\ 0 & -\frac{23}{52} & 1 & -\frac{23}{52} & 0 \\ 0 & -\frac{23}{208} & -\frac{63}{208} & -\frac{63}{208} & -\frac{23}{208} \\ 0 & 0 & -\frac{23}{52} & 1 & -\frac{23}{52} \end{bmatrix} \begin{matrix} \\ \\ c_{i-1}^{k+1} \\ c_i^{k+1} \\ c_{i+1}^{k+1} \\ \\ \\ \end{matrix}$$

$$\begin{matrix} d_{\ell-1}^k & d_\ell^k & d_{\ell+1}^k \end{matrix}$$

The construction of **B** and **Q** ends with the selection of leftover free parameters, and it is our custom to use these so that the maximum magnitude element (the infinity vector norm) of each column of **Q** is comparable to 1 in magnitude, expecting that this means the contribution of each d_ℓ in any residual will be comparable to the magnitude of d_ℓ itself. The residuals $f_i - \widetilde{f}_i$ to which d_ℓ contributes are those corresponding to the nonzero elements of the ℓ^{th} column of **Q**. The number of elements in $\{f\}$ corresponds to the number of rows in **P**. The number of columns of **P** corresponds to the number of elements in $\{c\}$ and the number of columns of **Q** corresponds to the number of elements in $\{d\}$. If the columns of **P** and **Q** are adjoined, the result is a square matrix (whose inverse is the matrix with the rows of **A** adjoined below by the rows of **B**). The number of columns of [**P Q**], being the number of elements in $\{c\} \bigcup \{d\}$, is also the number of elements in $\{f\}$.

Throughout the remainder of this chapter we shall be using the term **matrix** to refer to the *decomposition* information, **A** and **B**, and the *reconstruction* information, **P** and **Q**, in its entire matrix format; i.e., capable of acting simultaneously on all the information $\{f\}$, $\{c\}$, and $\{d\}$, as laid out in vectors, in the manner of (2.2), (2.3), and (2.4). We shall be using the term **filter** to refer to the nonzero entries in a representative row of **A** and **B** and a representative column of **P** and **Q**. We simply denote theses filters by $\mathbf{a}, \mathbf{b}, \mathbf{p}$ and $\mathbf{q}$. This helps us to compactly represents all involving filters of cubic B-spline as

$$\begin{array}{l} \mathbf{a} = \left[\begin{array}{ccccccc} \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} \end{array} \right] \\ \mathbf{b} = \left[\begin{array}{ccccc} \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} \end{array} \right] \\ \mathbf{p} = \left[\begin{array}{ccccc} \frac{1}{8} & \frac{1}{2} & \frac{3}{4} & \frac{1}{2} & \frac{1}{8} \end{array} \right] \\ \mathbf{q} = \left[\begin{array}{ccccccc} -\frac{23}{208} & -\frac{23}{52} & -\frac{63}{208} & 1 & -\frac{63}{208} & -\frac{23}{52} & -\frac{23}{208} \end{array} \right] . \end{array} \tag{3.3}$$

An important caveat to the filter vector notation in (3.3) is that **a** and **b** represent regular *rows* of **A** and **B**, while **p** and **q** represent regular *columns* of **P** and **Q**. Thus the application of **a** (or **b**) to a sample vector is similar to convolution, as the filter vector is slid along the sample vector. Conversely, the application of **p** (or **q**) is two convolutions, with one kernel consisting of the even entries and another filled with the odd entries (due to the column shift required by a 2-scale dilation). Section 6 illustrates how to interpret these filters algorithmically.

4. Other B-spline Multiresolution Filters

The construction in Sec. 3 is general enough to be used for other subdivision methods, particularly B-spline subdivisions. Due to the fact that having two levels of smoothness is enough for most applications in imaging and graphics, only quadratic and cubic B-spline subdivisions are considered here. However, the multiresolution filters obtained from this method of construction are not unique and there are variety of options. For example, the filters in (3.3) are result of starting with the width seven for **A**. Different filters can be derived by changing the width of **A**. Wider filters result in a better coarse approximation of the fine samples but they are harder to implement and require more computations. In addition, it is possible to add constraints to the construction to obtain better filter values, such as inverse powers of two. Here we report some other alternative filter sets that may be useful in different applications.

4.1. *Short Filters for Cubic B-spline*

If we start with a width of three for **A** in the construction, we obtain

$$\begin{aligned} \mathbf{a} &= \left[\ -\tfrac{1}{2}\ 1\ -\tfrac{1}{2}\ \right] \\ \mathbf{b} &= \left[\ \tfrac{1}{4}\ -1\ \tfrac{3}{2}\ -1\ \tfrac{1}{4}\ \right] \\ \mathbf{p} &= \left[\ \tfrac{1}{8}\ \tfrac{1}{2}\ \tfrac{3}{4}\ \tfrac{1}{2}\ \tfrac{1}{8}\ \right] \\ \mathbf{q} &= \left[\ \tfrac{1}{4}\ 1\tfrac{1}{4}\ \right]\ . \end{aligned} \tag{4.1}$$

Although these filters are very compact and easy to implement, they often fail to generate a good coarse approximation.

4.2. *Cubic B-spline Filters: Inverse Powers of Two*

Having powers and inverse powers of two is desirable for implementing multiresolution in hardware. It is possible to add constraints to the construction that is described in Sec. 3 to obtain filter values as inverse powers of two. This is not always successful, but it is certainly gratifying when it is. The construction is nicely successful for cubic B-spline with a width of seven for **a**

$$\begin{aligned} \mathbf{a} &= \left[\ \tfrac{1}{8}\ -\tfrac{1}{2}\ \tfrac{3}{8}\ 1\ \tfrac{3}{8}\ -\tfrac{1}{2}\ \tfrac{1}{8}\ \right] \\ \mathbf{b} &= \left[\ -\tfrac{1}{8}\ \tfrac{1}{2}\ -\tfrac{3}{4}\ \tfrac{1}{2}\ -\tfrac{1}{8}\ \right] \\ \mathbf{p} &= \left[\ \tfrac{1}{8}\ \tfrac{1}{2}\ \tfrac{3}{4}\ \tfrac{1}{2}\ \tfrac{1}{8}\ \right] \\ \mathbf{q} &= \left[\ \tfrac{1}{8}\ -\tfrac{1}{2}\ \tfrac{3}{8}\ 1\ \tfrac{3}{8}\ -\tfrac{1}{2}\ \tfrac{1}{8}\ \right]\ . \end{aligned}$$

The quality of the coarse approximation in this case is near to optimal filters in (3.3).

4.3. *Short Filters for Quadratic B-spline*

The local filters of quadratic B-spline can be constructed based on reversing Chaikin subdivision[5], for which the underlying scale functions are the quadratic B-splines. The smallest width that can generate a non-trivial filters is four, which results in the following filters

$$\begin{array}{l} \mathbf{a} = \left[\begin{array}{cccc} -\frac{1}{4} & \frac{3}{4} & \frac{3}{4} & -\frac{1}{4} \end{array} \right] \\ \mathbf{b} = \left[\begin{array}{cccc} -\frac{1}{4} & \frac{3}{4} & -\frac{3}{4} & \frac{1}{4} \end{array} \right] \\ \mathbf{p} = \left[\begin{array}{cccc} \frac{1}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} \end{array} \right] \\ \mathbf{q} = \left[\begin{array}{cccc} -\frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & \frac{1}{4} \end{array} \right] . \end{array} \tag{4.2}$$

These filters are appealingly simple, yet their quality is reasonably good (see Sec. 8).

4.4. *Wide Filters for Quadratic B-spline*

Starting with a width of eight for $\mathbf{a}$, the following filters are obtained

$$\begin{array}{l} \mathbf{a} = \left[\begin{array}{cccccccc} \frac{3}{40} & -\frac{9}{40} & -\frac{1}{40} & \frac{27}{40} & \frac{27}{40} & -\frac{1}{40} & -\frac{9}{40} & \frac{3}{40} \end{array} \right] \\ \mathbf{b} = \left[\begin{array}{cccc} -\frac{27}{160} & \frac{81}{160} & -\frac{81}{160} & \frac{27}{160} \end{array} \right] \\ \mathbf{p} = \left[\begin{array}{cccc} \frac{1}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} \end{array} \right] \\ \mathbf{q} = \left[\begin{array}{cccccccc} -\frac{1}{9} & -\frac{1}{3} & \frac{1}{27} & 1 & -1 & -\frac{1}{27} & \frac{1}{3} & \frac{1}{9} \end{array} \right] . \end{array} \tag{4.3}$$

As shown in Sec. 8.1, these filters generate very high compression rates for images compression.

5. Extraordinary (Boundary) Filters

All multiresolution filters in Sec. 3 and 4 are *regular*, meaning they are applicable only to data with full neighborhoods. Using symmetric extension (Sec. 7.3.1), we can apply such regular filters to curves, surfaces, and images with boundaries. However, boundary interpolation is often strongly desired. To have this property, we must sacrifice the regularity of the filters near to the boundary.

To fulfill the interpolation condition for B-spline representations, multiple knots are used at the ends of the knot sequence, corresponding to the beginning and ending portions of any data that the filters might operate on. This knot multiplicity creates irregular or *extraordinary* parts in the subdivision matrix. We use a block matrix notation to separate the boundary filters from the regular filters. For example, the $\mathbf{P}$ matrix for cubic B-Splines with the interpolation condition is shown in (5.1). In this notation $\mathbf{P}_s$ shows the extraordinary parts of the subdivision matrix near to the

start of the sample vector. Similarly, $\mathbf{P}_e$ refers to the extraordinary parts near to the end. And finally, $\mathbf{P}_r$ shows the regular portion of this matrix.

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_s \\ \mathbf{P}_r \\ \mathbf{P}_e \end{bmatrix} , \tag{5.1}$$

where

$$\mathbf{P}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & \cdots \\ 0 & \frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 & \cdots \\ 0 & \frac{3}{16} & \frac{11}{16} & \frac{1}{8} & 0 & 0 & \cdots \end{bmatrix} ,$$

$$\mathbf{P_r} = \begin{bmatrix} 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & 0 & 0 & 0 & \cdots \\ & & & & \vdots & & & & & \end{bmatrix} ,$$

$$\mathbf{P}_e = \begin{bmatrix} \cdots & 0 & 0 & \frac{1}{8} & \frac{11}{16} & \frac{3}{16} & 0 \\ \cdots & 0 & 0 & 0 & \frac{1}{4} & \frac{3}{4} & 0 \\ \cdots & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \cdots & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} .$$

5.1. *Boundary Filters for Cubic B-Spline*

Again we present our construction in the context of cubic B-Spline subdivision.

5.1.1. *Construction of* $\mathbf{A}$

Having extraordinary filters at the boundary of $\mathbf{P}$ affects our construction method and usually causes extraordinary filters for $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{Q}$ too. In the first step, we would like to find $\mathbf{A}$ such that $\mathbf{AP} = \mathbf{I}$. Any width of $\mathbf{A}$ filter can be investigated, but we have tried to find a width consistent with the regular filters' widths. For the first row of $\mathbf{A}$ consisting of only a_0, the only interaction with $\mathbf{P}$ corresponds to the first $\mathbf{P}$ column.

By the way the subdivision is defined at the boundary, investigating a minimal **A** filter is the obvious thing to do, since the subdivision simply reproduces the extreme samples $c_{\min} = f_{\min}$. Nevertheless, for completeness, the setup for determining that fact is:

$$c_{\min} = a_0 \, f_{\min} \;\Rightarrow\; a_0 = 1.$$

The second row of **A** , for estimating the second coarse sample, is more interesting. The **A** filter investigated is five elements long, which is the closest possible to the seven-element filter being used for the interior samples. For the five-element **A** filter being investigated, there are only four relevant **P** columns, and the equations that are generated by forming the relevant section of $\mathbf{AP} = \mathbf{I}$ are

$$\begin{aligned} 1\,a_0 + \frac{1}{2}\,a_1 &= 0 \\ \frac{1}{2}\,a_1 + \frac{3}{4}\,a_2 + \frac{3}{16}\,a_3 &= 1 \\ \frac{1}{4}\,a_2 + \frac{11}{16}\,a_3 + \frac{1}{2}\,a_4 &= 0 \\ \frac{1}{8}\,a_3 + \frac{1}{2}\,a_4 &= 0\ . \end{aligned}$$

This creates the second row of **A**

$$\mathbf{a}_2 = \left[\, -\tfrac{49}{139} \;\; \tfrac{98}{139} \;\; \tfrac{135}{139} \;\; -\tfrac{60}{139} \;\; \tfrac{15}{139} \,\right] \ .$$

The third row **A** can be found with the same method; for this row a seven element filter $[a_0, a_1, a_2, a_3, a_4, a_5, a_6]$ can be considered. This row has non-zero interaction with the first five columns of **P**, resulting in

$$\mathbf{a}_3 = \left[\, \tfrac{9}{50} \;\; -\tfrac{9}{25} \;\; -\tfrac{2}{25} \;\; \tfrac{32}{25} \;\; \tfrac{43}{100} \;\; -\tfrac{3}{5} \;\; \tfrac{3}{20} \,\right] \ .$$

If we use the same kind of the blocked matrix notation for **A**, where $\mathbf{A}_s$, $\mathbf{A}_r$ and $\mathbf{A}_e$ respectively refer to the extraordinary block near to the start, the regular block and the extraordinary block near to the end, then the result of the boundary analysis is:

$$\mathbf{A}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -\frac{49}{139} & \frac{98}{139} & \frac{135}{139} & -\frac{60}{139} & \frac{15}{139} & 0 & 0 & 0 & \cdots \\ \frac{9}{50} & -\frac{9}{25} & -\frac{2}{25} & \frac{32}{25} & \frac{43}{100} & -\frac{3}{5} & \frac{3}{20} & 0 & \cdots \end{bmatrix} \ ,$$

$$\mathbf{A_r} = \begin{bmatrix} 0 & 0 & \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \frac{23}{196} & -\frac{23}{49} & \frac{9}{28} & \frac{52}{49} & \frac{9}{28} & -\frac{23}{49} & \frac{23}{196} & \cdots \\ & & & & & \vdots & & & & & & \end{bmatrix},$$

$$\mathbf{A}_e = \begin{bmatrix} \cdots & 0 & 0 & \frac{3}{20} & -\frac{3}{5} & \frac{43}{100} & \frac{32}{25} & -\frac{2}{25} & -\frac{9}{25} & \frac{9}{50} \\ \cdots & 0 & 0 & 0 & 0 & \frac{15}{139} & -\frac{60}{139} & \frac{135}{139} & \frac{98}{139} & -\frac{49}{139} \\ \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

5.1.2. **B** *and* **Q**

To establish **B** filters near the boundary, we proceed in the way that we did for **A**. To begin, we would try solving for the fist row of **B** making it as near to the size of interior **B** filters as possible in that position. The first **B** filter configuration that yields nontrivial elements has the width five. The interactions of such a filter with the boundary **P** filter contribute to the equations $\mathbf{BP} = \mathbf{0}$ as

$$\begin{aligned} 1\,b_0 + \frac{1}{2}\,b_1 &= 0 \\ \frac{1}{2}\,b_1 + \frac{3}{4}\,b_2 + \frac{3}{16}\,b_3 &= 0 \\ \frac{1}{4}\,b_2 + \frac{11}{16}\,b_3 + \frac{1}{2}\,b_4 &= 0 \\ \frac{1}{8}\,b_3 + \frac{1}{2}\,b_4 &= 0\ . \end{aligned}$$

For the second boundary row of **B** filter, we have:

$$\begin{aligned} \frac{3}{4}\,b_0 + \frac{3}{16}\,b_1 &= 0 \\ \frac{1}{4}\,b_0 + \frac{11}{16}\,b_1 + \frac{1}{2}\,b_2 + \frac{1}{8}\,b_3 &= 0 \\ \frac{1}{8}\,b_1 + \frac{1}{2}\,b_2 + \frac{3}{4}\,b_3 + \frac{1}{2}\,b_4 &= 0 \\ \frac{1}{8}\,b_3 + \frac{1}{2}\,b_4 &= 0\ . \end{aligned}$$

Similar steps will be set up to generate the equations $\mathbf{AQ} = \mathbf{0}$. In addition, we need to the set up for contributing to the equations $\mathbf{BQ} = \mathbf{I}$ from a **Q** columns. When all equations have been generated and solved,

proceeding as we have done in terms of the lengths chosen for the boundary filters, we obtain the boundary $\mathbf{B}$ and $\mathbf{Q}$. The resulting filter for $\mathbf{B}$ is

$$\mathbf{B}_s = \begin{bmatrix} -\frac{45}{139} & \frac{90}{139} & -\frac{135}{278} & \frac{30}{139} & -\frac{15}{278} & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \frac{57}{490} & -\frac{114}{245} & \frac{171}{245} & -\frac{114}{245} & \frac{57}{490} & 0 & 0 & \cdots \end{bmatrix},$$

$$\mathbf{B_r} = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{13}{98} & -\frac{26}{49} & \frac{39}{49} & -\frac{26}{49} & \frac{13}{98} & 0 & \cdots \\ & & & & & \vdots & & & & & & & \end{bmatrix},$$

$$\mathbf{B}_e = \begin{bmatrix} \cdots & 0 & 0 & 0 & \frac{57}{490} & -\frac{114}{245} & \frac{171}{245} & -\frac{114}{245} & \frac{57}{490} & 0 & 0 \\ \cdots & 0 & 0 & 0 & 0 & 0 & -\frac{15}{278} & \frac{30}{139} & -\frac{135}{278} & \frac{90}{139} & -\frac{45}{139} \end{bmatrix}.$$

And the resulting filters for $\mathbf{Q}$ are

$$\mathbf{Q}_s = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -\frac{2033}{3000} & -\frac{49}{152} & 0 & 0 & 0 & 0 & \cdots \\ \frac{2137}{12000} & -\frac{289}{608} & -\frac{23}{208} & 0 & 0 & 0 & \cdots \\ \frac{139}{500} & 1 & -\frac{23}{52} & 0 & 0 & 0 & \cdots \\ \frac{139}{2000} & -\frac{347}{912} & -\frac{63}{208} & -\frac{23}{208} & 0 & 0 & \cdots \\ 0 & -\frac{115}{228} & 1 & -\frac{23}{52} & 0 & 0 & \cdots \\ 0 & -\frac{115}{912} & -\frac{63}{208} & -\frac{63}{208} & -\frac{23}{208} & 0 & \cdots \end{bmatrix},$$

$$\mathbf{Q_r} = \begin{bmatrix} 0 & 0 & -\frac{23}{52} & 1 & -\frac{23}{52} & 0 & 0 & \cdots \\ 0 & 0 & -\frac{23}{208} & -\frac{63}{208} & -\frac{63}{208} & -\frac{23}{208} & 0 & \cdots \\ & & & \vdots & & & & \end{bmatrix},$$

$$\mathbf{Q}_e = \begin{bmatrix} \cdots & 0 & -\frac{23}{208} & -\frac{63}{208} & -\frac{63}{208} & -\frac{115}{912} & 0 \\ \cdots & 0 & 0 & -\frac{23}{52} & 1 & -\frac{115}{228} & 0 \\ \cdots & 0 & 0 & -\frac{23}{208} & -\frac{63}{208} & -\frac{347}{912} & \frac{139}{2000} \\ \cdots & 0 & 0 & 0 & -\frac{23}{52} & 1 & \frac{139}{500} \\ \cdots & 0 & 0 & 0 & -\frac{23}{208} & -\frac{289}{608} & \frac{2137}{12000} \\ \cdots & 0 & 0 & 0 & 0 & -\frac{49}{152} & -\frac{2033}{3000} \\ \cdots & 0 & 0 & 0 & 0 & 0 & 1 \\ \cdots & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

5.2. *Boundary Filters for Short Cubic B-Spline*

Using the same kind of construction as Sec. 5.1, we can construct extraordinary filters for the narrow cubic B-Spline filters of Sec. 4.1. In this case, $\mathbf{A}$ becomes

$$\mathbf{A}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -1 & 2 & 0 & 0 & 0 & 0 & 0 & \cdots \end{bmatrix},$$

$$\mathbf{A_r} = \begin{bmatrix} 0 & 0 & -\frac{1}{2} & 2 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 2 & -\frac{1}{2} & 0 & 0 & 0 & \cdots \\ & & & & & \vdots & & & & & \end{bmatrix},$$

$$\mathbf{A}_e = \begin{bmatrix} \cdots & 0 & 0 & 0 & 2 & -1 \\ \cdots & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The corresponding $\mathbf{B}$ matrix is

$$\mathbf{B}_s = \begin{bmatrix} \frac{3}{4} & -\frac{3}{2} & \frac{9}{8} & -\frac{1}{2} & \frac{1}{8} & 0 & \cdots \end{bmatrix}$$

$$\mathbf{B_r} = \begin{bmatrix} 0 & 0 & \frac{1}{4} & -1 & \frac{3}{2} & -1 & \frac{1}{4} & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \frac{1}{4} & -1 & \frac{3}{2} & -1 & \frac{1}{4} & 0 & 0 & \cdots \\ & & & & & & \vdots & & & & & \end{bmatrix},$$

$$\mathbf{B}_e = \begin{bmatrix} \cdots & 0 & 0 & 0 & 0 & \frac{1}{8} & -\frac{1}{2} & \frac{9}{8} & -\frac{3}{2} & \frac{3}{4} \end{bmatrix} .$$

And finally, the $\mathbf{Q}$ matrix is

$$\mathbf{Q}_s = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \end{bmatrix} ,$$

$$\mathbf{Q}_\mathbf{r} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots \\ \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & 0 & \cdots \\ 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \cdots \\ & & \vdots & & & \end{bmatrix} ,$$

$$\mathbf{Q}_e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & \cdots \end{bmatrix} .$$

5.3. *Boundary Filters for Short Quadratic B-Spline*

By the same method, we can generate a full set of multiresolution matrices for quadratic B-spline subdivision (commonly referred to as Chaikin subdivision). The $\mathbf{P}$ filter for quadratic B-spline subdivision is $\mathbf{p} = \left[\frac{1}{4} \ \frac{3}{4} \ \frac{3}{4} \ \frac{1}{4}\right]$. The blocked matrix notation for the synthesis filter $\mathbf{P}$ is

$$\mathbf{P}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & \cdots \end{bmatrix} ,$$

$$\mathbf{P}_r = \begin{bmatrix} 0 & \frac{3}{4} & \frac{1}{4} & 0 & 0 & \cdots \\ 0 & \frac{1}{4} & \frac{3}{4} & 0 & 0 & \cdots \\ 0 & 0 & \frac{3}{4} & \frac{1}{4} & 0 & \cdots \\ 0 & 0 & \frac{1}{4} & \frac{3}{4} & 0 & \cdots \\ & & \vdots & & & \end{bmatrix} ,$$

$$\mathbf{P}_e = \begin{bmatrix} \cdots & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \cdots & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Similarly, the $\mathbf{A}$ matrix becomes

$$\mathbf{A}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -\frac{1}{2} & 1 & \frac{3}{4} & -\frac{1}{4} & 0 & 0 & \cdots \end{bmatrix},$$

$$\mathbf{A}_r = \begin{bmatrix} 0 & 0 & -\frac{1}{4} & \frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & -\frac{1}{4} & \frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & \cdots \\ & & & & \vdots & & & & \end{bmatrix},$$

$$\mathbf{A}_e = \begin{bmatrix} \cdots & 0 & 0 & -\frac{1}{4} & \frac{3}{4} & 1 & -\frac{1}{2} \\ \cdots & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

And $\mathbf{B}$ is

$$\mathbf{B}_s = \begin{bmatrix} -\frac{1}{2} & 1 & -\frac{3}{4} & \frac{1}{4} & 0 & 0 & 0\cdots \\ 0 & 0 & -\frac{1}{4} & \frac{3}{4} & -\frac{3}{4} & \frac{1}{4} & 0\cdots \end{bmatrix},$$

$$\mathbf{B}_r = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & \cdots \\ & & & & & \vdots & & & & & \end{bmatrix},$$

$$\mathbf{B}_e = \begin{bmatrix} \cdots & 0 & 0 & \frac{1}{4} & -\frac{3}{4} & 1 & -\frac{1}{2} \end{bmatrix}.$$

Finally, for $\mathbf{Q}$ we have

$$\mathbf{Q}_s = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ \frac{1}{2} & 0 & 0 & 0 & \cdots \\ -\frac{3}{4} & \frac{1}{4} & 0 & 0 & \cdots \\ -\frac{1}{4} & \frac{3}{4} & 0 & 0 & \cdots \\ 0 & -\frac{3}{4} & -\frac{1}{4} & 0 & \cdots \\ 0 & -\frac{1}{4} & -\frac{3}{4} & 0 & \cdots \end{bmatrix},$$

$$\mathbf{Q}_r = \begin{bmatrix} 0 & 0 & \frac{3}{4} & -\frac{1}{4} & 0 & 0 & \cdots \\ 0 & 0 & \frac{1}{4} & -\frac{3}{4} & 0 & 0 & \cdots \\ 0 & 0 & 0 & \frac{3}{4} & -\frac{1}{4} & 0 & \cdots \\ 0 & 0 & 0 & \frac{1}{4} & -\frac{3}{4} & 0 & \cdots \\ & & & \vdots & & & \end{bmatrix},$$

$$\mathbf{Q}_e = \begin{bmatrix} \cdots & 0 & 0 & 0 & \frac{1}{2} \\ \cdots & 0 & 0 & 0 & 0 \end{bmatrix}.$$

5.4. *Boundary Filters for Wide Quadratic B-Spline*

In the case of boundary filters for wide quadratic B-Spline, $\mathbf{P}$ is the same as Sec. 5.3 . The $\mathbf{A}$ matrix is

$$\mathbf{A}_s = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ -\frac{41}{141} & \frac{82}{141} & \frac{45}{47} & -\frac{5}{141} & -\frac{15}{47} & \frac{5}{47} & 0 & 0 & 0 & \cdots \\ \frac{41}{425} & -\frac{82}{425} & -\frac{41}{425} & \frac{287}{425} & \frac{297}{425} & -\frac{11}{425} & -\frac{99}{425} & \frac{33}{425} & 0 & \cdots \end{bmatrix},$$

$$\mathbf{A}_r = \begin{bmatrix} 0 & 0 & \frac{3}{40} & -\frac{9}{40} & -\frac{1}{40} & \frac{27}{40} & \frac{27}{40} & -\frac{1}{40} & -\frac{9}{40} & \frac{3}{40} & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \frac{3}{40} & -\frac{9}{40} & -\frac{1}{40} & \frac{27}{40} & \frac{27}{40} & -\frac{1}{40} & -\frac{9}{40} & \frac{3}{40} & 0 & \cdots \\ & & & & & & & \vdots & & & & & & \end{bmatrix},$$

$$\mathbf{A}_e = \begin{bmatrix} \cdots & 0 & \frac{33}{425} & -\frac{99}{425} & -\frac{11}{425} & \frac{297}{425} & \frac{287}{425} & -\frac{41}{425} & -\frac{82}{425} & \frac{41}{425} \\ \cdots & 0 & 0 & 0 & \frac{5}{47} & -\frac{15}{47} & -\frac{5}{141} & \frac{45}{47} & \frac{82}{141} & -\frac{41}{141} \\ \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The second decomposition matrix, $\mathbf{B}$, is

$$\mathbf{B}_s = \begin{bmatrix} -\frac{27}{80} & \frac{27}{40} & -\frac{81}{160} & \frac{27}{160} & 0 & 0 & \cdots \end{bmatrix},$$

$$\mathbf{B}_r = \begin{bmatrix} 0 & 0 & -\frac{27}{160} & \frac{81}{160} & -\frac{81}{160} & \frac{27}{160} & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & -\frac{27}{160} & \frac{81}{160} & -\frac{81}{160} & \frac{27}{160} & 0 & 0 & \cdots \\ & & & & & \vdots & & & & & \end{bmatrix},$$

$$\mathbf{B}_e = \begin{bmatrix} \cdots & 0 & 0 & \frac{27}{160} & -\frac{81}{160} & \frac{27}{40} & -\frac{27}{80} \end{bmatrix}.$$

And $\mathbf{Q}$ is

$$\mathbf{Q}_s = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots \\ \frac{4000}{3807} & -\frac{400}{1269} & 0 & 0 & 0 & \cdots \\ -\frac{2296}{3995} & -\frac{1928}{21573} & -\frac{88}{765} & 0 & 0 & \cdots \\ -\frac{328}{323595} & \frac{21416}{21573} & -\frac{88}{255} & 0 & 0 & \cdots \\ \frac{164}{765} & -\frac{49}{51} & \frac{58}{2295} & -\frac{1}{9} & 0 & \cdots \\ \frac{164}{2295} & -\frac{11}{459} & \frac{254}{255} & -\frac{1}{3} & 0 & \cdots \end{bmatrix},$$

$$\mathbf{Q}_r = \begin{bmatrix} 0 & \frac{1}{3} & -1 & \frac{1}{27} & -\frac{1}{9} & 0 & 0 & \cdots \\ 0 & \frac{1}{9} & -\frac{1}{27} & 1 & -\frac{1}{3} & 0 & 0 & \cdots \\ 0 & 0 & \frac{1}{3} & -1 & \frac{1}{27} & -\frac{1}{9} & 0 & \cdots \\ 0 & 0 & \frac{1}{9} & -\frac{1}{27} & 1 & -\frac{1}{3} & 0 & \cdots \\ & & & \vdots & & & & \end{bmatrix},$$

$$\mathbf{Q}_e = \begin{bmatrix} \cdots & 0 & \frac{1}{3} & \frac{254}{255} & -\frac{11}{459} & \frac{164}{2295} \\ \cdots & 0 & \frac{1}{9} & \frac{58}{2295} & -\frac{49}{51} & \frac{164}{765} \\ \cdots & 0 & 0 & -\frac{88}{255} & \frac{21416}{21573} & -\frac{328}{323595} \\ \cdots & 0 & 0 & -\frac{88}{765} & -\frac{1928}{21573} & -\frac{2296}{3995} \\ \cdots & 0 & 0 & 0 & -\frac{400}{1269} & \frac{4000}{3807} \\ \cdots & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

6. Efficient Algorithm

We show how an efficient algorithm can be made based on the multiresolution filters for quadratic B-spline subdivision, according to the matrix forms presented in Sec. 5.3.

For all algorithms, we have focused on doing just one step of decomposition or reconstruction. Each algorithm can be used multiple times to construct a hierarchical wavelet transform. In all cases F represents the vector of high-resolution data, C represent low-resolution data and D represents the detail vector.

Conceptually, a multiresolution algorithm performs the matrix-vector operations specified in (2.2), (2.3), and (2.4). However, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{P}$, and $\mathbf{Q}$ are regular banded matrices, so using matrix-vector operations is not efficient. By using the blocked and banded structure of these matrices, more efficient ($O(n)$ versus $O(n^2)$) algorithms can obtained.

The first algorithm is REDUCE-RESOLUTION. In this algorithm, $F[1..m]$ is the input fine data and the vector $C[1..n]$ is the output coarse approximation. The index i traverses F and j traverses C.

REDUCE-RESOLUTION($F[1..m]$)

1 $C_1 = F_1$
2 $C_2 = -\frac{1}{2}F_1 + F_2 + \frac{3}{4}F_3 - \frac{1}{4}F_4$
3 $j = 3$
4 **for** $i = 2$ **to** $m - 5$ **step** 2
5 $\quad C_j = -\frac{1}{4}F_i + \frac{3}{4}F_{i+1} + \frac{3}{4}F_{i+2} - \frac{1}{4}F_{i+3}$
6 $\quad j = j + 1$
7 **endfor**
8 $C_j = -\frac{1}{4}F_{m-3} + \frac{3}{4}F_{m-2} + F_{m-1} - \frac{1}{2}F_m$
9 $C_{j+1} = F_m$
10 **return** $C[1..j + 1]$

Lines 1–2 in REDUCE-RESOLUTION correspond to the $\mathbf{A}_s$ matrix, while lines 8–9 correspond to the $\mathbf{A}_e$ matrix. The `for` loop represents the application of the regular $\mathbf{A}_r$ block.

The second algorithm is FIND-DETAILS. We can again identify blocks corresponding to $\mathbf{B}_s$, $\mathbf{B}_r$, and $\mathbf{B}_e$.

FIND-DETAILS($F[1..m]$)

1 $D_1 = -\frac{1}{2}F_1 + F_2 - \frac{3}{4}F_3 + \frac{1}{4}F_4$
2 $D_2 = -\frac{1}{4}F_3 + \frac{3}{4}F_4 - \frac{3}{4}F_5 + \frac{1}{4}F_6$
3 $j = 3$

4 **for** $i = 5$ **to** $m - 5$ **step** 2
5 $D_j = \frac{1}{4}F_i - \frac{3}{4}F_{i+1} + \frac{3}{4}F_{i+2} - \frac{1}{4}F_{i+3}$
6 $j = j + 1$
7 **endfor**
8 $D_j = \frac{1}{4}F_{m-3} - \frac{3}{4}F_{m-2} + F_{m-1} - \frac{1}{2}F_m$
9 **return** $D[1..j]$

For reconstruction, we need to compute $\mathbf{P}C + \mathbf{Q}D$. The 2-scale column shift property causes to have two kinds of regular rows(odd and even) for $\mathbf{P}$ and $\mathbf{Q}$. This only requires a simple odd/even regular rules as demonstrated by the algorithm RECONSTRUCTION.

RECONSTRUCTION($C[1..n], D[1..s]$)
1 $E_1 = 0D_1$
2 $E_2 = \frac{1}{2}D_1$
3 $E_3 = -\frac{3}{4}D_1 + \frac{1}{4}D_2$
4 $E_4 = -\frac{1}{4}D_1 + \frac{3}{4}D_2$
5 $E_5 = -\frac{3}{4}D_2 - \frac{1}{4}D_3$
6 $E_6 = -\frac{1}{4}D_2 - \frac{3}{4}D_3$
7 $j = 7$
8 **for** $i = 3$ **to** $s - 1$
9 $E_j = \frac{3}{4}D_i - \frac{1}{4}D_{i+1}$
10 $E_{j+1} = \frac{1}{4}D_i - \frac{3}{4}D_{i+1}$
11 $j = j + 2$
12 **endfor**
13 $E_j = \frac{1}{2}D_s$
14 $E_{j+1} = 0D_s$
15
16 $F_1 = C_1 + E_1$
17 $F_2 = (\frac{1}{2}C_1 + \frac{1}{2}C_2) + E_2$
18 $j = 3$
19 **for** $i = 2$ **to** $n - 2$
20 $F_j = (\frac{3}{4}C_i + \frac{1}{4}C_{i+1}) + E_j$
21 $F_{j+1} = (\frac{1}{4}C_i + \frac{3}{4}C_{i+1}) + E_{j+1}$
22 $j = j + 2$
23 **endfor**
24 $F_j = (\frac{1}{2}C_{n-1} + \frac{1}{2}C_n) + E_j$
25 $F_{j+1} = C_n + E_{j+1}$
26 **return** $F[1..j + 1]$

Lines 1–14 in RECONSTRUCTION construct the $E = \mathbf{Q}D$ term. Lines 1 through 6 correspond to $\mathbf{Q}_s$, and lines 13–14 apply $\mathbf{Q}_e$. The `for` loop at line 8 is for the regular block $\mathbf{Q}_r$. In line 1, E_1 has been set to $0D_1$ instead of 0 to have general algorithm that can work for the data with any dimension induced by D. Lines 24 trough 37 make $F = \mathbf{P}C + E$. Again the terms $\mathbf{P}_e$, $\mathbf{P}_r$ and $\mathbf{P}_e$ are distinguishable in the algorithm.

Note that m, the size of the high-resolution data F, is equal to $n + s$; it is clear that the running time of all three algorithms is linear in m.

7. Extensions

The regular and extraordinary filters presented thus far are intended for use on non-periodic data sets, such as open-ended curves. For many applications, we may have data that does not fit this definition; for example, periodic curves, tensor-product surfaces, or 2D and 3D images. In this section, we show how to use the local filters for these kinds of objects.

7.1. *Periodic (Closed) Curves*

For many applications, boundary-interpolating filters are not desirable. For closed curves (isomorphic to a circle), we can instead use periodic filters. In periodic (equivalently *closed*) curves, the regular filter values are applied to all samples; there is no concept of a boundary, as the signal F is assumed to wrap around on itself. For $F = \{f_1, \ldots, f_m\}$, we implicitly set $f_{m+x} = f_x$ for $x \geq 1$ and $f_x = f_{m-x}$ for $x < 1$. Figure 2 illustrates this wrapping.

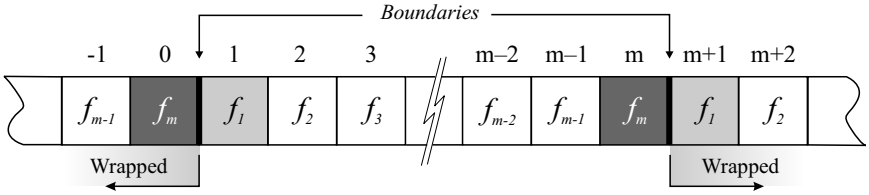

Fig. 2. For periodic curves, the left and right boundaries are implicitly connected, wrapping the sample vector back on itself.

In the matrix form of periodic subdivision, the regular columns of $\mathbf{P}$ will wrap around the top and bottom of the matrix, rather than being terminated with the special boundary filters in the non-periodic (open) case.

Consider the matrix corresponding to periodic cubic B-spline subdivision

$$\mathbf{P}_{periodic} = \begin{bmatrix} \frac{3}{4} & \frac{1}{8} & 0 & \cdots & 0 & \frac{1}{8} \\ \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 \\ & \vdots & & & & \\ \cdots & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & \cdots \\ \cdots & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots \\ & \vdots & & & & \\ \frac{1}{8} & 0 & \cdots & 0 & \frac{1}{8} & \frac{3}{4} \\ \frac{1}{2} & 0 & \cdots & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

The remaining matrices **A**, **B**, and **Q** are formed by a similar wrapping of the rows or columns.

From an implementation perspective, periodic data is easier to work with because there are no special boundary cases. We need only modify the indexing of the samples to ensure that the wrapping is done correctly.

7.2. *Tensor Product Surfaces*

Multiresolution schemes for 1D data, such as the cubic or quadratic B-splines schemes developed earlier, can be applied to surface patches by a straightforward extension.

A surface patch is defined by a regular 2-dimensional grid of vertices. The regularity allows the patch to be split into two arbitrary dimensions, usually denoted as the u and v directions. Each row aligned along the u direction is referred to as a v-curve (because the v value is constant), and vice versa.

To apply a multiresolution filter to the patch, all u and v curves can be considered as independent curves to which the ordinary multiresolution algorithms can be applied. For instance, to decompose a grid of vertices, the REDUCE-RESOLUTION algorithm could be called for all rows in the grid, and then for all columns in the smaller grid that results from reducing the resolution of all rows.

As discussed in the previous section, we can interpret a set of point samples as defining an open (non-periodic) or closed (periodic) curve. With

a tensor product surface, there are three unique ways to interpret the point grid.

7.2.1. *Open-Open Surfaces*

In an open-open tensor product surface, both the u and v curves are considered to be open curves. Open-open surfaces are isomorphic to a bounded plane (sheet). See Fig. 3(a).

7.2.2. *Open-Closed Surfaces*

We can treat a tensor-product surface as a set of open curves in one direction, and a set of closed curves in the other. In this case, the surface is isomorphic to an uncapped cylinder (see Fig. 3(b)).

7.2.3. *Closed-Closed Surfaces*

The final configuration of the u and v curves in a tensor product surface is when both dimensions are closed or periodic. In this configuration, the surface will be isomorphic to a torus, as shown in Fig. 3(c).

7.3. *2D Images*

Conceptually, there is no need to distinguish between tensor product surfaces and 2D images. Each is a collection of samples (nD points and intensity values, respectively) arranged in a regular grid, and linear combinations are valid operations on each sample type. Multiresolution filters can be applied to 2D images just as with tensor product surfaces: by treating all rows, and then all columns, of the image as independent 1D sample vectors.

In practice, however, there are some subtle but important differences that should be accounted for. In an image, positionality is implied by the location of a pixel, rather than the content of the pixel. As well, multiresolution operations on images are usually employed for filtering purposes, so having boundary interpolation gives incongruous importance to the image boundary. Thus our typical approaches to handling boundaries – interpolation or periodicity – make little sense in the image domain.

7.3.1. *Symmetric Extension*

The wrapping of samples done in the periodic case is a particular case of a more general approach called *symmetric extension*. The goal of symmetric

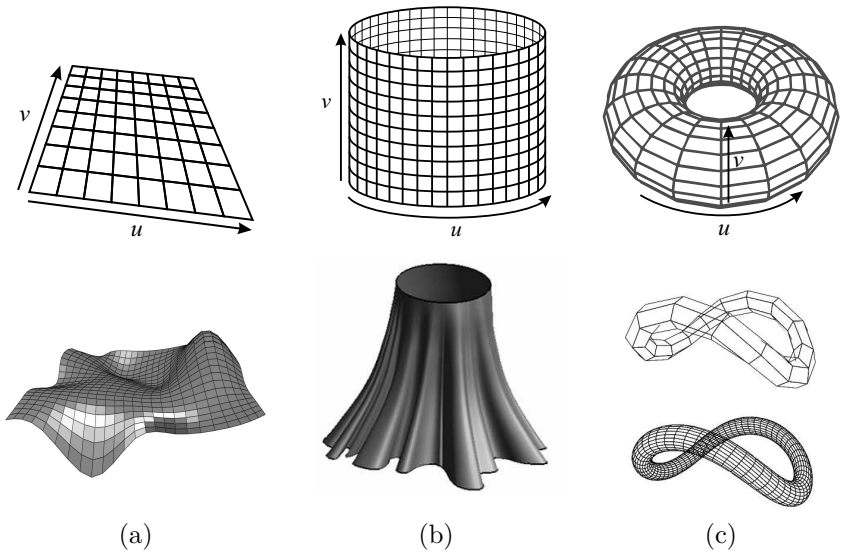

Fig. 3. There are three unique isomorphs for a tensor product surface, depending on how the u and v curves are interpreted: (a) open-open (bounded plane); (b) open-closed (uncapped cylinder); (c) closed-closed (torus).

extension is to avoid special boundary case evaluations by filling in sensible values for samples outside of the bounds of the real samples.

While wrapping may make sense for tileable images, in general it will not give a logical result because mixing the intensity values of the left and the right boundaries of the image is not reasonable. Due to the implied positionality of samples in images, the most natural "neighbor" sample when none exists would be the mirrored neighbor from the other side[12]. More formally, if $F = \{f_1, \ldots, f_m\}$, then we could set $f_{x<1} = f_{1+(1-x)}$ to mirror about the lower boundary, and similarly set $f_{x>m} = f_{m-(x-m)}$. See Fig. 4(top) for a diagrammatic representation of this type of symmetric extension, referred to as Type A.

An alternative approach is to mirror exactly about the boundary, which would produce duplicate entries of the first and last samples f_1 and f_m. In particular, we set $f_{x<1} = f_{1-x}$ to mirror about the lower boundary, and similarly set $f_{x>m} = f_{m-(x-m)+1}$. This is known as Type B symmetric extension; see the bottom image of Fig. 4.

The appropriate choice of symmetric extension depends on the multiresolution scheme. Consider cubic B-spline, with the filters given in (3.3).

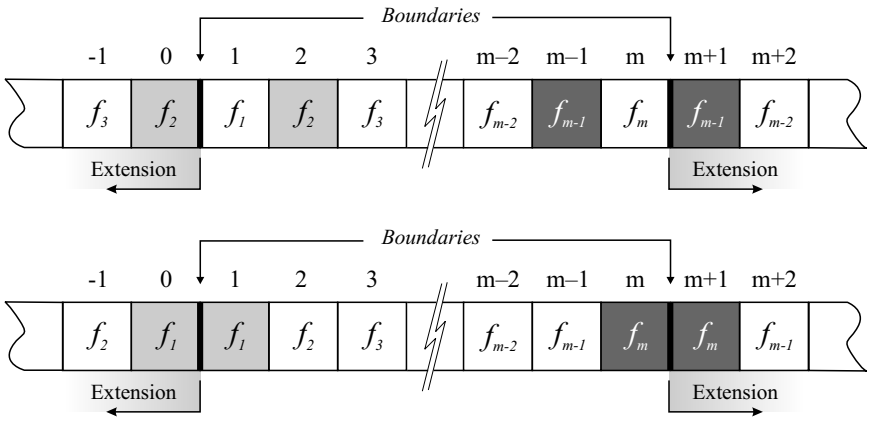

Fig. 4. Symmetric extension mirrors samples near the boundary. *Top:* mirrored about the first and last samples (Type A); *Bottom:* mirrored about the boundary (Type B).

Cubic B-spline is known as a *primal* or *edge-split* scheme, meaning that each coarse point and coarse edge has a corresponding fine point. For such schemes, Type A symmetric extension can be used for decomposing F. For reconstruction, the correct interpretation is achieved by using Type A for extending C and Type B for extending D (see the shaded entries in Fig. 5).

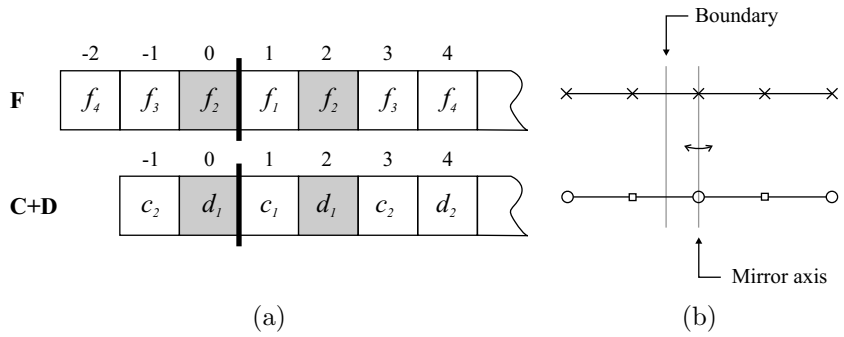

Fig. 5. Symettric extension for cubic B-spline. *Left:* Type A symmetric extension can be used when interpreting the samples in both decomposition and reconstruction, but Type B is required to interpret the details. *Right:* this is because the desired mirror axis is the same in both cases.

The Chaikin scheme is classified as a *dual* or *vertex-split* scheme, be-

cause each coarse sample is split into two fine samples, and there is no unambiguous relationship between points at each level. To use symmetric extension on such a scheme, we need both Type A and Type B. During decomposition, it is more natural to use Type B because the coarse vertex corresponding to the first fine vertex is split into a left and right component; see Fig. 6(b). During reconstruction, however, Type A extension for C and Type B extension for D provide the proper relationships.

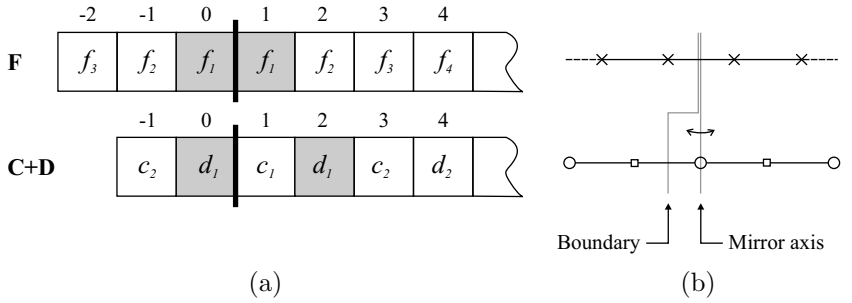

Fig. 6. Symettric extension for Chaikin multiresolution. *Left:* Type A and Type B symmetric extension are used in reconstruction (for the samples C and the details D, respectively), while Type B is used to decompose F. *Right:* this is because the desired mirror axis is different in each process.

7.4. *3D Images*

Multiresolution techniques for 3D images, such as volumetric data, naturally follow from 2D images. Consider the 3D image to have three axes: u, v, and w. Each w "curve" is actually an ordered set of samples (Fig. 7).

To apply multiresolution techniques to 3D image data, we can use the 2D image approach for each "slice" along the w direction, followed by each u-v curve aligned with w (highlighted in grey in Fig. 7).

As an example, consider a 3D image to be a set of still images from a video sequence. If the w axis quantifies time, then applying subdivision to each "slice" would smooth the image and subdividing each curve aligned with w would compute intermediate frames in the sequence. Conversely, decomposing a set of still images would reduce the resolution of each frame, while also removing every other frame.

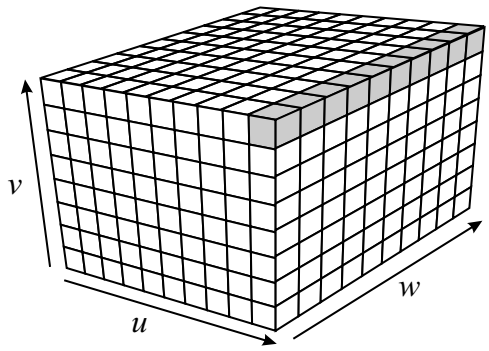

Fig. 7. Multiresolution techniques for 3D images are similar to those for 2D images: the appropriate filter is applied along each dimension independently.

8. Results, Examples and Applications

Multiresolution provides a tool for decomposing data into a hierarchy of components with different scales or resolutions. This hierarchy can be used for noise removal, compression, synthesizing and recognition of the objects[13,18,21]. In particular, multiresolution of smooth scalings and wavelets can nicely be used for smooth objects. Here we show some examples and applications of the quadratic and cubic B-spline multiresolution filters. We discuss iris synthesis and real-time visualization of volumetric data in detail.

8.1. *Example Applications*

Curves by example. We can use multiresolution filters on curves, such as those representing artistic silhouettes and line hand-gesture styles. Using analysis filters, we can extract styles (based on the characteristic details) from the curves, and these styles can then be applied to new base curves. Figure 8 shows an application of this, based on the interpolating quadratic B-spline wavelets from Sec. 5.3. This example illustrates the use of multiresolution for the common biometric task of feature extraction.

Removing noise from Curves. If we reconstruct a data set without using any (or using only a small portion) of the details, a simple de-noising is achieved as demonstrated in Fig. 9. In this example the cubic B-Spline filters of Sec. 5.1 have been used.

Image Compression. After decomposing an image to a low-resolution approximation and corresponding details D^i, we can lossily compress the

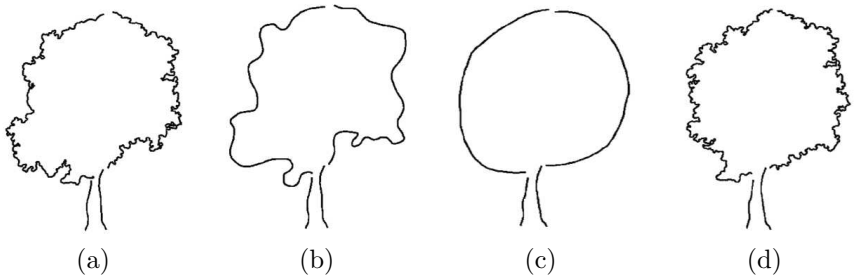

Fig. 8. Capturing stroke styles: (a) the original curve. (b) the curve from (a), reconstructed without D^i. (c) A new base curve. (d) The reconstructed curve with the (c) as the base curve and details D^i from (a).

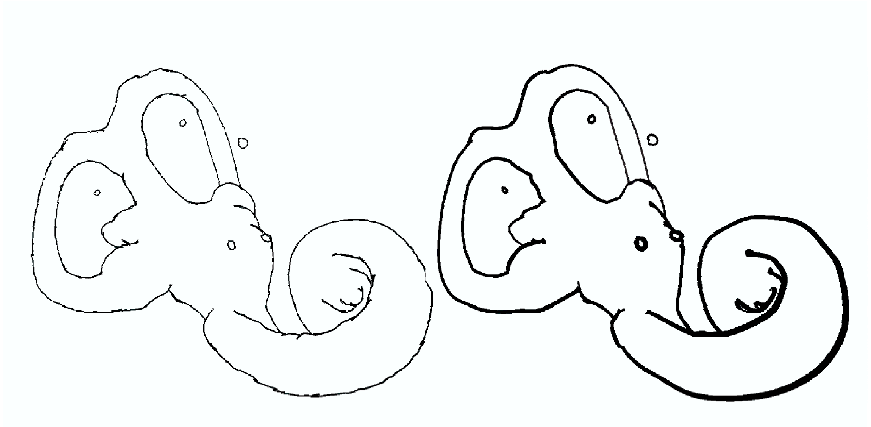

Fig. 9. Left: original silhouette from an inner-ear mesh. Right: the silhouette is denoised after decomposing twice with cubic B-spline filters and then reconstructing with partial details.

image by removing small magnitude details. This is one major step in current image compression techniques such as JPEG. We have compared our filters with Haar filters. All of our filters reported in Sec. 4 and (3.3) have better compression rates. We have also compared our filters with more successful image compression filters, D9/7 and D4, that have been used in JPEG2000[11]. Although our local filters are based on smooth scalings and wavelets (in contrast to D9/7 and D4), the resulting compression rate is comparable; see Fig. 10. In this comparison, we have removed the same amount of the details and compared the PSNR of the reconstructed im-

ages; higher PSNR indicates better detail retention. In particular, the wide quadratic B-spline filters given in (4.3) perform very close to the D9/7 filter.

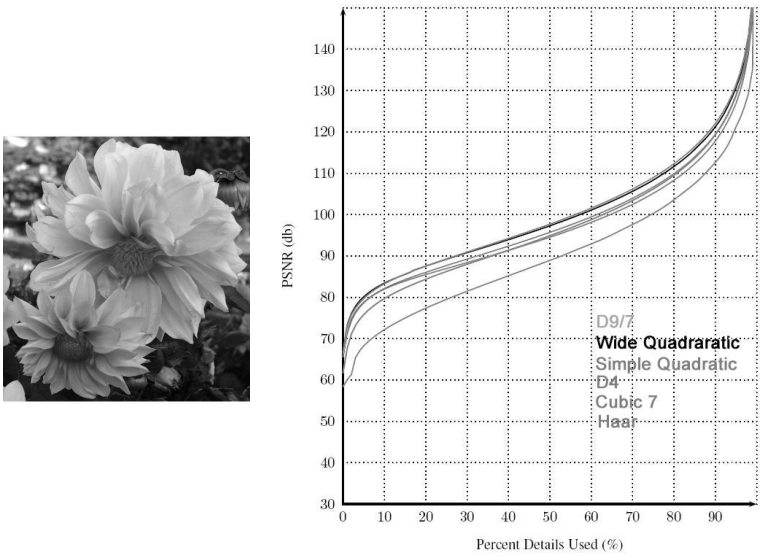

Fig. 10. Comparison of quadratic B-spline wavelets image compression with established techniques: (a) a sample image containing high-frequency data; (b) the resulting compression rates for various filters.

Terrain by Example. Terrain is typically represented with a height map (a regular grid of elevation data), which directly maps to an open-open tensor-product surface. Multiresolution helps us to use an existing terrain to synthesize new terrain by transferring the characteristic details[4]. For terrain synthesis, we can capture the details of a high-resolution target terrain and use them for a smooth base terrain to add realistic and more predictable noise. Figure 11 shows the application of the quadratic filters of (4.2) to terrain synthesis.

8.2. *Iris Synthesis*

Biometrics conventionally involves the analysis of biometric data for identification purposes. Due to logistical and privacy issues with collecting and organizing large amounts of biometric data, a new direction of biometric research concentrates on the synthesis of biometric information. One of the

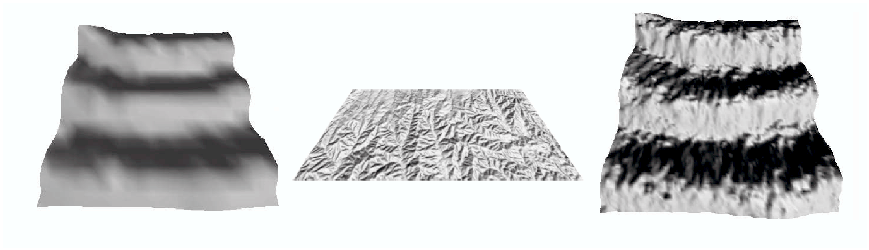

Fig. 11. Left: A smooth base terrain. Middle: a model terrain with high-frequency details. Right: the synthesized terrain has low-frequency characteristics of the base terrain and high-frequency traits of the model terrain.

primary goals of the synthesis of biometric data is to provide databases upon which biometric algorithms can be tested[25]. Along this direction, Wecker et al. employ the quadratic B-spline filters of (4.2) to augment existing iris databases with synthesized iris images[22]. Looking at any iris image, it is apparent that most of its characteristics are made of high frequency data. In addition, underlying sweep structures seems to be smooth (circular curves). Therefore, using smooth multiresolution filters for capturing these details is a promising technique. In this method, extracted details from different irises are combined to generate new irises. In fact, because we just combine portions of real irises, realistic iris are obtained. Because of the circular structure of iris, it is better to transform the iris images into polar coordinates, as shown in Fig. 12. For a 256x256 iris image, four levels of decomposition was found to produce good experimental results. Given a database of N input images, we decompose each of these images into their five components (four levels of details, and the very coarse approximation). Therefore, when synthesizing an iris image, we have N choices available for each of the necessary components which allows us to create a total of N^5 possible combinations. The original N iris images are included in these new combinations, as they are recreated when each of the selected components is from the same iris image. Clearly, given even a reasonable small database to start with, this method can generate an exponential increase in size. Some of these combinations may not result in good irises and extra conditions should also be taken to account[22].

Figure 13 shows a sample iris database, and Fig. 14 presents some of the iris images synthesized from the originals.

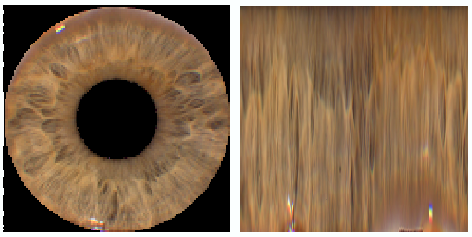

Fig. 12. A polar transform is used to unwrap the iris image.

Fig. 13. A sample database of iris images.

8.3. *Real-Time Contextual Close-up of Volumetric Data*

Medical image data sets can be very large. For instance, a CT scan of the head can have spatial dimensions up to $512x512x256$ voxels, while an MRI dataset can be as large as 256^3 voxels. Although many graphics display devices are now equipped with enough memory to store the entire volume, few commodity devices are actually capable of rendering the full resolution volume with the desired quality at interactive rates. Taerum et al.[20] propose a method based on B-spline filters that allow the data to be viewed and manipulated in real-time, while still allowing for full resolution viewing and detailed high resolution exploration of the data. In this method, the three different resolutions of the data-set are used. The lowest resolution is used during user interaction to maintain real-time feedback (Fig. 15). The

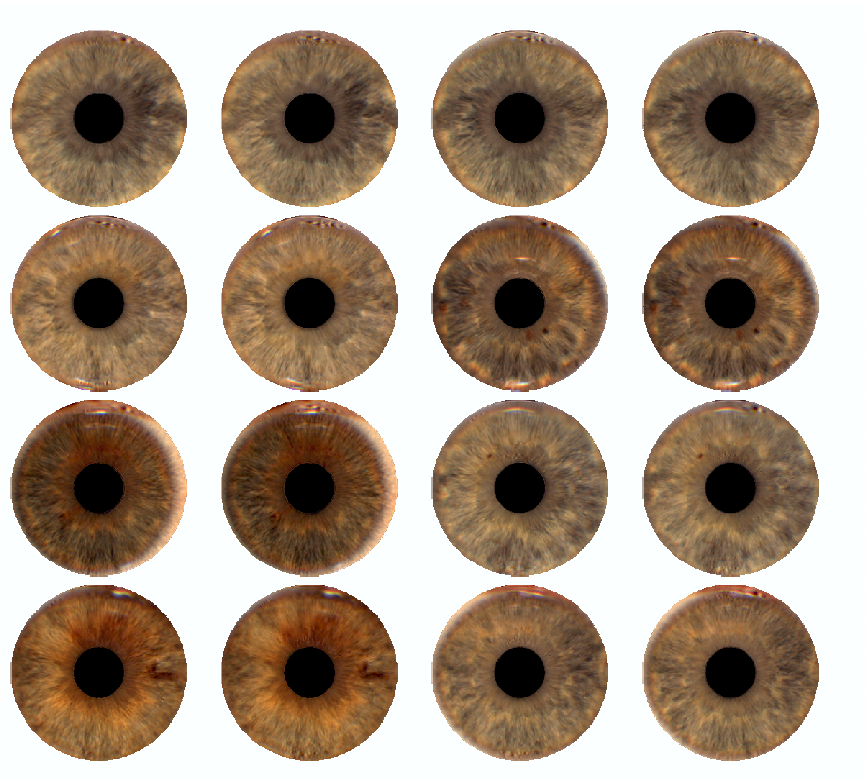

Fig. 14. Some synthetic iris images generated by combining elements of the original iris images from Fig. 13.

medium resolution is considered for presenting the overall context, while a "super-resolution" is used when rendering the contents of a user-defined lens, as demonstrated in Fig. 16. The method is based on the representation of volumetric data set as described in Sec. 7.4. Both quadratic (4.2) and cubic B-spline filters (3.3) have been used in this work.

9. Conclusion

We have presented several local multiresolution filters that can be employed in biometric applications involving discrete samplings of smooth data. Unlike Haar wavelets, these filters satisfy either the first or second level of smoothness. The biorthogonal construction also produces compact filters than semiorthogonal constructions, leading to more efficient algorithms.

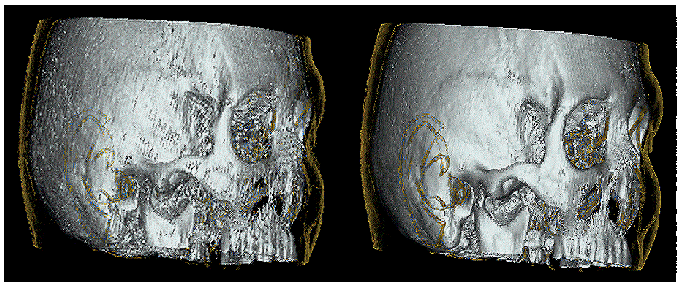

Fig. 15. Rendering CT scan data at multiple resolutions allows for real-time interactions while preserving visual clarity. *Left*: the lowest resolution is used for real-time interaction. *Right*: a medium resolution is shown when the data is not being manipulated.

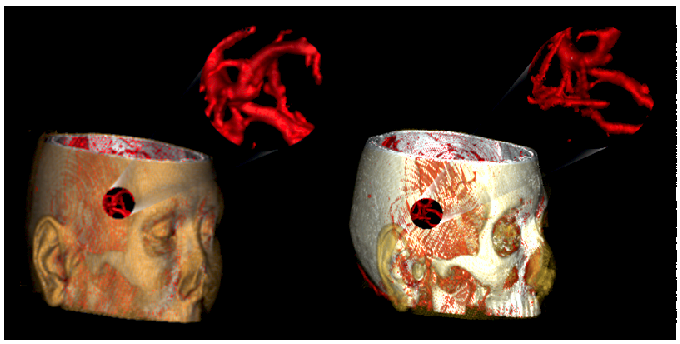

Fig. 16. The highest resolution of the CT scan data is used for generating contextual close-up views of the data.

We also consider several situations for the boundary, including interpolating and symmetric extension. With these considerations, the filters can be employed on many types of data, such as open and closed curves and surface patches, images, and volumes.

Finally, we discussed several applications of these filters to biometric-related problems, such as de-noising, feature (characteristic) extraction and transfer, data synthesis, and visualization.

10. Acknowledgement

This work has been supported by the National Science and Engineering Research Council (NSERC) of Canada. The authors would like to thank Reza Pakdel, Torin Taerum, Lakin Wecker, Katayoon Etemad, Meru Brunn

and John Brosz for their help and input to this work.

References

1. R. H. Bartels, J. Beatty, and B. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. 1987.
2. R. H. Bartels and F. F. Samavati. *Reversing subdivision rules: Local linear conditions and observations on inner products*. Journal of Computational and Applied Mathematics, 119(1–2):29–67, July 2000.
3. W. W. Boles and B. Boashash. *A Human Identification Technique Using Images of the Iris and Wavelet Transform*. IEEE Trans. on Signal Processing, 46(4), 1998, pp.1185-1188.
4. J. Brosz, F. F. Samavati and M. C. Sousa. *Terrain Synthesis By-Example*. Proceedings of the first International Conference on Computer Graphics Theory and Applications 2006.
5. G. Chaikin. *An Algorithm for High Speed Curve Generation*. Comp. Graph. and Im. Proc. 3 (1974) 346–349.
6. C. Chui. *An Introduction to Wavelets*. Academic Press, 1992.
7. J. Daugman. How Iris Recognition Works. *IEEE Trans, CSVT* , Vol. 14, PP. 21-30, 2004.
8. A. Finkelstein and D. H. Salesin. *Multiresolution curves*. in A. Glassner, editor, Proceedings of SIGGRAPH '94 (ACM Press, New York, 1994) 261–268.
9. Y. Huang,S. Luo and E. Chen *An Efficient Iris Recognition System*. Proceedings of the IEEE in Machine Learning and Cybernetics, Vol. 1, PP. 450- 454, 2002.
10. A. K. Jain, A. Prabhakar, L. Hong, and S. Pankanti. *Filterbank-based Fingerprint Matching*. IEEE Transactions on Image Processing, Vol. 9, No.5, pp. 846-859, May 2000.
11. S. lawson and J. Zhu. *Image Compression using Wavelets and JPEG2000: A Tutorial*. Electronics and Communication Engineering Journal, Vol. 14, No.3, pp. 112-121, June 2002.
12. S. Li and W. Li *Shape-Adaptive Discrete Wavelet Transforms for Arbitrarily Shaped Visual Object Coding*. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 10, No. 5, 2000.
13. T. Li, Q. Li, S. Zhu and M. Ogihara *A survey on Wavelets Applications in Data Mining*. ACM SIGKDDExplorations Newsletter archive, Vol. 4 , No. 2, PP. 49-68, 2002.
14. S. Lim , K. Lee, O. Byeon, and T. Kim. *Recognition Through Improvement of Feature Vector and Classifier*. ETRI Journal, Vol. 23, Number 2, June 2001.
15. L. Piegl and W. Tiller. *The NURBS Book*. 2nd Edition. Springer, 1997.
16. F. F. Samavati and R. H. Bartels. *Multiresolution Curve and Surface Representation by Reversing Subdivision Rules*. Computer Graphics Forum, 18(2), PP. 97-119, June 1999.
17. F. F. Samavati and R. H. Bartels. *Diagrammatic Tools for Generating*

Biorthogonal Multiresolutions. Technical Report 2003-728-31, Computer Science Department, University of Calgary, 2003.

18. E. J. Stollnitz, T. D. DeRose, and D. H. Salesin. *Wavelets for Computer Graphics*. Morgan Kaufmann Publishers, 1996.
19. G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.
20. T. Taerum, M. C. Sousa, F. F. Samavati, S. Chan, and R. Mitchell *Real-Time Super Resolution Contextual Close-up of Clinical Volumetric Data*. To appear in Eurographics/IEEE-VGTC Symposium on Visualization (EuroVIS 2006), May 2006, Lisbon, Portugal.
21. M. Unser and A. Aldroubi *A Review of Wavelets in Biomedical Applications*. Proceedings of the IEEE, 1996, Vol. 84, No. 4, PP. 626–638.
22. L. Wecker, F. F. Samavati and M. Gavrilova *Iris Synthesis: A Reverse Subdivision Application*. Proceedings of Graphite 2005, in association with ACM SIGGRAPH, PP. 121-125, Dunedin, New Zealand, Novembers 2005.
23. R. Wildes and J. Asmuth *A system for Automated Iris Recognition*. Proceedings of the Second IEEE Workshop on Application of Computer Vision, pp. 121-128, 1994.
24. M. Yang, D. J. Kriegman, and N. Ahuja *Detecting Faces in Images: A Survey* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 1, PP. 34-58, JANUARY 2002
25. S. N. Yanushkevich, A. Stoica, V. P. Shmerko and D. V. Popel *Biometric Inverse Problems* CRC Press/Taylor & Francis Group, Boca Raton, FL, 2005.