Title
Generalized B-spline subdivision-surface wavelets for geometry compression

Authors
Bertram, Martin
Duchaineau, Mark A.
Hamann, Bernd
et al.

Publication Date
2004

Peer reviewed

Generalized B-spline Subdivision-surface Wavelets for Geometry Compression

Martin Bertram[1] Mark A. Duchaineau[2] Bernd Hamann[3]
Kenneth I. Joy[3]

Keywords: arbitrary-topology meshes, biorthogonal wavelets, geometry compression, multiresolution methods, subdivision surfaces.

Abstract: We present a new construction of lifted biorthogonal wavelets on surfaces of arbitrary two-manifold topology for compression and multiresolution representation. Our method combines three approaches: subdivision surfaces of arbitrary topology, B-spline wavelets, and the lifting scheme for biorthogonal wavelet construction. The simple building blocks of our wavelet transform are local lifting operations performed on polygonal meshes with subdivision hierarchy. Starting with a coarse, irregular polyhedral base mesh, our transform creates a subdivision hierarchy of meshes converging to a smooth limit surface. At every subdivision level, geometric detail can be expanded from wavelet coefficients and added to the surface. We present wavelet constructions for bilinear, bicubic, and biquintic B-Spline subdivision. While the bilinear and bicubic constructions perform well in numerical experiments, the biquintic construction turns out to be unstable. For lossless compression, our transform can be computed in integer arithmetic, mapping integer coordinates of control points to integer wavelet coefficients. Our approach provides a highly efficient and progressive representation for complex geometries of arbitrary topology.

1 Introduction

Efficiently representing two-manifold geometries, like isosurfaces of trivariate functions, high-precision CAD models of arbitrary genus, and large-scale digital surfaces [27], is an important task in geometric modeling and scientific visualization. Multiresolution surface representations need to provide efficient access to local geometry satisfying user-defined bounds on error or complexity for compression, progressive transmission, and real-time visualization applications.

Biorthogonal wavelet representations [12, 42] are among the most efficient multiresolution

[1]Department of Computer Science, University of Kaiserslautern.
[2]Center for Applied Scientific Computing (CASC), Lawrence Livermore National Laboratory.
[3]Center for Image Processing and Integrated Computing (CIPIC), University of California in Davis.

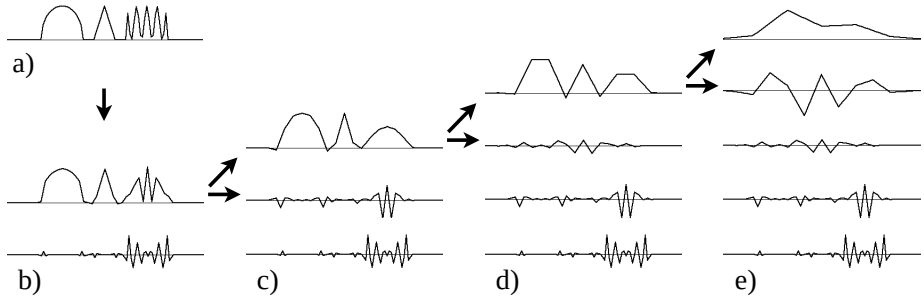

Figure 1: Decomposition steps of a DWT. The function a) is transformed by successive decomposition steps b),...,e), using a linear B-spline wavelet.

methods. The (biorthogonal) discrete wavelet transform (DWT) is often used in compression schemes for digital images, terrain models, and volume data, providing sparse data representations. Regularly-sampled data is transformed and reconstructed in linear computation time. The lifting scheme [43] for biorthogonal wavelets provides a simple construction and a maximum of efficiency of the transform, using highly localized digital filters. Many lifted construction approaches, including our method, allow the use of integer arithmetic for lossless compression. Compression is obtained, for example, by arithmetic encoding [34] of integer coefficients that are sparse or have small absolute values.

The DWT decomposes a function successively into certain frequency bands representing *details* of a function at different levels of resolution, see figure 1. There exist a variety of band-pass and low-pass filters with corresponding basis functions (wavelets and scaling functions) defining a DWT. Desired properties of the underlying basis functions are compact support, smoothness, symmetry, and orthogonality, which are, unfortunately, conflicting goals [12]. More details about wavelets and their construction can be found in the literature [10, 12, 33, 42, 15].

In this work, we present a construction of wavelets based on bilinear, bicubic, and biquintic subdivision surfaces providing linear computation time for wavelet decomposition and reconstruction. Numerical examples suggest stability of the bilinear and bicubic construction, while the decomposition of the biquintic construction is numerically unstable. The subdivi-

sion surfaces provided by our bicubic construction (using zero wavelet coefficients) coincide with multi-linear cell averaging (MCLA) [1], a variant of *Catmull-Clark* subdivision [7, 32, 40]. We believe that quadrilateral meshes with subdivision connectivity are a good alternative to triangle meshes, especially when representing surfaces that behave differently in two canonical directions. Quadrilateral meshes are often used, for example, in character animation [13], where the skin of an animated character is tied to a directed sceleton. Like Catmull-Clark surfaces, our surfaces can be converted into non-uniform rational B-splines (NURBS) patches [36] at any level of detail for use in CAD/CAM applications.

This paper is structured as follows: Secrion 2, contains a summary of related work. We introduce the one-dimensional lifting operations for the construction of symmetric wavelets with associated B-spline scaling functions in section 3. These lifting operations are generalized to polyhedral meshes in section 4 such that the one-dimensional wavelet constructions define tensor products on a regular, rectilinear grid. In our approach, these meshes may contain *extraordinary vertices*, *i.e.*, vertices with valence different from four, allowing the construction of arbitrary shapes. In section 5, we present a lossless geometry-compression algorithm and provide numerical results for our wavelet construction.

2 Related Work

Wavelets representing surfaces of arbitrary topology were originally explored by Lounsbery *et al.* [29, 30]. Starting with a subdivision surface scheme, like Catmull-Clark [7] or Loop [31] subdivision, wavelet transforms have been constructed using the recursively generated basis functions as scaling functions. Lounsbery [29] showed that the function spaces generated by subdivision rules are nested and that compactly-supported wavelets spanning complements of these spaces (within the next finer-resolution spaces) can be constructed. These wavelet constructions for smooth, non-interpolating subdivision surfaces, like Catmull-Clark and Loop

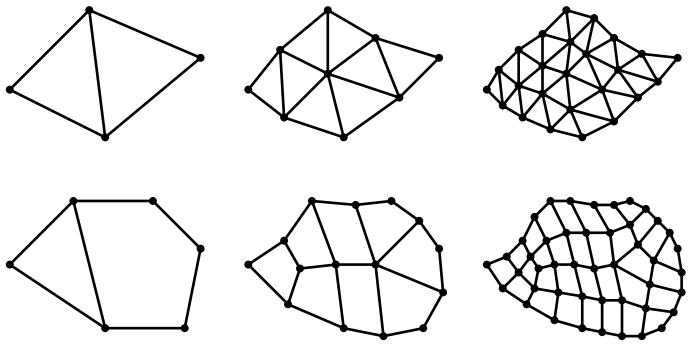

Figure 2: Meshes with regular subdivision connectivity. Loop subdivision and most subdivision-surface wavelet approaches use triangular mesh refinement (top). Our approach uses the connectivity of Catmull-Clark subdivision (bottom).

subdivision, have the disadvantage that the transform is based on a global system of equations. Only the inverse transform is generally computed in linear time based on local operations.

Other subdivision-surface wavelet constructions for functions defined on triangulated spherical domains were introduced by Schröder and Sweldens [38], Nielson *et al.* [35], and Bonneau [4, 5]. Their approaches can be extended to more general than spherical domains, but they are used for constructing functions on given domains rather than representing the underlying domain geometries. In contrast, our approach is capable of representing two-manifold geometries as well as functions defined on these.

Piecewise linear subdivision-surface wavelets defined on triangular meshes with regular refinement are often used for multiresolution representation and rendering of surfaces [16, 9, 18], and for solving partial differential equations (PDE's) [41]. An important problem is the generation of meshes with *subdivision connectivity*, *i.e.*, meshes that can be constructed by regular refinement of coarse base meshes, as shown in figure 2. Algorithms computing re-parametrizations of triangle meshes by mapping them into meshes with subdivision connectivity have recently been described [24, 16, 17, 28]. An algorithm converting triangulated surface models into meshes with subdivision connectivity is known as *multiresolution adaptive parametrization of surfaces* (MAPS) [26]. We have presented a meshing algorithm for isosurfaces providing meshes

with quadrilateral subdivision hierarchy that serve as input for our wavelet transform [2, 3].

More general multiresolution approaches for completely irregular mesh hierarchies without subdivision connectivity often rely on the same principles as wavelet transforms [21, 25]. Subdivision-surface wavelets with regular refinement, however, do not need to store any parameter or connectivity information, except for a coarse-resolution base mesh. Thus, most effective state-of-the-art compression algorithms rely on meshes with subdivision connectivity [22]. It is possible to generate mesh hierarchies with normal displacement, such that coordinates for every vertex can be reconstructed from a scalar-valued offset [20].

3 Symmetric Lifted Wavelets

Wavelet lifting was introduced by Sweldens [43], and it is often used for biorthogonal wavelet construction [23]. The lifting scheme subdivides the computation for a single filtering step of the DWT into a sequence of smaller filtering operations. Lifting increases the efficiency, simplifies the construction, and makes the use of integer arithmetic feasible [6]. In this section, we introduce a construction of lifted, one-dimensional wavelets that are generalized to polyhedral mesh domains in section 4. We first review some basics about the DWT.

3.1 Discrete Wavelet Transform

The DWT is a basis transform between certain spaces spanned by dilated and translated versions of a wavelet ψ and a scaling function ϕ:

$$\psi_i^j(x) \;=\; \psi(2^j x - i) \quad \text{and} \quad \phi_i^j(x) \;=\; \phi(2^j x - i). \tag{3.1}$$

A function f is initially represented in a basis of scaling functions at a high level of resolution, denoted by the index $j_\varepsilon > 0$:

$$f(x) \;=\; \sum_{i \in \mathbb{Z}} s_i^{j_\varepsilon}\; \phi_i^{j_\varepsilon}(x). \tag{3.2}$$

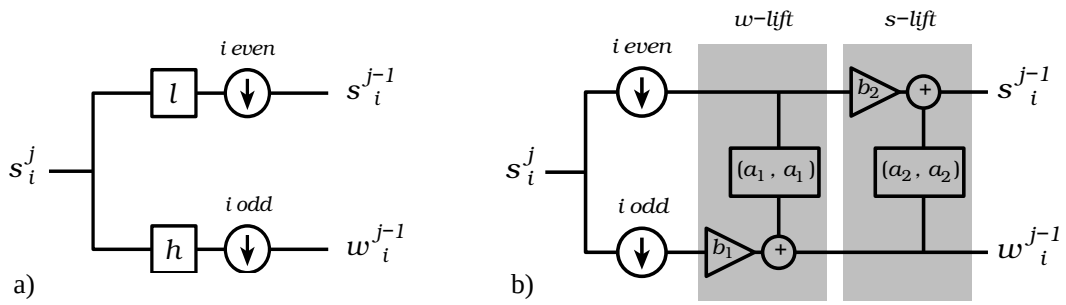

Figure 3: DWT viewed as a signal-processing algorithm. a) High- and low-frequency bands are separated by discrete filtering with sequences h and l and down-sampling. b) Lifting scheme: Convolutions with h and l are replaced by lifting operations.

A simple basis transform decomposes this representation into a high-frequency part, based on wavelets, and a low-frequency part, based on coarser scaling functions: at level $j_\varepsilon - 1$,

$$f(x) \;=\; \sum_{i\in\mathbb{Z}} w_i^{j_\varepsilon-1}\; \psi_i^{j_\varepsilon-1}(x) \;+\; \sum_{i\in\mathbb{Z}} s_i^{j_\varepsilon-1}\; \phi_i^{j_\varepsilon-1}(x). \tag{3.3}$$

This transform is called *decomposition* or *analysis*. Decomposition steps are recursively applied to the part represented by scaling functions until a base level $j = 0$ is reached. The function f is finally represented as

$$f(x) \;=\; \sum_{j=0}^{j_\varepsilon-1} \sum_{i\in\mathbb{Z}} w_i^j\; \psi_i^j(x) \;+\; \sum_{i\in\mathbb{Z}} s_i^0\; \phi_i^0(x). \tag{3.4}$$

figure 1. illustrates this basis transform.

A decomposition step is implemented by a discrete filtering with sequences h and l that transform scaling-function coefficients s_i^j at level j into scaling-function coefficients s_i^{j-1} and wavelet coefficients w_i^{j-1} at level $j-1$, see figure 3 a). The decomposition rules are defined as

$$s_i^{j-1} \;=\; \sum_{k\in\mathbb{Z}} l_{k-2i}\; s_k^j \quad \text{and} \tag{3.5}$$

$$w_i^{j-1} \;=\; \sum_{k\in\mathbb{Z}} h_{k-2i}\; s_k^j. \tag{3.6}$$

For the inverse DWT, every individual decomposition step is inverted by a *reconstruction* (or *synthesis*) step using filters $\tilde{h}$ and $\tilde{l}$,

$$s_i^j \;=\; \sum_{k\in\mathbb{Z}} \left(\tilde{l}_{i-2k}\; s_k^{j-1} \;+\; \tilde{h}_{i-2k}\; w_k^{j-1}\right). \tag{3.7}$$

Applying reconstruction steps in reverse order of the corresponding decomposition steps reproduces the initial representation defined by scaling functions, equation (3.2).

The time complexity for a decomposition step with n non-zero scaling-function coefficients is $O(n)$, provided that the filters h and l have finite length. Since the number of scaling-function coefficients is cut into half for every level of the transform, the total complexity for the DWT is $O\left(n + \frac{n}{2} + \frac{n}{4} + \ldots\right) = O(n)$.

3.2 Lifting Approach

Rather than computing the coefficients based on equations (3.5–3.7), we subdivide these summation steps into simple *lifting operations*, reducing the length of the discrete filters and thus decreasing the number of floating-point operations, see figure 3 b). The theory of lifting is due to Sweldens [43], related to an earlier approach by Dahmen [11].

In the following, we define our lifting operations using algorithmic notation. A decomposition step for the DWT is computed by re-labeling coefficients,

$$s_i^{j-1} \leftarrow s_{2i}^j \quad \text{and} \quad w_i^{j-1} \leftarrow s_{2i+1}^j, \tag{3.8}$$

followed by a sequence of alternating *s-lift* and *w-lift* operations. These operations modify one coefficient at a time, depending on its own and its two neighbors' values. The s-lift and w-lift operations are defined as

s-lift(a, b)**:**

$$s_i^{j-1} \leftarrow a w_{i-1}^{j-1} + b s_i^{j-1} + a w_i^{j-1} \quad \forall i \quad \text{and} \tag{3.9}$$

w-lift(a, b)**:**

$$w_i^{j-1} \leftarrow a s_i^{j-1} + b w_i^{j-1} + a s_{i+1}^{j-1} \quad \forall i. \tag{3.10}$$

An example for this lifting scheme is depicted in figure 4. Both filters h and l are constructed

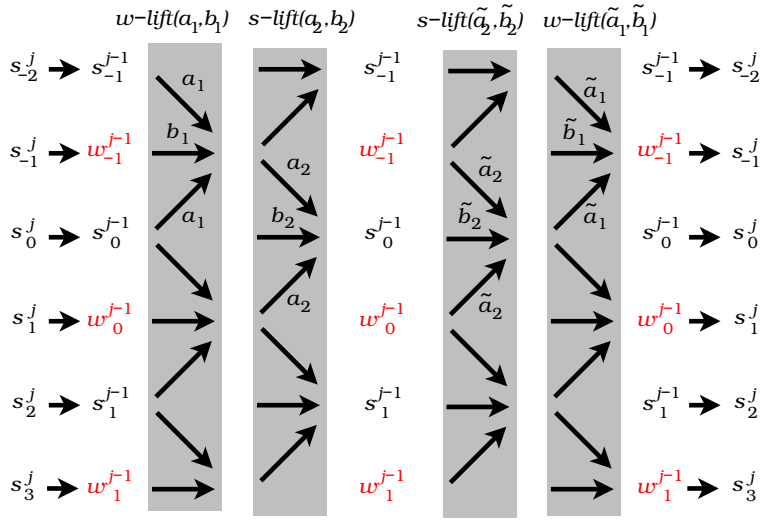

Figure 4: Lifting scheme for linear B-spline wavelet. Decomposition (left) and reconstruction (right) are composed of one s-lift and one w-lift operation. The lifting parameters satisfy the relations $\tilde{a}_i = -\frac{a_i}{b_i}$ and $\tilde{b}_i = \frac{1}{b_i}$.

simultaneously by a sequence of lifting operations. By using only s-lift and w-lift operations, we restrict the class of wavelets that can be constructed, but we also reduce the number of operations required for the transform. As we show in section 4, corresponding lifting operations can be constructed for subdivided polygon meshes. Another advantage of this construction is the fact that every lifting step is inverted by the same type of lifting operation, replacing a by $\tilde{a} = -\frac{a}{b}$ and b by $\tilde{b} = \frac{1}{b}$, where b must be non-zero. Hence, we obtain an efficient algorithm for the inverse transform by applying the inverse of every single lifting operation in reverse order.

3.3 B-spline Wavelet Construction

We choose B-splines as scaling functions due to their wide use and applicability in CAGD and approximation theory. Considering the dyadic refinement process for B-splines [8], the reconstruction filter $\tilde{l}$ is defined by the *two-scale relation*

$$\phi(x) \;=\; \sum_{i\in\mathbb{Z}} \tilde{l}_i \; \phi(2x-i). \tag{3.11}$$

The non-zero values for $\tilde{l}_i$ can be obtained from Pascal's triangle by dividing the entries in the $(n+2)$th row by 2^n, where n is the polynomial degree, see table 2. Using w-lift and s-lift

operations, we can construct symmetric filters with odd numbers of non-zero entries, like filters $\tilde{l}$ for B-splines with odd polynomial degrees. Even degrees are not considered, since this would require a dual mesh construction, according to Doo-Sabin subdivision [14].

Analogously to the two-scale relation for scaling functions, every wavelet can be represented as a linear combination of finer-level scaling functions,

$$\psi(x) \;=\; \sum_{i\in\mathbb{Z}} \tilde{h}_i \; \phi(2x-i). \tag{3.12}$$

To improve the approximation properties of our transform, we want to construct wavelets that have at least two vanishing moments [42]. A wavelet has n vanishing moments when its convolutions with n polynomials $(1,\; x,\; x^2, \cdots, x^{n-1})$ are zero. The first moment of a wavelet is zero if and only if the corresponding filter $\tilde{h}$ satisfies the condition

$$\sum_{i\in\mathbb{Z}} \tilde{h}_i \;=\; 0. \tag{3.13}$$

For our lifting approach, the second moment vanishes, due to symmetry.

3.3.1 Linear B-spline Wavelets

We start with constructing the inverse DWT (reconstruction) defined by filters $\tilde{h}$ and $\tilde{l}$, since $\tilde{l}$ is already determined by the choice of scaling functions. One single w-lift operation is required to compute a convolution with $\tilde{l}$. To satisfy equation (3.13), an additional and necessary $s-lift$ operation is computed first (otherwise it would modify $\tilde{l}$). These two lifting operations define the reconstruction scheme, and the corresponding inverse lifting operations define the decomposition scheme for our DWT, see figure 4.

Since convolutions with $\tilde{h}$ and $\tilde{l}$ are computed simultaneously by the same lifting operations, the construction of $\tilde{h}$ and $\tilde{l}$ is constrained by the lifting parameters $\tilde{a}_1, \tilde{b}_1, \tilde{a}_2$, and $\tilde{b}_2$, as illustrated in figure 5 a). The filter $\tilde{l}$ is constrained by

$$\tilde{l}_0 \;=\; \tilde{b}_2 \quad \text{and} \quad \tilde{l}_1 \;=\; \tilde{b}_2\tilde{a}_1. \tag{3.14}$$

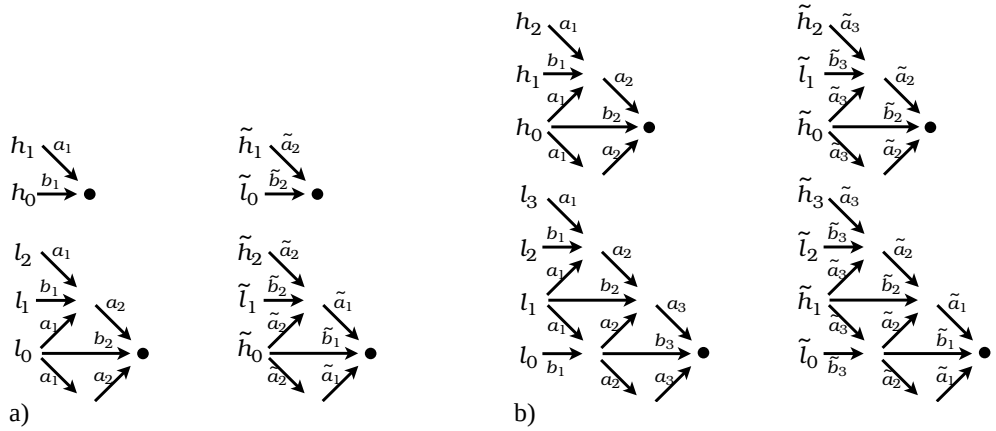

Figure 5: Constraints for construction of filters h, l, $\tilde{h}$, and $\tilde{l}$ using a) linear and b) cubic B-spline scaling functions. Every path of arrows corresponds to a summation term whose factors are represented by the individual arrows.

Using for $\tilde{l}$ the values from table 2, we obtain

$$\tilde{a}_1 \;=\; \tfrac{1}{2} \quad \text{and} \quad \tilde{b}_2 \;=\; 1. \tag{3.15}$$

The constraints for $\tilde{h}$ are given by

$$\begin{aligned} \tilde{h}_0 \;&=\; 2\tilde{a}_2\tilde{a}_1 \;+\; \tilde{b}_1 \;=\; \tilde{a}_2 \;+\; \tilde{b}_1, \\ \tilde{h}_1 \;&=\; \tilde{a}_2, \quad \text{and} \\ \tilde{h}_2 \;&=\; \tilde{a}_2\tilde{a}_1 \;=\; \tfrac{1}{2}\tilde{a}_2. \end{aligned} \tag{3.16}$$

We note that $\tilde{l}$ and $\tilde{h}$ are symmetric and that the coefficients with negative indices do not produce additional constraints. Hence, equation (3.13) becomes

$$\tilde{h}_0 \;+\; 2\tilde{h}_1 \;+\; 2\tilde{h}_2 \;=\; 4\tilde{a}_2 \;+\; \tilde{b}_1 \;=\; 0. \tag{3.17}$$

This implies that $\tilde{a}_2$ and $\tilde{b}_1$ are proportional. Since either $\tilde{a}_2$ or $\tilde{b}_1$ appears on every right-hand term of equations (3.16), the filter $\tilde{h}$ is already determined, except for a scaling factor. By choosing $\tilde{b}_1 = 1$, we obtain the lifting parameters shown in table 1. The remaining filters h and l can be derived from these lifting parameters. They are summarized in table 2.

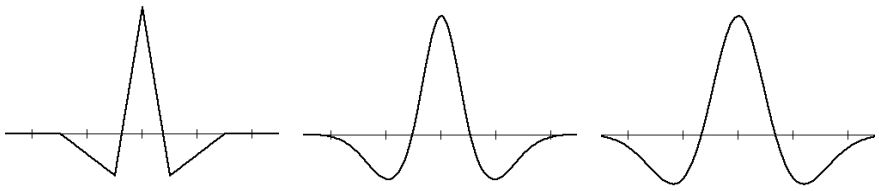

Figure 6: Linear, cubic, and quintic B-spline wavelets.

degree	a_1	b_1	a_2	b_2	a_3	b_3	a_4	b_4
linear	w-lift($-\frac{1}{2}$,	1)	s-lift($\frac{1}{4}$,	1)				
cubic	s-lift($-\frac{1}{4}$,	1)	w-lift(-1,	1)	s-lift($\frac{3}{8}$,	2)		
quintic	w-lift($-\frac{1}{6}$,	1)	s-lift($-\frac{9}{16}$,	1)	w-lift($-\frac{4}{3}$,	1)	s-lift($\frac{5}{8}$,	4)

degree	$\tilde{a}_4$	$\tilde{b}_4$	$\tilde{a}_3$	$\tilde{b}_3$	$\tilde{a}_2$	$\tilde{b}_2$	$\tilde{a}_1$	$\tilde{b}_1$
linear					s-lift($-\frac{1}{4}$,	1)	w-lift($\frac{1}{2}$,	1)
cubic			s-lift($-\frac{3}{16}$,	$\frac{1}{2}$)	w-lift(1,	1)	s-lift($\frac{1}{4}$,	1)
quintic	s-lift($-\frac{5}{32}$,	$\frac{1}{4}$)	w-lift($\frac{4}{3}$,	1)	s-lift($\frac{9}{16}$,	1)	w-lift($\frac{1}{6}$,	1)

Table 1: Lifting parameters for DWT and inverse DWT shown in order of computation.

3.3.2 Cubic and Quintic B-spline Wavelets

In analogy to linear B-spline wavelets, one can construct wavelets for cubic or quintic scaling functions using one or two additional lifting steps, respectively. In the cubic case, the filter $\tilde{l}$ requires at least one w-lift and one s-lift operation, due to its width. The vanishing-moment condition, equation (3.13), requires an additional s-lift computed first, see figure 5 b). The filter $\tilde{l}$ is constrained by

$$\tilde{l}_0 \;=\; \tilde{b}_3(\tilde{b}_1 \;+\; 2\tilde{a}_2\tilde{a}_1), \quad \tilde{l}_1 \;=\; \tilde{b}_3\tilde{a}_2 \quad \text{and} \quad \tilde{l}_2 \;=\; \tilde{b}_3\tilde{a}_2\tilde{a}_1. \tag{3.18}$$

The choice of filter $\tilde{l}$, shown in table 2, implies

$$\tilde{a}_1 \;=\; \tfrac{1}{4}, \quad \tilde{a}_2 \;=\; \tilde{b}_1, \quad \text{and} \quad \tilde{b}_3\tilde{b}_1 \;=\; \tfrac{1}{2}. \tag{3.19}$$

degree	h_0	$h_{\pm1}$	$h_{\pm2}$	$h_{\pm3}$	l_0	$l_{\pm1}$	$l_{\pm2}$	$l_{\pm3}$	$l_{\pm4}$
linear	1	$-\frac{1}{2}$			$\frac{3}{4}$	$\frac{1}{4}$	$-\frac{1}{8}$		
cubic	$\frac{3}{2}$	-1	$\frac{1}{4}$		$\frac{5}{4}$	$\frac{5}{32}$	$-\frac{3}{8}$	$\frac{3}{32}$	
quintic	$\frac{5}{2}$	$-\frac{15}{8}$	$\frac{3}{4}$	$-\frac{1}{8}$	$\frac{231}{96}$	$-\frac{7}{32}$	$-\frac{21}{24}$	$\frac{15}{32}$	$-\frac{5}{64}$

degree	$\tilde{h}_0$	$\tilde{h}_{\pm1}$	$\tilde{h}_{\pm2}$	$\tilde{h}_{\pm3}$	$\tilde{h}_{\pm4}$	$\tilde{l}_0$	$\tilde{l}_{\pm1}$	$\tilde{l}_{\pm2}$	$\tilde{l}_{\pm3}$
linear	$\frac{3}{4}$	$-\frac{1}{4}$	$-\frac{1}{8}$			1	$\frac{1}{2}$		
cubic	$\frac{5}{8}$	$-\frac{5}{64}$	$-\frac{3}{16}$	$-\frac{3}{64}$		$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{8}$	
quintic	$\frac{77}{128}$	$\frac{7}{128}$	$-\frac{7}{32}$	$-\frac{15}{128}$	$-\frac{5}{256}$	$\frac{5}{8}$	$\frac{15}{32}$	$\frac{3}{16}$	$\frac{1}{32}$

Table 2: Filters for DWT and inverse DWT.

The equations for $\tilde{h}$, after eliminating $\tilde{a}_1$ and $\tilde{a}_2$, are given by

$$\begin{aligned} \tilde{h}_0 &= \tilde{b}_2 + 2\tilde{a}_3\tilde{b}_1, \\ \tilde{h}_1 &= \tfrac{1}{4}\tilde{b}_2 + \tfrac{7}{4}\tilde{a}_3\tilde{b}_1, \\ \tilde{h}_2 &= \tilde{a}_3\tilde{b}_1, \quad \text{and} \\ \tilde{h}_3 &= \tfrac{1}{4}\tilde{a}_3\tilde{b}_1. \end{aligned} \tag{3.20}$$

Using equation (3.13), we obtain

$$3\tilde{b}_2 = -16\tilde{a}_3\tilde{b}_1. \tag{3.21}$$

Again, we observe that the remaining lifting parameters do not modify $\tilde{h}$, except for scaling, since $\tilde{b}_2$ and $\tilde{a}_3\tilde{b}_1$ are proportional. Hence, we can choose $\tilde{b}_1 = \tilde{b}_2 = 1$ and uniquely determine the remaining parameters.

A similar construction based on four lifting operations is feasible for quintic B-spline subdivision. Again, all lifting parameters are uniquely determined from the vanishing-moment condition, except for scaling. These lifting parameters are listed in tables 1 and 2 and the corresponding wavelets are depicted in figure 6. Wavelets of high polynomial degree can be used to represent smooth surfaces at very high precision, but they are less efficient when processing noisy data sets. As we demonstrate in section 5, the low-pass filter of our quintic wavelet construction becomes unstable when successively applied to noisy data, due to the narrow support of wavelets. Therefore, we prefer the use of linear and cubic wavelet transforms. More general constructions are feasible, based on different scaling functions.

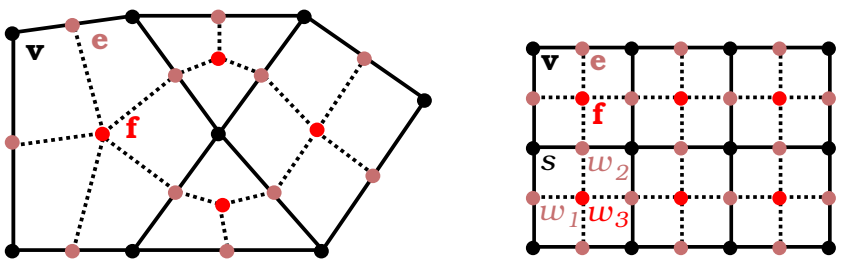

Figure 7: Topology of Catmull-Clark subdivision. **f** (red), **e** (pink), and **v** (black). a) Irregular mesh; b) regular mesh: Coefficients s and w_3 correspond to **v** and **f** vertices, respectively. Coefficients w_1 and w_2 correspond to **e** vertices of vertical and horizontal edges.

4 Generalization to Arbitrary Polyhedral Meshes

In this section, we construct s-lift and w-lift operations for polyhedral meshes, such that we exactly reproduce tensor-products of the corresponding one-dimensional operations when using a regular, rectilinear grid. Our one-dimensional wavelet constructions are thus generalized to arbitrary polygon meshes with subdivision connectivity.

4.1 Index-free Notation for Subdivision Rules

Subdivision surfaces are limit surfaces resulting from recursive refinement of polyhedral base meshes. A subdivision step refines a *submesh* to a *supermesh* by inserting vertices. The coordinates of all vertices of a supermesh are computed as linear combinations of local vertices in the submesh using the same masks for every level (linear, stationary schemes). Most subdivision schemes converge rapidly to a continuous limit surface. The mesh obtained from a few subdivisions is already a good approximation for surface rendering. Parametrization and exact evaluation of limit-surface points is feasible [40].

In our approach, we use the hierarchical mesh connectivity defined by Catmull-Clark subdivision [7], which is a generalization of uniform bicubic B-splines to arbitrary control meshes. A mesh is refined by inserting a new vertex inside every face and on every edge and by connecting these vertices to quadrilaterals, see figure 7 a). Vertices in a supermesh correspond to a face

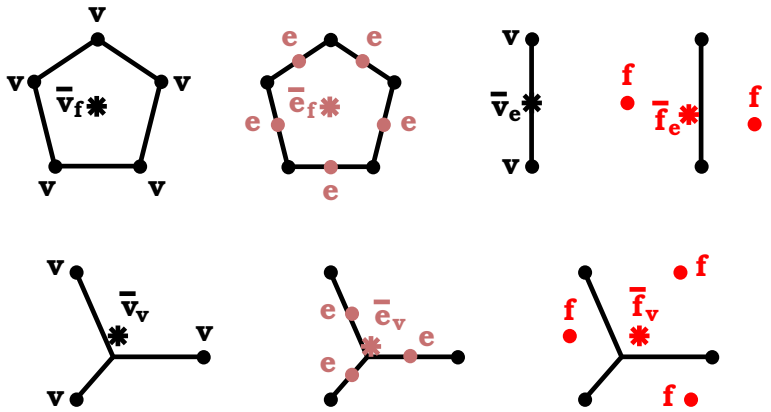

Figure 8: Examples for index-free notation. $\overline{\mathbf{v}}_\mathbf{f}$ denotes the centroid of a face, $\overline{\mathbf{e}}_\mathbf{f}$ the centroid of its associated $\mathbf{e}$ vertices, etc.

(polygon), an edge, or a vertex in the submesh and are denoted by $\mathbf{f}$, $\mathbf{e}$, and $\mathbf{v}$, respectively. We use this subdivision topology to construct generalized bilinear, bicubic, and biquintic scaling functions and wavelets. Subdivision schemes generating even-degree B-splines, like Doo-Sabin subdivision [14], typically use a dual mesh structure, which is incompatible with our symmetric lifting operations.

To describe subdivision rules determining new vertex positions, we introduce an index-free notation. Therefore, we use the averaging operator $\overline{k}_l$, where k and l can represent either $\mathbf{f}$, $\mathbf{e}$, or $\mathbf{v}$. This averaging operator returns the arithmetic average of all vertices of type k that are adjacent to l or that correspond to adjacent/incident faces and edges. In particular, we use the following notation that is illustrated in figure 8: $\overline{\mathbf{v}}_\mathbf{f}$: centroid of each face; $\overline{\mathbf{e}}_\mathbf{f}$: centroid of $\mathbf{e}$ vertices of each face; $\overline{\mathbf{v}}_\mathbf{e}$: midpoint of each edge; $\overline{\mathbf{f}}_\mathbf{e}$: midpoint of both adjacent $\mathbf{f}$ vertices of each edge; $\overline{\mathbf{v}}_\mathbf{v}$: centroid of all adjacent $\mathbf{v}$ vertices; $\overline{\mathbf{e}}_\mathbf{v}$: centroid of all $\mathbf{e}$ vertices of incident edges; $\overline{\mathbf{f}}_\mathbf{v}$: centroid of all $\mathbf{f}$ vertices of incident faces.

Catmull-Clark subdivision in index-free notation is defined by the rules

$$\begin{aligned} \mathbf{f}' &\leftarrow \overline{\mathbf{v}}_\mathbf{f}, \\ \mathbf{e}' &\leftarrow \tfrac{1}{2}\left(\overline{\mathbf{v}}_\mathbf{e} + \overline{\mathbf{f}'}_\mathbf{e}\right), \quad \text{and} \\ \mathbf{v}' &\leftarrow \tfrac{1}{n_\mathbf{v}}\left(\overline{\mathbf{f}'}_\mathbf{v} + \overline{\mathbf{v}}_\mathbf{v} + (n_\mathbf{v}-2)\mathbf{v}\right), \end{aligned} \tag{4.1}$$

where $n_{\mathbf{v}}$ is the valence (number of incident edges) of vertex $\mathbf{v}$. The order of vertex modifications is important, since the result of an operation may define the input for any of the subsequent operations. In the third modification step, all $\mathbf{v}$ vertices need to be duplicated, since they depend on each other. This kind of dependency is avoided in our lifting scheme, since it would result in a global system of equations for the inverse operation.

4.2 Generalized Lifting Operations

In the case of a tensor-product wavelet transform, we apply a decomposition step of the one-dimensional DWT to all rows and then columns of a data set. This results in sets of coefficients s, w_1, w_2, w_3 for four different types of basis functions, given by

$$\begin{aligned} \phi(x,y) &= \phi(x)\phi(y), \\ \psi_1(x,y) &= \phi(x)\psi(y), \\ \psi_2(x,y) &= \psi(x)\phi(y), \quad \text{and} \\ \psi_3(x,y) &= \psi(x)\psi(y), \end{aligned} \tag{4.2}$$

respectively.

A decomposition step can be considered as an operation applied to a rectilinear supermesh that computes $\mathbf{v}$ vertex positions for an approximating submesh and replaces the remaining $\mathbf{e}$ and $\mathbf{f}$ vertices by difference vectors representing details that are missing in the submesh. The $\mathbf{v}$ vertices represent coefficients s for scaling functions, $\mathbf{e}$ vertices represent wavelet coefficients w_1 and w_2 (depending on the orientation of edges), and $\mathbf{f}$ vertices represent wavelet coefficients w_3, see figure 7 b).

Rather than applying a one-dimensional decomposition step first to all rows and subsequently to all columns of a data set, we apply every individual lifting operation to the rows and columns, consecutively. The overall order of operations remains unchanged when considering only rows or only columns, and the resulting transform is the same. (A nice analogy is

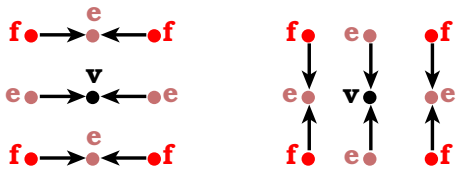

Figure 9: An s-lift operation applied to a rectilinear grid is composed of a vertical (left) and a horizontal (right) one-dimensional s-lift operation.

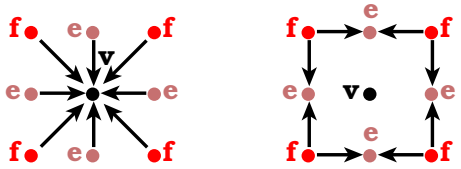

Figure 10: Same s-lift operation as shown in figure 9 with vertex modifications performed in different order. First, **v** vertices are modified (left) and then **e** vertices are determined (right).

the evaluation of tensor-product surfaces, like B-spline surfaces, where the computation for rows and columns can be performed in any order.) Figure 9 illustrates the computation of a two-dimensional *s*-lift operation. For computing lifting operations applied to the rows of a rectilinear mesh, we use the averaging operator $\overline{k}_l^x$ returning for every vertex of type l the average of its neighbors of type k within the same row. Analogously, the operator $\overline{k}_l^y$ is used for lifting the individual columns. Applying the one-dimensional lifting operations, defined in equations (3.9) and (3.10), to the rows and then to the columns, results in the following tensor-product operations.

tensor-product s-lift(a, b)**:**

$$\begin{aligned} \tilde{\mathbf{v}} &\leftarrow b\mathbf{v} + 2a\overline{\mathbf{e}}_{\mathbf{v}}^{x} \\ \mathbf{e}' &\leftarrow b\mathbf{e} + 2a\overline{\mathbf{f}}_{\mathbf{e}} \\ \mathbf{v}' &\leftarrow b\tilde{\mathbf{v}} + 2a\overline{\mathbf{e}'}_{\mathbf{v}}^{y} \end{aligned} \tag{4.3}$$

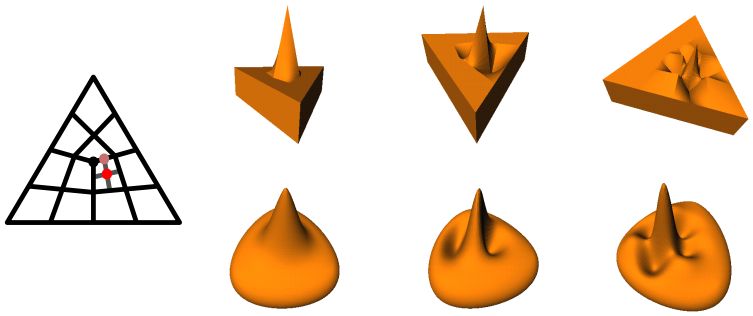

Figure 11: Basis functions around an extraordinary vertex of valence three. Left: mesh configuration (top face of prismatic base mesh with three vertices corresponding to basis functions). Top row: linear construction; bottom row: cubic construction; from left to right: scaling functions, wavelets corresponding to an edge, wavelets corresponding to a face.

tensor-product w-lift(a, b)**:**

$$\begin{aligned} \tilde{\mathbf{f}} &\leftarrow b\mathbf{f} + 2a\overline{\mathbf{e}}^x_{\mathbf{f}} \\ \mathbf{e}' &\leftarrow b\mathbf{e} + 2a\overline{\mathbf{v}}_{\mathbf{e}} \\ \mathbf{f}' &\leftarrow b\tilde{\mathbf{f}} + 2a\overline{\mathbf{e}'}^y_{\mathbf{f}} \end{aligned} \tag{4.4}$$

Changing the order of computation such that every vertex is updated only once, as shown in figure 10, results in an equivalent definition of operations:

s-lift(a, b)**:**

$$\begin{aligned} \mathbf{v}' &\leftarrow b^2\mathbf{v} + 4a^2\overline{\mathbf{f}}_{\mathbf{v}} + 4ab\overline{\mathbf{e}}_{\mathbf{v}} \\ \mathbf{e}' &\leftarrow b\mathbf{e} + 2a\overline{\mathbf{f}}_{\mathbf{e}} \end{aligned} \tag{4.5}$$

w-lift(a, b)**:**

$$\begin{aligned} \mathbf{f}' &\leftarrow b^2\mathbf{f} + 4a^2\overline{\mathbf{v}}_{\mathbf{f}} + 4ab\overline{\mathbf{e}}_{\mathbf{f}} \\ \mathbf{e}' &\leftarrow b\mathbf{e} + 2a\overline{\mathbf{v}}_{\mathbf{e}} \end{aligned} \tag{4.6}$$

These lifting operations are now defined in a notation suitable for arbitrary polyhedral meshes defining two-manifold surfaces, since the averaging operators are well-defined for extraordinary vertices. When applied to a rectilinear mesh, equations (4.5) and (4.6) reproduce tensor products of the corresponding one-dimensional operations.

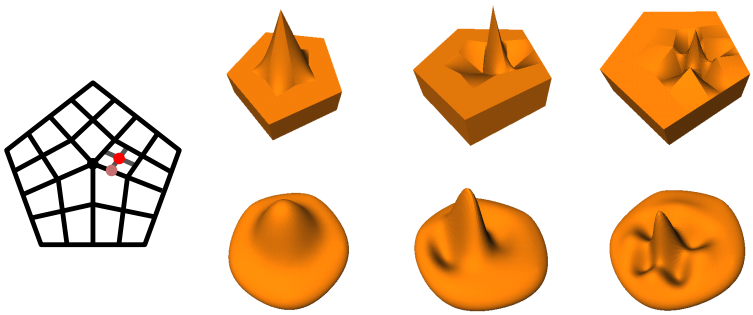

Figure 12: Basis functions around an extraordinary vertex of valence five.

For every modification step, the overall weight of adjacent vertices that is added to the modified vertex depends only on the lifting parameters a and b and is independent of the vertex valence. Wavelet coefficients that are located closely to extraordinary vertices may therefore behave similarly to wavelet coefficients located in rectilinear areas, considering their order of magnitude.

The one-dimensional wavelets constructed in section 3 are completely defined in terms of s-lift and w-lift operations. The corresponding wavelet transforms for polygon meshes are already defined in table 1 using equations (4.5) and (4.6) rather than (3.9) and (3.10). For the inverse transform, every single vertex-modification step is inverted using reverse order of operations.

Examples for the two-dimensional basis functions are shown in figures 11 and 12. These examples are obtained from prisms as base meshes that are recursively subdivided. First, the DWT (successive decomposition) is applied to a fine-resolution mesh. Then, a single vertex is pulled away from the surface, and the inverse DWT (successive reconstruction) reproduces the shape of the surface modified by the basis function corresponding to this vertex. Applying the inverse DWT using zero wavelet coefficients corresponds to a stationary subdivision process, since no geometric detail is added.

It can be observed that our generalized bicubic DWT reproduces the subdivision rules of multi-linear cell averaging (MLCA) [1]. This subdivision scheme is very similar to Catmull-

Clark subdivision, except that it behaves slightly different at extraordinary vertices. It was shown by Peters and Reif [37] that the limit surfaces of this scheme are C^1-continuous (including a number of other Catmull-Clark style schemes). Subdivision schemes reproducing B-splines of higher order were recently presented [39, 44].

To provide an example how the DWT is finally implemented, we insert the lifting operations defined by equations (4.5) and (4.6) into the decomposition formula for the cubic wavelet transform, see table 1. The resulting vertex-modification rules for the generalized bicubic wavelet transform are defined by these rules

$$\begin{aligned}
\mathbf{v} &\leftarrow \mathbf{v} + \tfrac{1}{4}\overline{\mathbf{f}}_{\mathbf{v}} - \overline{\mathbf{e}}_{\mathbf{v}} \\
\mathbf{e} &\leftarrow \mathbf{e} - \tfrac{1}{2}\overline{\mathbf{f}}_{\mathbf{e}} \\
\mathbf{f} &\leftarrow \mathbf{f} + 4\overline{\mathbf{v}}_{\mathbf{f}} - 4\overline{\mathbf{e}}_{\mathbf{f}} \\
\mathbf{e} &\leftarrow \mathbf{e} - 2\overline{\mathbf{v}}_{\mathbf{e}} \\
\mathbf{v} &\leftarrow 4\mathbf{v} + \tfrac{9}{16}\overline{\mathbf{f}}_{\mathbf{v}} + 3\overline{\mathbf{e}}_{\mathbf{v}} \\
\mathbf{e} &\leftarrow 2\mathbf{e} + \tfrac{3}{4}\overline{\mathbf{f}}_{\mathbf{e}}
\end{aligned} \tag{4.7}$$

The inverse DWT is implemented analogously by applying the inverse of every individual vertex modification in reverse order.

4.3 Integer Arithmetic for Lossless Compression

Wavelets are often used for data compression, since they de-correlate local similarity of represented functions. Smooth functions are approximated using very few coefficients, and representations of locally supported details require only a few additional wavelet coefficients in the corresponding regions to be non-zero. Thus, wavelet coefficients have expectedly small absolute values and are efficiently compressed, for example by arithmetic coding [34]. Coding schemes exploit the uneven distribution of coefficient values to reduce storage space. For high compression rates, the range of coefficient values must be very small, which is achieved by quantizing

coefficients (rounding to integers) introducing a quantization error.

In the case of biorthogonal wavelet bases, it is often difficult to control the effect of quantization errors on the reconstructed functions. Instead of quantizing wavelet coefficients, we can perform the computation of the wavelet transform in integer arithmetic, providing a tool for lossless compression [6]. Therefore, we assume that the coordinates of control points at the finest level of resolution have finite precision and are represented by integer numbers.

The lifting operations defined in equations (4.5) and (4.6) can be computed in integer arithmetic, if the lifting parameter $b \neq 0$ is an integer. (By modifying consecutive operations, b can always be scaled to an integer.) Using the rounding operator $[\cdot]$ returning an integer closest to its argument, the integer lifting operations and their inverse are defined as follows:

integer s-lift(a, b)**:**

$$\begin{aligned} \mathbf{v} &\leftarrow b^2\mathbf{v} + [4a^2\overline{\mathbf{f}}_{\mathbf{v}} + 4ab\overline{\mathbf{e}}_{\mathbf{v}}] \\ \mathbf{e} &\leftarrow b\mathbf{e} + [2a\overline{\mathbf{f}}_{\mathbf{e}}] \end{aligned} \tag{4.8}$$

integer w-lift(a, b)**:**

$$\begin{aligned} \mathbf{f} &\leftarrow b^2\mathbf{f} + [4a^2\overline{\mathbf{v}}_{\mathbf{f}} + 4ab\overline{\mathbf{e}}_{\mathbf{f}}] \\ \mathbf{e} &\leftarrow b\mathbf{e} + [2a\overline{\mathbf{v}}_{\mathbf{e}}] \end{aligned} \tag{4.9}$$

inverse integer s-lift(a, b)**:**

$$\begin{aligned} \mathbf{e} &\leftarrow \tfrac{1}{b}\left(\mathbf{e} - [2a\overline{\mathbf{f}}_{\mathbf{e}}]\right) \\ \mathbf{v} &\leftarrow \tfrac{1}{b^2}\left(\mathbf{v} - [4a^2\overline{\mathbf{f}}_{\mathbf{v}} + 4ab\overline{\mathbf{e}}_{\mathbf{v}}]\right) \end{aligned} \tag{4.10}$$

inverse integer w-lift(a, b)**:**

$$\begin{aligned} \mathbf{e} &\leftarrow \tfrac{1}{b}\left(\mathbf{e} - [2a\overline{\mathbf{v}}_{\mathbf{e}}]\right) \\ \mathbf{f} &\leftarrow \tfrac{1}{b^2}\left(\mathbf{f} - [4a^2\overline{\mathbf{v}}_{\mathbf{f}} + 4ab\overline{\mathbf{e}}_{\mathbf{f}}]\right) \end{aligned} \tag{4.11}$$

An integer-to-integer DWT is constructed from above lifting operations analogously to equation (4.7). To improve compression rates, we divide the lifting parameters a and b in the last s-lift operation by two in the cubic construction, and by four in the quintic construction, such

wavelet transform	"two-blobs"	"five-blobs"
linear	24.6	31.0
cubic	11.8	14.8
quintic	18.8	20.5
uncompressed	172.1	172.1

Table 3: Storage requirements in kilobytes for surfaces shown in figure 13. The uncompressed representation uses four bytes to store a coordinate.

that b is always one. This modification reduces the precision for resulting scaling-function coefficients at the coarser levels (and also affects the normalization of basis functions on different levels). This loss of precision does not introduce artifacts when all levels of detail are used for reconstruction, since the inverse DWT exactly reproduces every coefficient. It may cause artifacts, however, when displaying coarse levels of resolution.

5 Numerical Results

5.1 Lossless Compression and Level of Detail

Surfaces can be represented by fine-resolution control meshes with subdivision connectivity. At the very finest level, a surface nearly interpolates its control points. When representing a geometric shape, we sample the control points from this geometry rather than solving a global interpolation problem. The coordinates of these control points can be represented by integer numbers (at finite precision). Lossless compression of this surface representation is feasible by applying our integer wavelet transform and, for example, arithmetic coding [34].

We have applied our compression scheme to the isosurfaces, "two-blobs" and "five-blobs", obtained from trivariate scalar fields, each defined as a sum of Gaussians, see figure 13. The isosurfaces are approximated by alternating mesh subdivision and Newton iteration, projecting the mesh vertices onto the isosurfaces. The finest level of detail is obtained after five subdivision steps, resulting in a total of 14338 vertices for each surface. Every vertex is represented by integer coordinates with a precision of 0.01 percent of the height of the surface model. Lossless

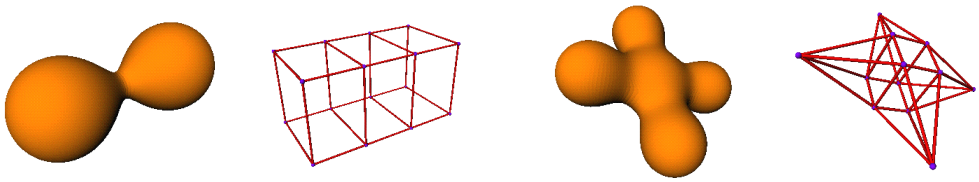

Figure 13: Isosurfaces "two-blobs" (left) and "five-blobs" (right) with associated base meshes.

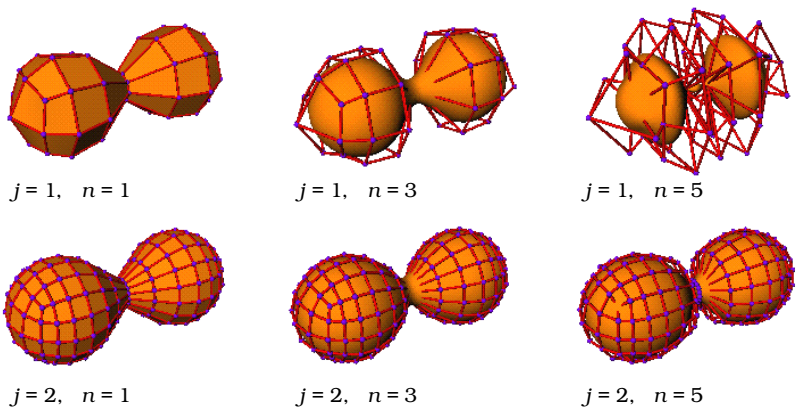

Figure 14: Isosurface "two-blobs" at two levels of resolution ($j = 1, 2$) obtained from a floating-point implementation of the DWT with polynomial degree n.

compression rates obtained with arithmetic coding are listed in table 3. In addition, we need to store the base-mesh connectivity and the histogram of coordinate values for the arithmetic coder.

A different application for our wavelet transform is level-of-detail representation. Coarse surface representations are obtained by replacing wavelet coefficients of finer levels by zero. In this case, the inverse wavelet transform is equivalent to a subdivision-surface scheme. Coarse representations for the "two-blobs" and "five-blobs" isosurfaces are shown in figures 14 and 15, where we used floating-point arithmetic. In the quintic case, surfaces may be poorly represented by coarse levels of resolution, since the support of the basis functions is so large that "self-overlaps" occur when wrapped around small surface components.

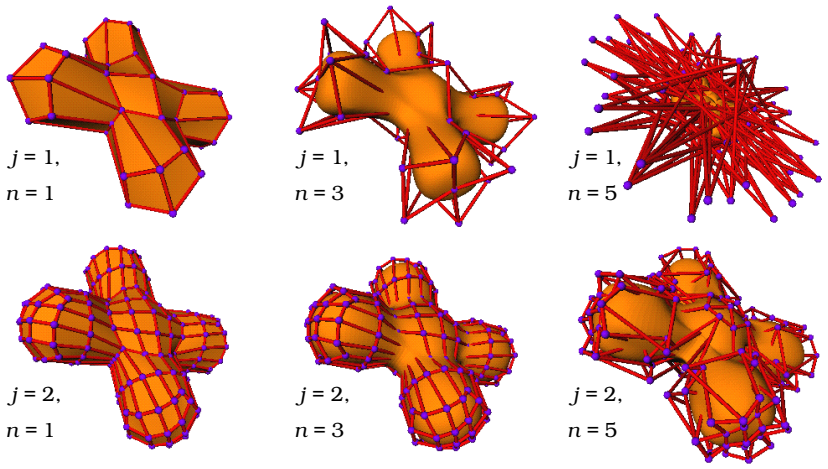

Figure 15: Isosurface "five-blobs" at two levels of resolution ($j = 1, 2$).

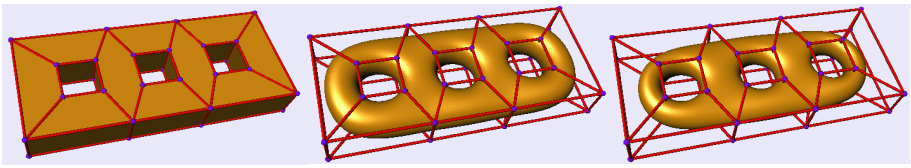

Figure 16: Different surfaces are produced by the individual subdivision rules of our generalized linear, cubic, and quintic wavelet transforms. The base mesh for this example is composed of 40 vertices and 40 faces.

5.2 Examining Stability

When displaying coarse levels of surface detail using the low-pass filter l of a wavelet transform, it is important to understand the behavior of this filter when applied multiple times. In the case of semi-orthogonal wavelets [42], a convolution with l is equivalent to least squares fitting based on a set of coarser scaling functions. This is due to the orthogonality of the spaces V_j and W_j, spanned by scaling functions and wavelets, respectively. In the case of biorthogonal wavelets, however, this fitting process is a local operation that is not optimal, with respect to its residual. In some cases, the distance to the original surface may grow rapidly when l is applied multiple times, resulting in unstable behavior of the wavelet transform.

To examine the stability of our wavelet transforms, we perform a simple experiment: We construct a subdivision surface from a coarse base mesh by applying the inverse DWT with zero wavelet coefficients, see figure 16. After a fixed number of subdivisions, we add white noise to

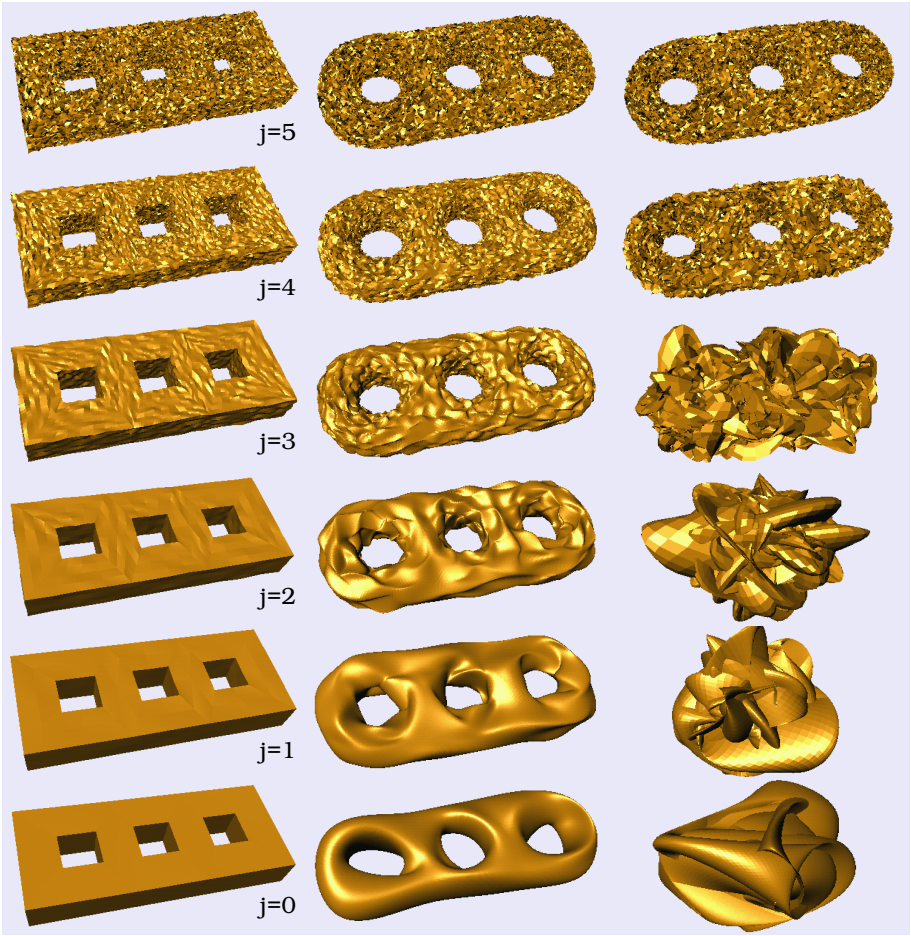

Figure 17: Examining stability by low-pass filtering noisy fine-resolution models (fifth subdivision, 43008 points). The linear and cubic transforms remove most of the noise while the quintic wavelet turns out to be unstable.

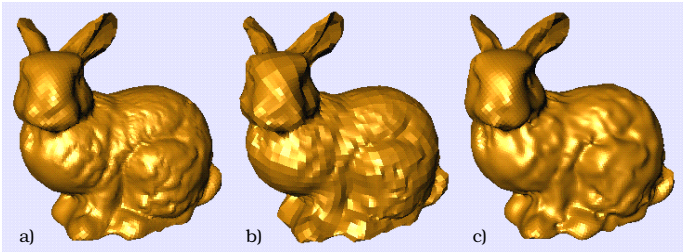

Figure 18: Compressing the Stanford bunny (shrink-wrapped version) composed of 24576 control points, each using 12 bytes. a) Original mesh (sixth subdivision level of a cube), b) 100:1 compression using linear wavelet, c) 100:1 compression based on cubic wavelet.

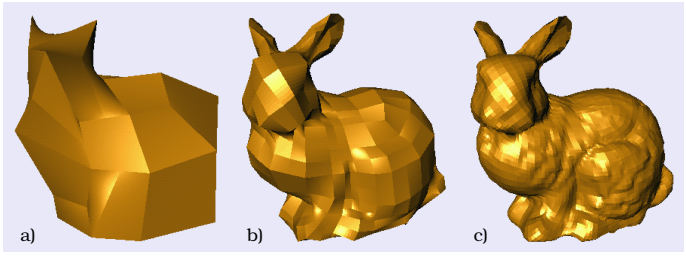

Figure 19: Selecting wavelet coefficients by magnitude using linear B-spline wavelets. Reconstruction from a) 0.1 percent, b) one percent, c) 10 percent of coefficients.

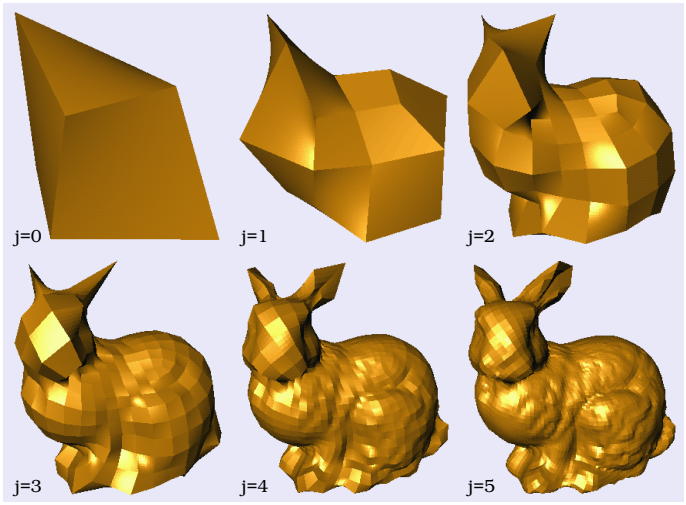

Figure 20: Progressive transmission of Stanford bunny using linear wavelet. Shown are six levels of resolution ($j = 0, 1, \cdots, 5$).

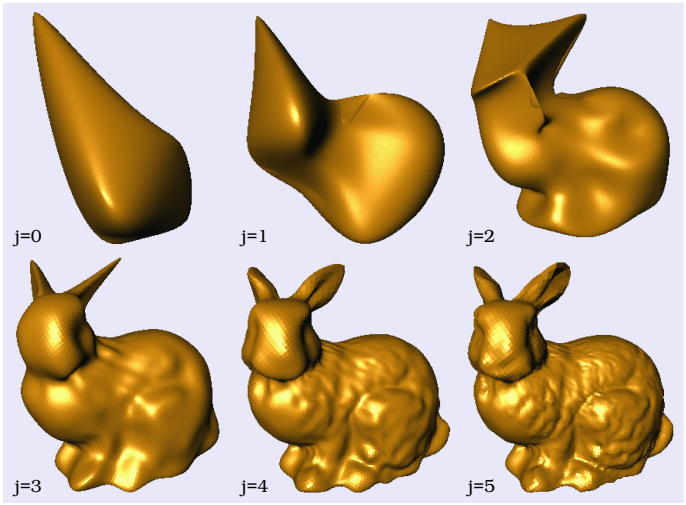

Figure 21: Progressive transmission of Stanford bunny using cubic wavelet.

every control point and display the individual levels of resolution obtained by low-pass filtering the noisy surface with l, which corresponds to removing wavelet coefficients on the finest levels.

The results of the noise-removal experiment are shown in figure 17. The linear wavelet transform provides the best fitting operation, quickly reducing the noise to an invisible amount after just a few steps. The cubic wavelet transform leaves the amplitude of the noise nearly constant over a number of fitting steps. This wavelet construction is still stable enough for practical applications. The quintic wavelet, however, exhibits instable behavior. The noise amplitude grows rapidly and deforms the coarse-resolution surfaces such that the object soon becomes unrecognizable. This transform can only be used for lossless compression based on integer arithmetic where the coefficients are divided by four and thus are diminished in every decomposition step.

5.3 Compression and Progressive Transmission

Surfaces of known topology can be constructed from sampled three-dimensional points by *shrink-wrapping* a given base mesh towards these points [24]. After subdividing a base mesh, its vertices are projected along normal vectors towards closest sample points. The mesh is then relaxed perpendicularly to its normals. Multiple attraction and relaxation steps are necessary after every subdivision step.

We have shrink-wrapped a surface to 35947 samples of the Stanford bunny, courtesy of the Stanford University Computer Graphics Laboratory. As a base mesh, we used the bounding box of the samples, which produced 24578 vertices after six levels of subdivision. The shrink-wrapped mesh and a 100:1 compression based on arithmetic coding of quantized wavelet coefficients are shown in figure 18. Table 4 lists our compression results. Higher compression rates can be obtained by by offsetting surface detail rather than adding arbitrary vectors to every control point, providing scalar-valued coefficients [20].

wavelet transform	compression rate	code length [kB]	$\mathbf{L}^2$-error
linear	10	29.5	0.0070
	100	2.9	0.13
cubic	10	29.5	0.0024
	100	2.9	0.30
uncompressed	1	294.9	0

Table 4: Compression results for shrink-wrapped Stanford bunny. The uncompressed mesh is composed of 24578 vertices using four bytes for every coordinate. The errors were computed with respect to the finest mesh, in percent of the diagonal of the bounding box.

level j	number of vertices	$\mathbf{L}^2$-error for linear wavelet	$\mathbf{L}^2$-error for cubic wavelet
0	8	5.33	13.4
1	26	2.46	6.70
2	98	0.955	3.13
3	386	0.388	0.746
4	1538	0.135	0.167
5	6146	0.0504	0.0216
6	24578	0	0

Table 5: Approximation errors for different levels of resolution with respect to the highest-resolution mesh (level 6).

For progressive transmission, the coefficients are sorted by magnitude before they are encoded, see figure 19. Unfortunately, the order of coefficients must be transmitted, as well. A simpler method processes the coefficients of the coarse levels first and then adds finer levels progressively, as shown in figures 20, 21, and table 5.

6 Conclusions

We have constructed biorthogonal wavelets for linear, cubic, and quintic B-spline subdivision surfaces, suitable for multiresolution representation, compression, and progressive transmission of geometric models of arbitrary genus. Our wavelet transforms require surfaces to be represented as meshes with subdivision connectivity, which can be obtained by re-meshing [26, 24, 3, 28, 17]. To make wavelet techniques even more attractive for level-of-detail representation, future work will be directed at topological changes between different levels of resolution, according to *hybrid meshes* [19].

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. Additionally, this work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award), ACI 9982251 (LSSDSV), through the National Partnership for Advanced Computational Infrastructure (NPACI), and a large Information Technology Research (ITR) grant; the National Institutes of Health under contract P20 MH60975-06A2, funded by the National Institute of Mental Health and the National Science Foundation; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 memorandum agreement B347878, and agreements B503159 and B523294; and the Lawrence Berkeley National Laboratory.

References

[1] C. Bajaj, S. Schaefer, J. Warren, and G. Xu, *A subdivision scheme for hexahedral meshes*, The Visual Computer, special issue on subdivision, vol. 18, 2002, pp. 343–356.

[2] M. Bertram, *Multiresolution Modeling for Scientific Visualization*, Dissertation, University of California, Davis, July 2000.

[3] M. Bertram, M.A. Duchaineau, B. Hamann, and K.I. Joy, *Bicubic subdivision-surface wavelets for large-scale isosurface representation and visualization*, IEEE Visualization, 2000, pp. 389–396 & 579.

[4] G.-P. Bonneau, *Optimal triangular Haar bases for spherical data*, IEEE Visualization, 1999, pp. 279–284 & 534.

[5] G.-P. Bonneau, *Multiresolution analysis on irregular surface meshes*, IEEE TVCG, Vol. 4, No. 4, 1998, pp. 365–378.

[6] R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, *Wavelet transforms that map integers to integers*, Applied and Computational Harmonic Analysis, Vol. 5, No. 3, Academic Press, 1998, pp. 332–369.

[7] E. Catmull and J. Clark, *Recursively generated B-spline surfaces on arbitrary topological meshes*, Computer-Aided Design, Vol 10, No. 6, Nov. 1978, pp. 350–355.

[8] A.S. Cavaretta , C.A. Micchelli , and W. Dahmen, *Stationary Subdivision*, American Mathematical Society, Boston, MA, 1991.

[9] A. Certain, J. Popovic, T. DeRose, T. Duchamp, D. Salesin, and W. Stuetzle, *Interactive multiresolution surface viewing*, ACM Siggraph , 1996, pp. 91–97.

[10] C.K. Chui, *An Introduction to Wavelets*, Academic Press, 1992.

[11] W. Dahmen, *Decomposition of refinable spaces and applications to operator equations*, Numerical Algorithms, Vol. 5, 1993, pp. 229–245.

[12] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, 1992.

[13] T. DeRose, M. Kass, and T. Truong, *Subdivision surfaces in character animation*, ACM Siggraph, 1998, pp. 85–94.

[14] D. Doo and M. Sabin, *Behaviour of recursive division surfaces near extraordinary points*, Computer-Aided Design, Vol 10, No. 6, 1978, pp. 356–360.

[15] M.A. Duchaineau, *Dyadic Splines*, Dissertation, Department of Computer Science University of California, Davis, 1996.

[16] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, *Multiresolution analysis of arbitrary meshes*, ACM Siggraph, 1995, pp. 173–182.

[17] M.S. Floater, K. Hormann, and M. Reimers, *Parameterization of manifold triangulations*, in Approximation Theory X: Abstract and Classical Analysis, C.K. Chui, L.L. Schumaker, and J. Stockler (eds), Vanderbilt University Press, Nashville, 2002, pp. 197–209.

[18] M.S. Floater, E.G. Quak, and M. Reimers, *Filter bank algorithms for piecewise linear prewavelets on arbitrary triangulations*, Journal of Computational and Applied Mathematics, Vol. 119, 2000, pp. 185–207.

[19] I. Guskov, A. Khodakovsky, P. Schröder, and W. Sweldens, *Hybrid meshes: multiresolution using regular and irregular refinement*, Eighteenth Annual Symposium on Computational Geometry, 2002, pp. 264–272.

[20] I. Guskov, K. Vidimce, W. Sweldens, and P. Schröder, *Normal meshes*, ACM Siggraph, 2000, pp. 95–102.

[21] I. Guskov, W. Sweldens, and P. Schröder, *Multiresolution signal processing for meshes*, ACM Siggraph, 1999, pp. 325–334.

[22] A. Khodakovsky, P. Schöder, and W. Swelddens, *Progressive geometry compression*, ACM Siggraph, 2000, pp. 271–278.

[23] L. Kobbelt and P. Schröder, *A multiresolution framework for variational subdivision*, ACM Transactions on Graphics, Vol. 17, No. 4, 1998, pp. 209–237.

[24] L.P. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel, *A shrink wrapping approach to remeshing polygonal surfaces*, Eurographics 1999, pp. 119–129.

[25] L. Kobbelt, J. Vorsatz, and H.-P. Seidel, *Multiresolution hierarchies on unstructured triangle meshes*, Computational Geometry: Theory and Applications, Vol. 14, No. 1-3, Elsevier, 1999, pp. 5–24.

[26] A.W.F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin, *MAPS: multiresolution adaptive parameterization of surfaces*, ACM Siggraph, 1998, pp. 95–104.

[27] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, *The digital Michelangelo project: 3D scanning of large statues*, ACM Siggraph, 2000, pp. 131–144.

[28] N. Litke, A. Levin, and P. Schröder, *Fitting Subdivision Surfaces*, IEEE Visualization, 2001, pp. 319–324 & 568.

[29] J.M. Lounsbery, *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*, Dissertation, Department of Mathematics, University of Washington, Seattle, 1994.

[30] M. Lounsbery, T. DeRose, and J. Warren, *Multiresolution analysis for surfaces of arbitrary topological type*, ACM Transactions on Graphics, Vol. 16, No. 1, ACM, 1997, pp. 34–73.

[31] C.T. Loop, *Smooth subdivision surfaces based on triangles*, M.S. Thesis, Department of Mathematics, University of Utah, Salt Lake City, 1987.

[32] R. MacCracken and K.I. Joy, *Free-form deformations with lattices of arbitrary topology*, ACM Siggraph, 1996, pp. 181–188.

[33] S.G. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 1998.

[34] A. Moffat, R.M. Neal, and I.H. Witten, *Arithmetic coding revisited*, ACM Transactions on Information Systems, Vol. 16, No. 3, 1998, pp. 256–294.

[35] G.M. Nielson, I.-H. Jung, and J. Sung, *Haar wavelets over triangular domains with applications to multiresolution models for flow over a sphere*, IEEE Visualization, 1997, pp. 143–150.

[36] J. Peters, *Patching Catmull-Clark meshes*, ACM Siggraph, 2000, pp. 255–258.

[37] J. Peters and U. Reif, *Analysis of algorithms generalizing B-spline subdivision*, SIAM Journal on Numerical Analysis, Vol. 13, No. 2, 1998, pp. 728–748.

[38] P. Schröder and W. Sweldens, *Spherical wavelets: efficiently representing functions on the sphere*, ACM Siggraph, 1995, pp. 161–172.

[39] J. Stam, *On subdivision schemes generalizing uniform B-spline surfaces of arbitrary degree*, Computer Aided Geometric Design, vol. 18, no. 5, 2001, pp. 383–396.

[40] J. Stam, *Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values*, ACM Siggraph, 1998, pp. 395–404.

[41] R. Stevenson. *Piecewise linear (pre-)wavelets on non-uniform meshes*, in W. Hackbusch and G. Wittum (eds), Multigrid Methods V, Lecture Notes in Computational Science and Engineering, vol. 3, Fifth European Multigrid Conference, Springer, 1998, pp. 306-319.

[42] E.J. Stollnitz, T.D. DeRose, and D.H. Salesin, *Wavelets for Computer Graphics, Theory and Applications*, Morgan Kaufmann, 1996.

[43] W. Sweldens, *The lifting scheme: A custom-design construction of biorthogonal wavelets*, Applied and Computational Harmonic Analysis, Vol. 3, No. 2, 1996, pp. 186–200.

[44] D. Zorin and P. Schroder, A unified framework for primal/dual quadrilateral subdivision schemes, Computer-Aided Geometric Design, vol. 18, no. 5, 2001. pp. 429–454.