

Multiresolution Analysis for Surfaces of Arbitrary Topological Type

MICHAEL LOUNSBERY, TONY D. DeROSE, and JOE WARREN

University of Washington, Seattle

Multiresolution analysis and wavelets provide useful and efficient tools for representing functions at multiple levels of detail. Wavelet representations have been used in a broad range of applications, including image compression, physical simulation, and numerical analysis. In this article, we present a new class of wavelets, based on subdivision surfaces, that radically extends the class of representable functions. Whereas previous two-dimensional methods were restricted to functions defined on \mathbb{R}^2 , the subdivision wavelets developed here may be applied to functions defined on compact surfaces of arbitrary topological type. We envision many applications of this work, including continuous level-of-detail control for graphics rendering, compression of geometric models, and acceleration of global illumination algorithms. Level-of-detail control for spherical domains is illustrated using two examples: shape approximation of a polyhedral model, and color approximation of global terrain data.

Categories and Subject Descriptors: G.1.2 [**Approximation**]: Spline Approximation; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*surfaces and object representations*; J.6 [**Computer-Aided Engineering**]: Computer-Aided Design (CAD)

General Terms: Algorithms, Design, Theory

Additional Key Words and Phrases: Compression, geometric modeling, level-of-detail control, splines, subdivision surfaces, wavelets.

1. INTRODUCTION

Multiresolution analysis and wavelets have received considerable attention in recent years, fueled largely by the diverse collection of problems that benefit from their use. The basic idea behind multiresolution analysis is to decompose a complicated function into a “simpler” low resolution part, together with a collection of perturbations, called wavelet coefficients, necessary to recover the original function. For many of the functions encountered in practice, a large percentage of the wavelet coefficients are small, meaning that good approximations can be obtained by using only a

This work was made possible in part by Alias/Wavefront, and the National Science Foundation under grants CCR-8957323, CDA-9123308, and CCR-9113239.

Authors' current addresses: M. Lounsbery, Alias/Wavefront, Seattle, WA; T. DeRose, Pixar Animation Studios, Richmond, CA; J. Warren, Rice University, Houston, TX.

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.

© 1997 ACM 0730-0301/97/0100-0034 \$03.50

few of the largest coefficients. Impressive “lossy” compression rates for images have been achieved using this type of approximation [DeVore et al. 1992].

There are many constructions of wavelets for functions parametrized over the interval [Andersson et al. 1993; Chui and Quak 1992; Cohen et al. 1993; Meyer 1992]. These have found use in signal processing [Mallat 1989], B-spline modeling [Finkelstein and Salesin 1994], motion planning [Liu et al. 1994], and many other applications involving functions parametrized in one dimension.

Two-dimensional wavelets are important for a variety of applications including image compression. They are generally constructed by forming tensor products of univariate wavelets [Daubechies 1992], in much the same way that tensor product B-spline surfaces are formed by products of univariate B-splines. Unfortunately, tensor-product constructions require that the functions to be decomposed be defined on \mathbb{R}^2 or on a periodic version of \mathbb{R}^2 , that is, the cylinder or torus. There also exist nontensor-product constructions for wavelets on \mathbb{R}^2 [Daubechies 1992; Jia and Micchelli 1991], but none of these methods are applicable to functions defined on more general topological domains, such as spheres. Thus, existing methods are not well suited for decomposing and compressing surfaces such as the ones shown in Figures 10 and 12, since they are described by parametric functions on the sphere.

In this article we show that by using techniques from subdivision surfaces, multiresolution analysis can be extended to functions defined on domains of arbitrary topological type (the topological type of a two-dimensional surface or domain refers to its genus, presence of boundary curves, etc.). This generalization, which we term “subdivision wavelets,” dramatically extends the class of applications to which multiresolution analysis can be applied, including:

- Polyhedral compression.* Using wavelet-based techniques, most polyhedral models may be compressed to yield a more compact approximation. Compression saves both storage space and the time that is required to process a surface model. This article develops efficient wavelet compression methods for surfaces similar to those that have proven effective for images and one-dimensional functions. These techniques are capable of L^∞ and L^2 approximation, and run more quickly than many other compression methods. This application is explored in more detail in Eck et al. [1995].
- Continuous level-of-detail control.* When a complex shape is rendered in an animation, a fully detailed representation of the shape contains much more detail than is required for all but the closest view. Using a compressed subdivision wavelet representation of complex objects, it is possible to greatly reduce the number of polygons in a scene without significantly impacting the visible detail (see Figure 10). Moreover, it is possible to smoothly vary the detail, avoiding the discontinuous jumps that occur when suddenly switching between distinct models. This appli-

cation is discussed in more detail in Section 7.5. Nonwavelet treatments of this problem may be found in Turk [1992], Schroeder et al. [1992], and Hoppe et al. [1993].

- Compression of functions defined on surfaces.* Consider the situation shown in Figure 12 where a globe (a geometric sphere) is pseudo-colored according to elevation. The pseudo-coloring can be thought of as a function that maps each point on the sphere to an RGB triple. A straightforward method for storing the function is to store its value at a large number of regularly distributed points; in this case more than one million points were used. The methods in this article can be used to create compressed wavelet approximations of varying complexity. (The mesh lines on the original surface are so dense that the image shown in Figure 11(b) is nearly black.)
- Multiresolution editing of surfaces.* Hierarchical B-splines, as introduced by Forsey and Bartels [1988], provide a powerful mechanism for editing shapes at various levels of detail. However, hierarchical B-splines can only represent a restricted class of surface topologies. The methods described here provide an alternative to hierarchical B-splines, and are capable of representing smooth multiresolution surfaces of arbitrary topological type. Editing at fractional levels of detail can also be achieved using the methods developed by Finkelstein and Salesin [1994].
- Surface optimization.* The multiple levels of approximation produced by wavelet techniques offer a sort of multigrid technique for optimization. Pentland [1992] uses wavelet methods to implement multigrid optimization for surface interpolation over regular grids. Meyers [1994a; 1994b] shows how wavelets can accelerate the reconstruction of surfaces from contour data. This previous work suggests that subdivision wavelets may find use in optimization over surfaces of arbitrary topological type.
- Numerical solution of integral and differential equations.* Wavelet representations of functions on surfaces of arbitrary topological type appear well suited for adaptive numerical solution, along the lines of Beylkin et al. [1991].

A carefully constructed special-purpose algorithm may produce superior results for some of these applications. However, the above examples indicate the versatility of subdivision wavelets, and show them to be a reusable tool that naturally and efficiently solve or accelerate a wide range of common problems in computer graphics and modeling. As an illustration of this versatility, a complex object of arbitrary topological type may be modeled at multiple levels of detail, stored in a compressed form, viewed in an animation, and rendered using global illumination—all naturally implemented using the same underlying surface wavelet representation.

This article presents a theoretical foundation for developing multiresolution analysis for surfaces of arbitrary topological type. (Additional details of implementation and an expanded treatment of applications may be found in Lounsbery [1994], Eck et al. [1995], and Certain et al. [1996].)

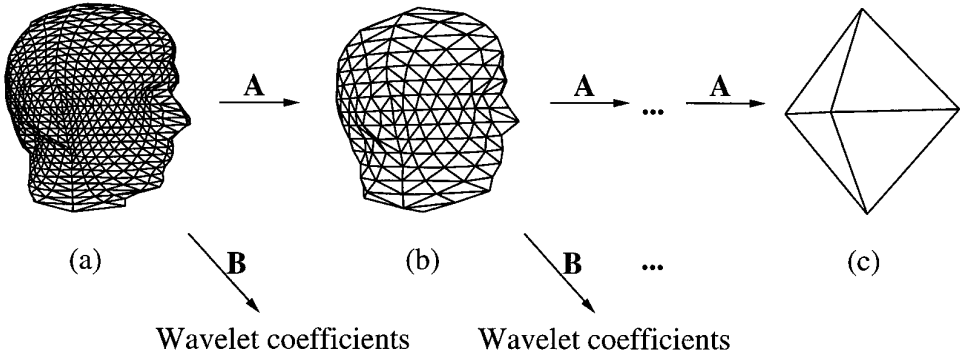


Fig. 1. Decomposition of a polyhedral surface.

The construction of subdivision wavelets described herein applies directly to a variety of existing subdivision schemes. These include piecewise linear subdivision (producing polyhedra); the schemes of Catmull and Clark [1978], Halstead et al. [1993], Loop [1987] or Dyn et al. [1990] (producing tangent-plane-smooth G^1 subdivision surfaces); and the modifications by Hoppe [1994] and Hoppe et al. [1994] (producing piecewise smooth surfaces with selected discontinuities). More generally, the techniques presented here may be used to construct wavelets for any local, stationary, continuous, uniformly convergent subdivision scheme (these terms will be described in Section 4).

The remainder of the article is structured as follows. In Section 2, we present a high-level preview of how to convert a polyhedral object to multiresolution form. In Section 3, we provide some background on multiresolution analysis. In Section 4, we provide a brief overview of subdivision surfaces, and we show that they can be used to define a collection of scaling functions necessary for multiresolution analysis. In Section 5, we show that inner products of these scaling functions can be computed exactly. In Section 6, we use the inner products to construct wavelets, and to construct locally supported approximations to them. In Section 7, we apply the theory to polyhedral compression. In Section 8, we apply the theory to the problem of compression and editing of the smooth surfaces described by Dyn et al. [1990]. Finally, in Section 9, we summarize the contributions of the article, and list several topics for future work.

2. A PREVIEW OF THE METHOD

Although the mathematical underpinnings of multiresolution analysis of surfaces are somewhat involved, the resulting algorithms are quite simple. Before diving into the details, we give here a brief description of how the method can be applied to decompose the polyhedral object shown in Figure 1(a).

As mentioned in Section 1, a main idea behind multiresolution analysis is the decomposition of a function (in this case a polyhedron expressed as a parametric function on the sphere) into a low resolution part and a “detail”

part. The low resolution part of the polyhedron in Figure 1(a) is shown in Figure 1(b). The vertices in Figure 1(b) are computed as certain weighted averages of the vertices in Figure 1(a). These weighted averages essentially implement a low pass filter denoted as **A**. The detail part consists of a collection of fairly abstract coefficients, called wavelet coefficients, computed as weighted differences of the vertices in Figure 1(a). These differencing weights form a high-pass filter **B**. The decomposition process, technically called *analysis*, can be used to further split Figure 1(b) into an even lower resolution version and corresponding wavelet coefficients. This cascade of analysis steps, referred to as a *filter bank* algorithm, culminates with the coarsest-level representation in Figure 1(c), together with wavelet coefficients at each level.

The analysis filters **A** and **B** are constructed so that the original polyhedron can be recovered exactly from the low-resolution version and the wavelet coefficients. Recovery, technically called *synthesis*, reconstructs Figure 1(a) from Figure 1(b) and the finest-level wavelet coefficients. Recovery refines each triangle of Figure 1(b) into four subtriangles by introducing new vertices at edge midpoints, followed by perturbing the resulting collection of vertices according to the wavelet coefficients. The refining and perturbing steps are described by two other filters **P** (the refining filter) and **Q** (the perturbing filter), collectively called synthesis filters.

The trick is to develop the four analysis and synthesis filters so that: (1) the low-resolution versions are good approximations of the original object (in a least-squares sense); (2) the magnitude of a wavelet coefficient reflects a coefficient's importance by measuring the error introduced when the coefficient is set to zero; and (3) analysis and synthesis filter banks should have time complexity that is linear in the number of vertices.

3. BACKGROUND ON MULTIREOLUTION ANALYSIS

Multiresolution analysis as formulated by Mallat [1989] and Meyer [1993] provides a convenient framework for developing the analysis and synthesis filters. There are two basic ingredients for a multiresolution analysis: an infinite chain of nested linear function spaces $V^0 \subset V^1 \subset V^2 \subset \dots$ and an inner product $\langle f, g \rangle$ defined on any pair of functions $f, g \in V^j$, for some $j < \infty$. Intuitively, V^j contains functions of resolution j , with the detail increasing as j increases.

The inner product is used to define the orthogonal complement spaces W^j as

$$W^j := \{f \in V^{j+1} \mid \langle f, g \rangle = 0 \quad g \in V^j\}.$$

Orthogonal complements are often written as $V^{j+1} = V^j \oplus W^j$ because any function $f^{j+1} \in V^{j+1}$ can be written uniquely as an orthogonal decomposition

$$f^{j+1} = f^j + h^j,$$

where $f^j \in V^j$ and $h^j \in W^j$. Orthogonal decompositions are important for approximation purposes: it is easy to show that f^j is the best approximation to f^{j+1} in that it is the unique function in V^j that minimizes the least-squares residual $\langle f^{j+1} - f^j, f^{j+1} - f^j \rangle$. Thus, given a high-resolution function f^{j+1} , its low-resolution part is f^j , and its detail part is h^j .

The following terminology is now standard: *scaling functions* refer to bases for the spaces V^j , and *wavelets* refer to bases for the orthogonal complement spaces. As shown in Section 6.3, the analysis and synthesis filters are determined by considering various ways of changing bases between scaling functions and wavelets.

The strictest form of wavelets, such as those constructed by Daubechies [1988] and Mallat [1989], are *fully orthogonal*, meaning that every wavelet $\psi_i^j(\mathbf{x})$ is orthogonal to every other wavelet $\psi_{i'}^{j'}(\mathbf{x})$; that is, $\langle \psi_i^j(\mathbf{x}), \psi_{i'}^{j'}(\mathbf{x}) \rangle = 0$ unless $j = j'$ and $i = i'$.

It has been shown to be impossible to construct wavelets that are simultaneously locally supported, fully orthogonal, and symmetric. It is therefore sometimes convenient to relax the fully orthogonal condition to the *semiorthogonal* setting, and require only orthogonality between wavelets at different levels: $\langle \psi_i^j(\mathbf{x}), \psi_{i'}^{j'}(\mathbf{x}) \rangle = 0$ unless $j = j'$. The locally supported B-spline wavelets constructed by Chui [1992] are a good example of a semiorthogonal construction.

Finally, the least restrictive form of wavelets are said to be *biorthogonal*, a term coined by Cohen et al. [1992] to refer to the setting where orthogonality is dropped entirely. It is not even necessary for the the wavelet spaces W^j to be orthogonal complements—in general W^j is just some complement of V^j in V^{j+1} . Our construction, described in Section 6, results in locally supported biorthogonal wavelets.

4. NESTED LINEAR SPACES THROUGH SUBDIVISION

A fundamental requirement for multiresolution analysis is a sequence of nested linear spaces. In this section, we carry this property to surfaces of arbitrary topological type, demonstrating the existence of scaling functions on subdivision surfaces.

The nested sequence of linear spaces required by multiresolution analysis are ordinarily obtained by defining a single scaling function $\phi(x)$ that satisfies a *refinement equation* of the form

$$\phi(x) = \sum_i p_i \phi(2x - i)$$

for some fixed constants p_i . The refinement equation (sometimes called a two-scale relation) guarantees that the spaces defined as

$$V^j := \text{Span}\{\phi(2^j x - i) | i = -\infty, \dots, \infty\}$$

are nested. In other words, the nested spaces are generated by translations and dilations of a single refinable function $\phi(x)$.

To generalize these ideas to domains of arbitrary topological type, one could attempt to make definitions for what it means to dilate and translate a function on an arbitrary topological domain. One could then try to find a refinable scaling function and proceed as before to define orthogonal complements, wavelets, and so on. We have instead chosen what appears to be a simpler approach.

In this section, recursive subdivision is used to define a collection of functions $\phi_i^j(\mathbf{x})$ that are refinable—in the sense that each function with superscript j lies in the span of the functions with superscript $j + 1$; the argument \mathbf{x} is a point that ranges over a domain 2-manifold of arbitrary topological type. In one respect, this is a generalization of the approach taken by Daubechies [1992], whose locally supported orthogonal scaling functions are also defined through a recursive subdivision procedure. Although in general the $\phi^{j+1}(\mathbf{x})$ are not simple dilates of the $\phi^j(\mathbf{x})$, one can nonetheless use them to define a sequence of nested spaces.

(We should note that the realization that translation and dilation are not strictly necessary for the construction of wavelets has also been noted independently by Sweldens [1994] and Carnicer et al. [1994].)

4.1 Subdivision Surfaces

Intuitively speaking, subdivision surfaces are defined by iteratively refining a control polyhedron M^0 so that the sequence of increasingly faceted polyhedra M^1, M^2, \dots converge to some limit surface M^∞ . In each subdivision step, the vertices of M^{j+1} are computed as affine combinations of the vertices of M^j . Thus, if \mathbf{V}^j is a matrix whose i th row consists of the x , y , and z coordinates of vertex i of M^j , there exists a nonsquare matrix of constants \mathbf{P}^j such that

$$\mathbf{V}^{j+1} = \mathbf{P}^j \mathbf{V}^j. \quad (1)$$

The matrix \mathbf{P}^j therefore characterizes the subdivision method. The beauty of subdivision surface schemes is that the entries of \mathbf{P}^j depend only on the connectivity of the vertices in M^j , not on the geometric positions of the vertices. Subdivision schemes are typically *local*, meaning that each vertex of M^{j+1} is computed as an affine combination of nearby vertices of M^j .

The simplest example of such a scheme is polyhedral subdivision. Given a polyhedron M^0 with triangular faces, a new polyhedron M^1 is built by splitting each triangular face of M^0 into four subfaces as in Figure 2. The matrix \mathbf{P}^0 characterizing the first subdivision step is also shown in Figure 2. Running this subdivision scheme for j steps on an initial triangular mesh M^0 produces a mesh M^j . M^j includes the vertices of M^0 together with new vertices introduced through subdivision. The valence (the number of edges incident to a vertex) of the vertices of M^j corresponding to the original vertices in M^0 remains fixed. The new vertices introduced through subdivision however are always of valence six, corresponding to a regular triangular tiling of the surface. As the mesh is further subdivided, the so-called

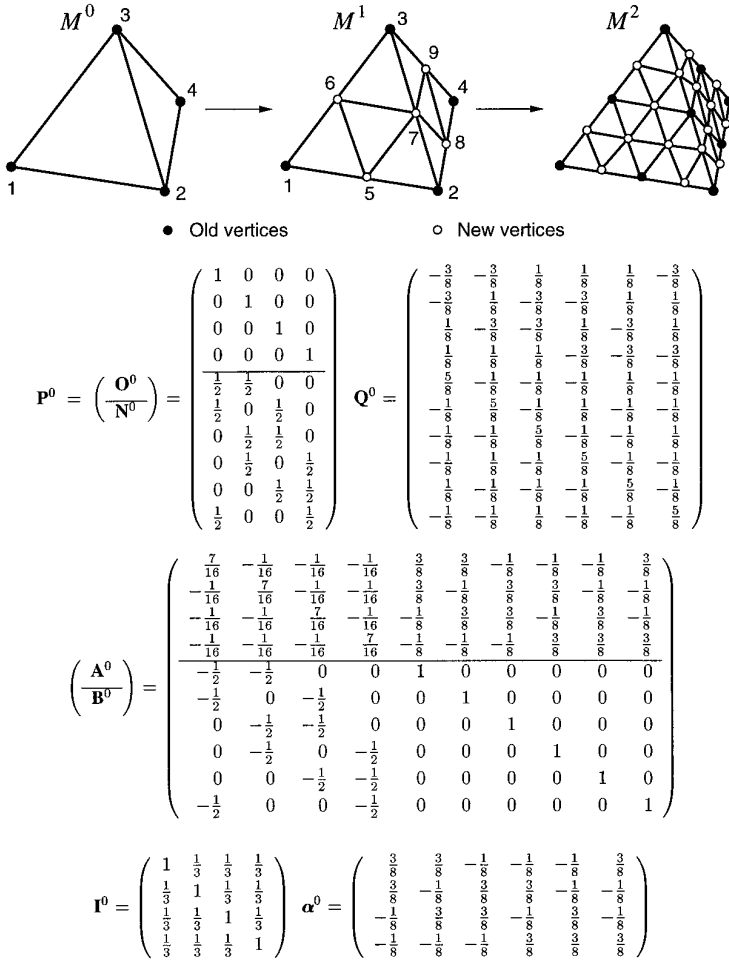


Fig. 2. Polyhedral subdivision of a tetrahedron. The matrices shown here represent various filters associated with the scheme, and are explained in Section 6.

extraordinary points (any original vertex of valence other than six) become increasingly isolated in an otherwise regular tiling of the surface.

Polyhedral subdivision converges to the original polyhedron covering M^0 , that is, to a C^0 surface. However, other schemes have been developed that converge to tangent-plane smooth limit surfaces that either approximate or interpolate the vertices of M^0 . Subdivision schemes can be further categorized as being either *primal* or *dual*. A subdivision scheme is primal if the faces of the mesh are split into subfaces by the refinement procedure. Catmull and Clark subdivision [1978; Halstead et al. 1993] is a primal scheme based on subdivision of quadrilateral faces. Polyhedral subdivision, the “butterfly” scheme of Dyn et al. [1990] and Loop’s method [1987] are primal schemes based on subdivision of triangular faces. A scheme is dual if the structure of the refined mesh is obtained by doing a primal subdivision followed by taking the polyhedral dual of the result. Doo and Sabin

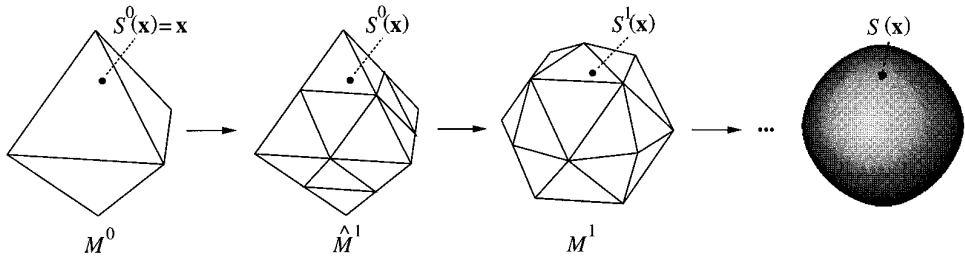


Fig. 3. The subdivision limiting process.

[1978] and Doo [1978] subdivision is a dual scheme based on quadrilaterals. Additionally, a *stationary* subdivision scheme is one in which there exists an integer j such that the same subdivision rule is used for refinement steps numbered larger than j .

Although one can certainly construct fractal-like subdivision schemes that do not converge to well defined surfaces, in this article we shall consider only local schemes that uniformly converge to a continuous surface no matter where the control vertices are placed. In the remainder of this article, subdivision schemes will be assumed to be local, stationary, continuous, and uniformly convergent. For simplicity we shall restrict the discussion to primal triangular subdivision schemes, although our results also hold for primal quadrilateral schemes.

4.2 Refinable Scaling Functions Through Subdivision

We show here that subdivision can be used to define a collection of refinable scaling functions. We do this by first showing that subdivision surfaces can be parametrized by a function $S(\mathbf{x})$, where \mathbf{x} ranges over M^0 . We then show that the parametrization can be used to define the scaling functions. Parametrizing the scaling functions over a domain M^0 of arbitrarily topological type differs sharply from the more usual method of parametrizing surface basis functions over a piece of the plane.

In general terms, a surface parametrization is nothing more than a correspondence between points in a two-dimensional domain and points on the surface. The idea behind establishing a parametrization for a subdivision surface is to track a point \mathbf{x} on M^0 through the subdivision process (see Figure 3). The point \mathbf{x} being tracked will converge to a point on the limit surface, thereby establishing a correspondence S between points on M^0 and points on the limit surface. We now make these ideas more precise.

For primal subdivision schemes, the refinement step that carries the mesh M^{s-1} into M^s consists of two substeps: the *splitting* step and the *averaging* step. In the splitting step, each face of M^{s-1} is split into four subfaces by introducing vertices at midpoints of edges, creating auxiliary mesh \hat{M}^s , as shown in Figure 3. The averaging step uses local weighted averaging to compute the vertex positions of M^s from the vertex positions of \hat{M}^s . All primal subdivision schemes share the splitting step—they differ only in the weights used in the averaging step.

The parametrization $S(\mathbf{x})$ can now be established by using a limiting process. The limiting process producing $S(\mathbf{x})$ consists of three steps:

- (1) $S^0(\mathbf{x}) := \mathbf{x}, \mathbf{x} \in M^0$.
- (2) Suppose that $S^{s-1}(\mathbf{x})$ lies in triangle $(\hat{v}_a^s, \hat{v}_b^s, \hat{v}_c^s)$ of \hat{M}^s with barycentric coordinates (α, β, γ) . Then

$$S^s(\mathbf{x}) := \alpha v_a^s + \beta v_b^s + \gamma v_c^s, \quad (2)$$

where (v_a^s, v_b^s, v_c^s) is the triangle of M^s corresponding to $(\hat{v}_a^s, \hat{v}_b^s, \hat{v}_c^s)$ of \hat{M}^s .

- (3) $S(\mathbf{x}) := \lim_{s \rightarrow \infty} S^s(\mathbf{x})$.

Next, we show that the parametrization $S(\mathbf{x})$ induces a collection of refinable scaling functions.

LEMMA 4.2.1. *For all $j \geq 0$, all $s \geq j$, and all i ranging over the vertices of M^j , there exist functions $\phi_i^{s \leftarrow j} : M^0 \rightarrow \mathbb{R}$ such that*

$$S^s(\mathbf{x}) = \sum_i v_i^j \phi_i^{s \leftarrow j}(\mathbf{x}).$$

PROOF. It is convenient to write the statement of the lemma in matrix form as

$$S^s(\mathbf{x}) = \Phi^{s \leftarrow j}(\mathbf{x}) \mathbf{V}^j,$$

where $\Phi^{s \leftarrow j}(\mathbf{x})$ is the row vector whose i th component is $\phi_i^{s \leftarrow j}(\mathbf{x})$.

The linear combination of Equation 2 above can be rewritten in matrix notation as

$$S^s(\mathbf{x}) = \mathbf{b}^s(\mathbf{x}) \mathbf{V}^s,$$

where $\mathbf{b}^s(\mathbf{x})$ is the barycentric coordinate vector of \mathbf{x} with respect to M^s ; that is,

$$\mathbf{b}^s(\mathbf{x}) = (0 \cdots 0\alpha 0 \cdots 0\beta 0 \cdots 0\gamma 0 \cdots 0),$$

where α occurs at index a , β occurs at index b , γ occurs at index c . At each refinement step $k = 1, \dots, s$, the vertices of M^k can be computed from affine combinations of the vertices of M^{k-1} . Therefore, there must exist a chain of (non-square) matrices $\mathbf{P}^0, \dots, \mathbf{P}^{k-1}$ such that

$$\mathbf{V}^k = \mathbf{P}^{k-1} \mathbf{P}^{k-2} \cdots \mathbf{P}^0 \mathbf{V}^0.$$

Thus,

$$S^s(\mathbf{x}) = \mathbf{b}^s(\mathbf{x}) \mathbf{P}^{s-1} \mathbf{P}^{s-2} \cdots \mathbf{P}^j \mathbf{V}^j.$$

The desired result follows immediately by making the definition

$$\Phi^{s \leftarrow j}(\mathbf{x}) := \mathbf{b}^s(\mathbf{x}) \mathbf{P}^{s-1} \mathbf{P}^{s-2} \dots \mathbf{P}^j. \quad \square$$

As a simple corollary to Lemma 1, we note that

$$\Phi^{s \leftarrow j}(\mathbf{x}) = \Phi^{s \leftarrow j+1}(\mathbf{x}) \mathbf{P}^j. \quad (3)$$

THEOREM 4.2.1. *For any subdivision procedure, and for any $j \geq 0$, there exist scalar-valued functions $\phi_i^j(\mathbf{x})$, $\mathbf{x} \in M^0$, such that*

$$S(\mathbf{x}) = \sum_i v_i^j \phi_i^j(\mathbf{x}). \quad (4)$$

PROOF. Using Lemma 1 and the definition of $S(\mathbf{x})$:

$$S(\mathbf{x}) = \lim_{s \rightarrow \infty} (\Phi^{s \leftarrow j} \mathbf{V}^j).$$

Since the subdivision scheme is assumed to be local and uniformly convergent for any choice of control points, $S(\mathbf{x})$ must exist if we choose all entries of \mathbf{V}^j to be the origin, except for v_i^j , the i th entry of \mathbf{V}^j . This choice of control points leads to a surface of the form

$$\left(\lim_{s \rightarrow \infty} \phi_i^{s \leftarrow j}(\mathbf{x}) v_i^j \right) = v_i^j \left(\lim_{s \rightarrow \infty} \phi_i^{s \leftarrow j}(\mathbf{x}) \right).$$

Since the surface is well defined, the sequence on the right must converge to $v_i^j \phi_i^j(\mathbf{x})$, where

$$\phi_i^j(\mathbf{x}) := \lim_{s \rightarrow \infty} \phi_i^{s \leftarrow j}(\mathbf{x}).$$

By linearity, the surface for an arbitrary set of control points can be written as in Equation 4. \square

We will find it convenient to rewrite Equation 4 in matrix form as

$$S(\mathbf{x}) = \Phi^j(\mathbf{x}) \mathbf{V}^j, \quad (5)$$

where $\Phi^j(\mathbf{x})$ denotes the row matrix of scaling functions $\phi_i^j(\mathbf{x})$, and where \mathbf{V}^j is as in Equation 1. Equation 5 shows an analogy with B-splines, where the $\phi_i^j(\mathbf{x})$ are comparable to the B-spline basis functions, and the \mathbf{V}^j are akin to the control points.

As a corollary to Theorem 4.2.1, if the subdivision scheme generates continuous surfaces, the scaling functions ϕ_i^j are continuous, hence they are also integrable [Bartle 1964].

We may now establish the refinability of the scaling functions defined in Theorem 4.2.1.

THEOREM 4.2.2. *The scaling functions $\phi_i^j(\mathbf{x})$ are refinable.*

PROOF. Starting with Equation 3 and taking limits as s tends toward infinity, it follows from the existence of the $\phi_i^j(\mathbf{x})$ that

$$\Phi^j(\mathbf{x}) = \Phi^{j+1}(\mathbf{x})\mathbf{P}^j. \quad (6)$$

This equation establishes refinability since it states that each of the functions $\phi_i^j(\mathbf{x})$ can be written as a linear combination of the functions $\phi_i^{j+1}(\mathbf{x})$. \square

For primal subdivision schemes, it is convenient to write Equation 6 in block matrix form by writing $\Phi^{j+1}(\mathbf{x})$ as

$$\Phi^{j+1}(\mathbf{x}) = (\mathcal{O}^{j+1}(\mathbf{x}) \quad \mathcal{N}^{j+1}(\mathbf{x})), \quad (7)$$

where $\mathcal{O}^{j+1}(\mathbf{x})$ consists of all scaling functions $\phi_i^{j+1}(\mathbf{x})$ associated with the old vertices of M^j (the black vertices in Figure 2) and $\mathcal{N}^{j+1}(\mathbf{x})$ consists of the remaining scaling functions associated with the new vertices added when obtaining M^{j+1} from M^j (the white vertices in Figure 2). Equation 6 can now be expressed in block matrix form:

$$\Phi^j(\mathbf{x}) = (\mathcal{O}^{j+1}(\mathbf{x}) \quad \mathcal{N}^{j+1}(\mathbf{x})) \begin{pmatrix} \mathbf{O}^j \\ \mathbf{N}^j \end{pmatrix}, \quad (8)$$

where \mathbf{O}^j and \mathbf{N}^j represent the portions of the subdivision matrix \mathbf{P}^j which weight the “old” and “new” vertices, respectively. The block matrix decomposition of \mathbf{P}^0 for the example tetrahedron appears in Figure 2.

4.3 Nested Linear Spaces

Given these relations, a chain of nested linear spaces $V^j(M^0)$ associated with a mesh M^0 can now be defined as follows:

$$V^j(M^0) := \text{Span}(\Phi^j(\mathbf{x})),$$

where the $V^j(M^0)$ are spaces of scalar-valued functions.

Equation 6 implies that these spaces are indeed nested; that is,

$$V^0(M^0) \subset V^1(M^0) \subset \dots$$

The notation $V^j(M^0)$ is to emphasize that the linear spaces are adapted to M^0 in that they consist of functions having M^0 as the domain.

5. INNER PRODUCTS

Given a chain of nested linear spaces, the other necessary ingredient for the creation of a multiresolution analysis is the existence of an inner product on these spaces. In this section, we define an inner product and give a method for exactly computing the inner product of any pair of functions defined through subdivision.

Inner products between pairs of scaling functions are used in the construction of wavelets, as described in Section 6. For an efficient implementation of subdivision wavelets, neither the inner products nor the wavelets need to be computed at run time, if the base mesh M^0 is known in advance.

5.1 Definition

Let two functions $f, g \in V^j(M^0)$, $j < \infty$ be linear combinations of (scalar-valued) scaling functions defined through subdivision, as in Section 4. We define the inner product of f and g to be

$$\langle f, g \rangle := \int_{\mathbf{x} \in M^0} f(\mathbf{x})g(\mathbf{x})d\mathbf{x}, \quad (9)$$

where the area form $d\mathbf{x}$ is taken to be the area for a triangulation homeomorphic to M^0 consisting of equilateral triangles with equal area. Equivalently, the inner product can be expressed as

$$\langle f, g \rangle := \sum_{\tau \in \Delta(M^0)} \frac{1}{\text{Area}(\tau)} \int_{\mathbf{x}' \in \tau} f(\mathbf{x}')g(\mathbf{x}')d\mathbf{x}',$$

where $\Delta(M^0)$ denotes the set of triangular faces of M^0 , and where $d\mathbf{x}'$ is the usual Euclidean area form for the triangle τ in \mathbb{R}^3 .

Our definition of inner product implies that triangles of different geometric size and shape are weighted equally; that is, the inner product is independent of the geometric positions of the vertices of M^0 . This inner product has two important consequences.

First, in the process of constructing the least-squares best wavelet approximation to a function, each approximated triangle is weighted equally, independent of its geometric size. Although this simplification ignores the fact that for most data sets all triangles are not really the same size, we have obtained good results for many examples, including those described in Section 7.5.

Second, because all triangles can be treated equally, the wavelet spaces are invariant of the geometry of the mesh and filters only change at the extraordinary points. Thus, a significant amount of precomputation of inner products and wavelets can be performed, allowing the wavelet algorithms to be implemented more efficiently.

An alternative is to define the inner product so as to weight the integral by the areas of triangles in M^0 . Whether such a definition has enough important practical benefit to offset its much increased computation may be an interesting topic for future research. A step in this direction has already been made by Schröder and Sweldens [1995].

5.2 Computation

For piecewise linear subdivision (leading to polyhedral surfaces), the scaling functions $\phi_i^j(\mathbf{x})$ are simply the hat functions over M^0 . For the case when

functions f and g are combinations of piecewise linear scaling functions, it is therefore fairly simple to directly compute the integral of Equation 9.

In general, however, the lack of a closed form for the scaling functions makes explicit integration quite difficult. In these cases, one could estimate the inner product $\langle f, g \rangle$ by subdividing the scaling functions some number of times and directly integrating the resulting piecewise linear approximation. In this section, we will see that such estimation is unnecessary, and that exact integration is still possible for the general case.

One way to compute inner products is to follow the approach in Halstead et al. [1993]. Their approach for computing integrals over subdivision surfaces was to observe that away from extraordinary points, integrals could be computed exactly since Catmull-Clark subdivision converges to uniform bicubic B-splines in *regular* regions (those regions removed from extraordinary points). They also observed that the subdivision process could then be used to compute an integral over the entire surface by evaluating a geometric series. To generalize their method to other schemes, one must first compute inner products in regular regions. This is relatively easy for some schemes, such as Loop's, that converge to polynomials in regular regions, but it is harder for schemes such as the butterfly method that are nowhere polynomial.

A general method for integration in regular regions was recently developed by Dahmen and Micchelli [1993]. It operates by reducing the problem of computing inner products to one of computing eigenvectors of a matrix defined by the refinement equations.

Although it is possible to combine the methods of Halstead et al. [1993] and Dahmen and Micchelli [1993], we present here a method that is somewhat simpler, in that it computes inner products directly as the solution to a homogeneous system of linear equations.

Let f and g be functions given as expansions in ϕ_i^j :

$$f(\mathbf{x}) = \sum_i f_i^j \phi_i^j(\mathbf{x}) \quad g(\mathbf{x}) = \sum_i g_i^j \phi_i^j(\mathbf{x}).$$

Bilinearity of the inner product allows $\langle f, g \rangle$ to be written in matrix form as

$$\langle f, g \rangle = \mathbf{g}^T \mathbf{I}^j \mathbf{f},$$

where \mathbf{f} and \mathbf{g} are column matrices consisting of the coefficients of f and g , respectively, and where \mathbf{I}^j is the square matrix whose i, i' -th entry is $(\mathbf{I}^j)_{i,i'} = \langle \phi_i^j, \phi_{i'}^j \rangle$.

The i th row of \mathbf{I}^j contains the inner product of ϕ_i^j with each of the other scaling functions $\phi_{i'}^j$. It is convenient to view these entries geometrically by constructing an *inner product mask* around each vertex. The inner product mask for the i th vertex of M^j assigns to each vertex i' of M^j a multiple of the scalar $\langle \phi_i^j, \phi_{i'}^j \rangle$.

In the case of polyhedral subdivision, explicit calculation leads to the inner product mask around a vertex of valence n shown in Figure 4, where

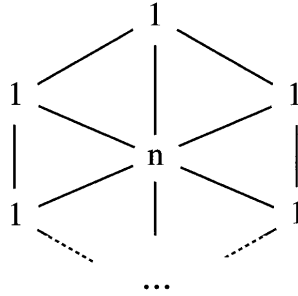


Fig. 4. The polyhedral inner product mask around a vertex of degree n .

the central weight is simply the valence of the central vertex. As an example, the complete inner product matrix \mathbf{I}^0 for the tetrahedron appears in Figure 2. Note that in this case the mask entries must be divided by 3 to obtain the matrix entries.

For more general subdivision procedures, such as the butterfly scheme, the limit surface has no closed form, precluding a brute-force explicit integration. However, as we now show, it is possible to exactly determine the entries of \mathbf{I}^j by solving a linear system, without resorting to numerical integration. We will first present the general integration procedure, and then illustrate it using a simple one-dimensional example.

The key to the linear system is to observe that when all triangles at subdivision level j are weighted equally, as was done in the inner product formulation in Section 5.1, a recurrence relation exists between \mathbf{I}^j and \mathbf{I}^{j+1} . Specifically, \mathbf{I}^j can be written as

$$\mathbf{I}^j = \int_{\mathbf{x} \in M^0} (\Phi^j(\mathbf{x}))^T \Phi^j(\mathbf{x}) d\mathbf{x}, \tag{10}$$

where the integrand represents a matrix outer product, and where the integral of a matrix of functions is defined to be the matrix of integrals. The refinement property of Equation 6 can now be used to establish the recurrence

$$\begin{aligned} \mathbf{I}^j &= \int_{\mathbf{x} \in M^0} (\mathbf{P}^j)^T (\Phi^{j+1}(\mathbf{x}))^T \Phi^{j+1}(\mathbf{x}) \mathbf{P}^j d\mathbf{x} \\ &= (\mathbf{P}^j)^T \mathbf{I}^{j+1} \mathbf{P}^j. \end{aligned}$$

Since the subdivision rules are local, the corresponding scaling functions are locally supported, and the support of a scaling function $\phi_i^j(\mathbf{x})$ shrinks as j increases. Thus, beyond some level j' the support of each of the scaling functions $\phi_i^{j'}$ will contain at most one extraordinary point. Furthermore, the local neighborhood at level $j' + 1$ will be a shrunken version of the local neighborhood at level j' . Hence, the scaling functions in $\Phi^{j'}(\mathbf{x})$ and $\Phi^{j'+1}(\mathbf{x})$ are closely related.

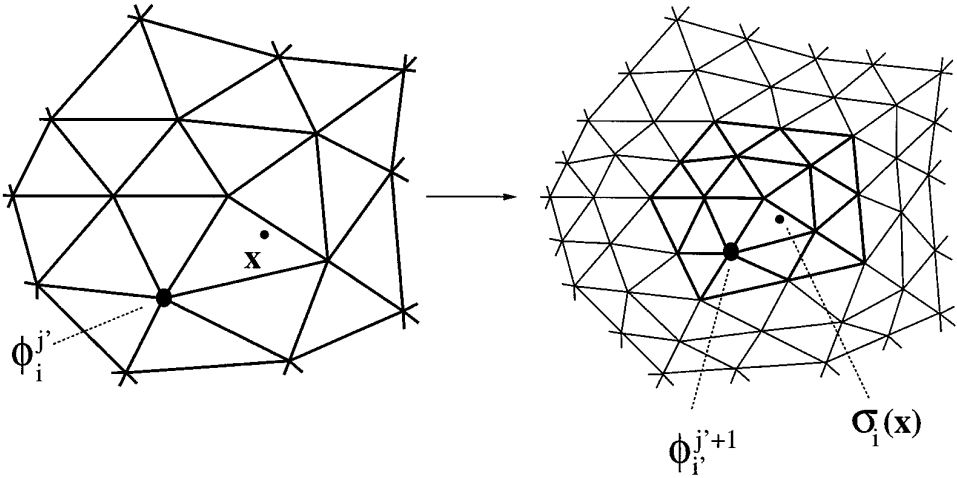


Fig. 5. The diagram on the left depicts the triangulation of M^0 produced after j' subdivision steps, that is, after j' recursive midpoint splits; the diagram on the right corresponds to $j' + 1$ subdivision steps. $\phi_i^{j'}$ denotes the scaling function for the i -th vertex of $M^{j'}$, and $\phi_{i'}^{j'+1}$ denotes the scaling function for the corresponding i' -th vertex of $M^{j'+1}$. The map σ_i is such that the barycentric coordinates of \mathbf{x} and $\sigma_i(\mathbf{x})$ within their respective surrounding triangles are equal.

More precisely, for each scaling function $\phi_i^{j'}$ there is at least one i' such that

$$\phi_i^{j'}(\mathbf{x}) = \phi_{i'}^{j'+1}(\sigma_i(\mathbf{x})),$$

where $\sigma_i : M^0 \rightarrow M^0$ is a piecewise linear function that maps triangles in the support of $\phi_i^{j'}$ into the corresponding triangles of $\phi_{i'}^{j'+1}$, as depicted in Figure 5. As a consequence of the parametric construction in Section 4.2, triangle areas of M^0 shrink by a factor of 4 under subdivision, and hence the Jacobian of σ_i is $1/4$. Each of the entries $(\mathbf{I}^{j'})_{hi}$ in $\mathbf{I}^{j'}$ therefore has one or more corresponding entries $(\mathbf{I}^{j'+1})_{h'i'}$ in $\mathbf{I}^{j'+1}$, up to a factor of $1/4$; that is, $1/4 (\mathbf{I}^{j'})_{hi} = (\mathbf{I}^{j'+1})_{h'i'}$.

The resulting $m \times m$ matrix equation

$$\mathbf{I}^{j'} = (\mathbf{P}^{j'})^T \mathbf{I}^{j'+1} \mathbf{P}^{j'} \quad (11)$$

represents a homogeneous system of m^2 equations in the m^2 unknown entries of $\mathbf{I}^{j'}$. Due to the symmetry of $\mathbf{I}^{j'}$, the system reduces to $m(m + 1)/2$ homogeneous equations in as many unknowns. A square inhomogeneous system is produced once an absolute scale is chosen for the homogeneous system. We typically set the scale by requiring that the sum of the entries of $\mathbf{I}^{j'}$ is 1—this is equivalent to selecting an area form that assigns unit area to M^0 . Although we have been unable to prove that the system is uniquely solvable (equivalently, that the system matrix is of full rank), this has been true in hundreds of cases we have tried, including all those used to generate the figures in this article. We therefore conjecture that under mild conditions on the subdivision scheme the system is uniquely solvable.

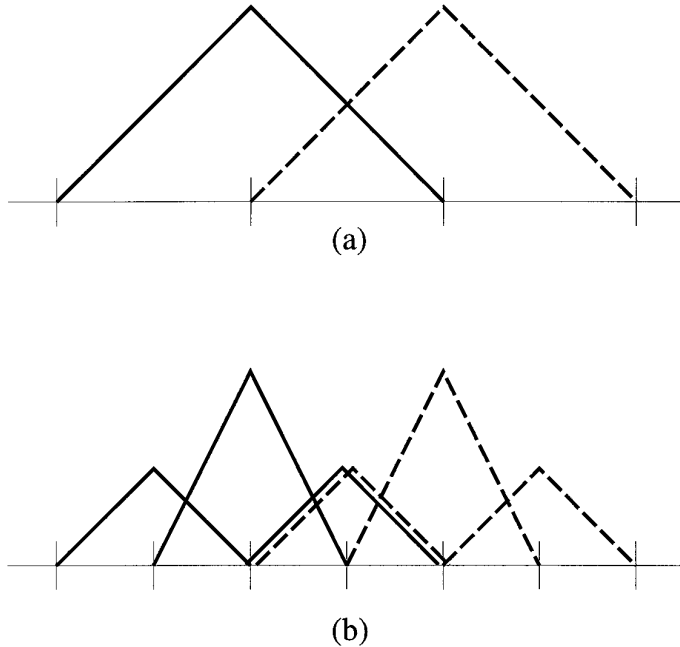


Fig. 6. Computing the inner product $\langle \phi_i^j(x), \phi_{i+1}^j(x) \rangle$ for piecewise linear scaling functions. (a) the graph of $\phi_i^j(x)$ is drawn using solid lines, and the graph of $\phi_{i+1}^j(x)$ is drawn using dashed lines; (b) $\phi_i^j(x)$ and $\phi_{i+1}^j(x)$ after refinement. The original problem reduces to that of computing the nine inner products between a solid function and a dashed function.

Once the entries of $\mathbf{I}^{j'}$ have been determined, the remaining inner product matrices $\mathbf{I}^{j'-1}, \mathbf{I}^{j'-2}, \dots, \mathbf{I}^0$ can be successively determined via Equation 11.

As a simple example of the integration process, consider the one-dimensional case where the scaling functions $\phi_i^j(x)$ are piecewise linear “hat” functions parametrized over the infinite real line, as illustrated in Figure 6. For this case, a scaling function ϕ_i^j with knots on the even integers can be refined according to:

$$\phi_i^j(x) = \frac{1}{2} \phi_{2i-1}^{j+1}(x) + \phi_{2i}^{j+1}(x) + \frac{1}{2} \phi_{2i+1}^{j+1}(x) \tag{12}$$

where scaling functions at level $j + 1$ are hat functions with knots on the integers.

For any level j , these functions lead to only two nonzero cases of inner products:

- (1) $\langle \phi_i^j(x), \phi_i^j(x) \rangle$ —the inner product of a scaling function with itself.
- (2) $\langle \phi_i^j(x), \phi_{i+1}^j(x) \rangle$ —the inner product of a scaling function with its immediate neighbor.

All other possible inner products are either equivalent to one of these, or zero, due to the local support of $\phi_i^j(x)$.

Choosing our scale such that case 1 evaluates to 1, explicit integration of the piecewise linear functions shows that the inner product for case 2 is $\frac{1}{4}$. We now determine these same integrals using the general integration technique.

We define the unknown inner product values for case 1 and case 2 to be

$$\begin{aligned} x_1 &:= \langle \phi_i^j(x), \phi_i^j(x) \rangle \\ x_2 &:= \langle \phi_i^j(x), \phi_{i+1}^j(x) \rangle. \end{aligned} \quad (13)$$

Figure 6(a) illustrates case 2—the inner product of two neighboring piecewise linear scaling functions at level j with knots on the even integers.

The next step is to refine with Equation 12:

$$x_2 = \left\langle \frac{1}{2} \phi_{2i-1}^{j+1}(x) + \phi_{2i}^{j+1}(x) + \frac{1}{2} \phi_{2i+1}^{j+1}(x), \right. \\ \left. \frac{1}{2} \phi_{2i+1}^{j+1}(x) + \phi_{2i+2}^{j+1}(x) + \frac{1}{2} \phi_{2i+3}^{j+1}(x) \right\rangle.$$

Bilinearity of inner products allows this to be expanded to:

$$\begin{aligned} x_2 &= \frac{1}{4} \langle \phi_{2i-1}^{j+1}(x), \phi_{2i+1}^{j+1}(x) \rangle + \frac{1}{2} \langle \phi_{2i-1}^{j+1}(x), \phi_{2i+2}^{j+1}(x) \rangle \\ &\quad + \frac{1}{4} \langle \phi_{2i-1}^{j+1}(x), \phi_{2i+3}^{j+1}(x) \rangle \\ &\quad + \frac{1}{2} \langle \phi_{2i}^{j+1}(x), \phi_{2i+1}^{j+1}(x) \rangle + \langle \phi_{2i}^{j+1}(x), \phi_{2i+2}^{j+1}(x) \rangle + \frac{1}{2} \langle \phi_{2i}^{j+1}(x), \phi_{2i+3}^{j+1}(x) \rangle \\ &\quad + \frac{1}{4} \langle \phi_{2i+1}^{j+1}(x), \phi_{2i+1}^{j+1}(x) \rangle + \frac{1}{2} \langle \phi_{2i+1}^{j+1}(x), \phi_{2i+2}^{j+1}(x) \rangle \\ &\quad + \frac{1}{4} \langle \phi_{2i+1}^{j+1}(x), \phi_{2i+3}^{j+1}(x) \rangle. \end{aligned}$$

The result after refinement is shown in Figure 6(b). Through refinability, Equation 13 has been expressed in terms of the nine possible inner products involving a solid function and a dashed-line function. Most of these terms drop out because the supports of their respective functions do not overlap:

$$x_2 = \frac{1}{2} \langle \phi_i^{j+1}(x), \phi_{i+1}^{j+1}(x) \rangle + \frac{1}{4} \langle \phi_{i+1}^{j+1}(x), \phi_{i+1}^{j+1}(x) \rangle + \frac{1}{2} \langle \phi_{i+1}^{j+1}(x), \phi_{i+2}^{j+1}(x) \rangle. \quad (14)$$

The inner products in the first and last terms of Equation 14 are equivalent to case 2. However, they are parametrized over a narrower domain, and are therefore reduced by a scale of $\frac{1}{2}$ (recall that the appropriate scale for surfaces is $\frac{1}{4}$). Likewise, the inner product of the middle term is similar to case 1. We can therefore rewrite Equation 14 in terms of the unknowns x_1 and x_2 :

$$x_2 = \frac{1}{8} x_1 + \frac{1}{2} x_2.$$

Similar analysis of case 1 yields:

$$x_1 = \frac{3}{4} x_1 + x_2.$$

After subtracting out the unknowns on the left, these equations can be written as a homogeneous linear system of the form

$$\begin{pmatrix} \frac{1}{8} & -\frac{1}{2} \\ -\frac{1}{4} & 1 \end{pmatrix}.$$

Arbitrarily setting $x_1 := 1$ again yields the inner product for case 2: $x_2 = \frac{1}{4}$. This agrees with the result obtained earlier through explicit integration.

6. MULTIREOLUTION ANALYSIS BASED ON SUBDIVISION

In previous sections we have established nested linear spaces and an inner product relative to a subdivision rule. We are now in a position to construct wavelets, that is, a set of functions $\Psi^j(\mathbf{x}) = (\psi_1^j(\mathbf{x}), \psi_2^j(\mathbf{x}), \dots)$ that span wavelet spaces $W^j(M^0)$.

It is of significant practical importance that the analysis and synthesis filters associated with these wavelets are constructed and applied in linear time. This practical concern drives much of the development in this section, the remainder of which is structured as follows. In Section 6.1 we first describe a simple construction leading to semiorthogonal wavelets. Although these wavelets do not satisfy the linear time requirements, a variant of the construction is used in Section 6.2 to produce biorthogonal wavelets. In Section 6.3 it is shown that for interpolating subdivision schemes the biorthogonal wavelets possess linear time analysis and synthesis procedures.

6.1 The Semiorthogonal Construction

Our construction of semiorthogonal wavelets consists of two steps. First, we build a basis for $V^{j+1}(M^0)$ using the scaling functions $\Phi^j(\mathbf{x})$ together with

the new scaling functions $\mathcal{N}^{j+1}(\mathbf{x})$ in $V^{j+1}(M^0)$. It is straightforward to show that $\Phi^j(\mathbf{x})$ and $\mathcal{N}^{j+1}(\mathbf{x})$ together span $V^{j+1}(M^0)$ if and only if the matrix \mathbf{O}^j (encoding the subdivision rule around the old vertices) is invertible. Most primal subdivision methods, such as polyhedral subdivision and the butterfly scheme, have this property.¹

Given a function $S^{j+1}(\mathbf{x})$ in $V^{j+1}(M^0)$ expressed as an expansion in the basis $(\Phi^j(\mathbf{x}), \mathcal{N}^{j+1}(\mathbf{x}))$, an approximation in $V^j(M^0)$ can be obtained by *restriction to $\Phi^j(\mathbf{x})$* , that is, by setting to zero the coefficients corresponding to $\mathcal{N}^{j+1}(\mathbf{x})$. However, this generally does not produce the best least-squares approximation. To ensure the best least-squares approximation after restriction to $\Phi^j(\mathbf{x})$, we may orthogonalize the new basis functions $\mathcal{N}^{j+1}(\mathbf{x})$ by subtracting out their least-squares projection into $V^j(M^0)$. Expressed in matrix form:

$$\Psi^j(\mathbf{x}) = \mathcal{N}^{j+1}(\mathbf{x}) - \Phi^j(\mathbf{x})\alpha^j. \quad (15)$$

The coefficients α^j are the solution to the linear system formed by taking the inner product of each side of Equation 15 with $\Phi^j(\mathbf{x})$, and using the fact that $\langle \Phi^j(\mathbf{x}), \Psi^j(\mathbf{x}) \rangle = 0$:

$$\begin{aligned} \langle \Phi^j(\mathbf{x}), \Phi^j(\mathbf{x}) \rangle \alpha^j &= \langle \Phi^j(\mathbf{x}), \mathcal{N}^{j+1}(\mathbf{x}) \rangle, \\ &= (\mathbf{P}^j)^T \langle \Phi^{j+1}(\mathbf{x}), \mathcal{N}^{j+1}(\mathbf{x}) \rangle \end{aligned} \quad (16)$$

where $\langle \mathbf{F}, \mathbf{G} \rangle$ stands for the matrix whose i, i' th entry is $\langle (\mathbf{F})_i, (\mathbf{G})_{i'} \rangle$. The matrix $\langle \Phi^j(\mathbf{x}), \Phi^j(\mathbf{x}) \rangle$ is therefore simply \mathbf{I}^j , and the matrix $\langle \Phi^{j+1}(\mathbf{x}), \mathcal{N}^{j+1}(\mathbf{x}) \rangle$ is a submatrix of \mathbf{I}^{j+1} consisting of those columns corresponding to members of $\mathcal{N}^{j+1}(\mathbf{x})$. The system of equations may be solved for the coefficients α^j . As an example, Figure 2 shows the matrix α^0 for the tetrahedron.

Although this construction produces semiorthogonal wavelets—that is, the wavelets $\Psi^j(\mathbf{x})$ are orthogonal to the scaling functions in $V^j(M^0)$ —they are not practically useful because analysis and synthesis both require quadratic time. To see why, note that the inner product matrix \mathbf{I}^j in Equation 16 must be inverted. Although \mathbf{I}^j is sparse, its inverse is dense, implying that the wavelets are globally supported on M^0 . The synthesis matrices \mathbf{Q}^j mentioned in Section 2 and more fully described in Section 6.3 are therefore dense, leading to quadratic time synthesis. The methods used in Section 6.3 can be used to show that the analysis matrices \mathbf{A}^j are also dense, making analysis a quadratic time process as well.

As we show in the next two sections, linear time analysis and synthesis can be achieved, at least for interpolating subdivision schemes, by modifying the above construction to produce biorthogonal wavelets.

¹One notable exception is Catmull-Clark subdivision for vertices of valence three. However, the subdivision rule for such vertices can be easily modified to produce an invertible matrix.

6.2 The Biorthogonal Construction

Linear time synthesis can be achieved by using locally supported wavelets, the construction of which is a common problem in the wavelet literature, and has been handled in various ways. For instance, Daubechies [1988] uses Fourier techniques to develop locally supported fully orthogonal wavelets, and Chui [1992] relies on the rich theory of B-splines to develop minimally supported semiorthogonal spline wavelets. However, neither of these approaches is appropriate for our use: Fourier techniques rely on translation and dilation, and B-splines rely on planar topologies.

We currently do not know of an orthogonal or semiorthogonal construction possessing linear time analysis and synthesis for arbitrary topological domains. Our approach is to relax semiorthogonality and construct instead biorthogonal wavelets. That is, we no longer require the wavelets $\Psi^j(\mathbf{x})$ to be orthogonal to $V^j(M^0)$, but to preserve good approximation properties, we require the wavelets to be “as orthogonal as possible” subject to the linear time requirement. (We should note that a similar approach has been taken in independent work by Dahmen et al. [1993].)

Our biorthogonal construction proceeds by selecting the supports of the wavelets *a priori*. The idea is to “partially orthogonalize” each new scaling function $\phi_i^{j+1}(\mathbf{x})$ in $\mathcal{N}^{j+1}(\mathbf{x})$ by subtracting off a locally supported least-squares best projection of it into $V^j(M^0)$. Stated another way, we allow each column of the matrix α^j in Equation 15 to have only a constant number of nonzero entries. We then determine the nonzero entries of α^j by minimizing the L^2 norm of the projection of $\Phi^j(\mathbf{x})$ into $V^j(M^0)$.

More concretely, let $\phi_i^{j+1}(\mathbf{x})$ denote a new scaling function, and let \mathcal{F}_i^j denote any subset of the indices of the scaling functions in $V^j(M^0)$; typically \mathcal{F}_i^j consists of the indices of scaling functions in $V^j(M^0)$ supported in some neighborhood of $\phi_i^{j+1}(\mathbf{x})$. We then take the wavelet $\psi_i^j(\mathbf{x})$ to be

$$\psi_i^j(\mathbf{x}) = \phi_i^{j+1}(\mathbf{x}) + \sum_{i' \in \mathcal{F}_i^j} \alpha_{i',i}^j \phi_{i'}^j(\mathbf{x}) \quad (17)$$

where the coefficients $\alpha_{i',i}^j$ are such that the L^2 norm of the projection of $\psi_i^j(\mathbf{x})$ into $V^j(M^0)$ is minimized. These coefficients are therefore determined by solving the following local linear system:

$$\sum_{i' \in \mathcal{F}_i^j} \alpha_{i',i}^j \langle \phi_{i'}^j(\mathbf{x}), \phi_k^j(\mathbf{x}) \rangle = \langle \phi_i^{j+1}(\mathbf{x}), \phi_k^j(\mathbf{x}) \rangle, \quad (18)$$

for all k in \mathcal{F}_i^j .

One way to control the support of the wavelets in a symmetric fashion is to select the index set \mathcal{F}_i^j as follows. The new vertex v_i^{j+1} of M^{j+1} is associated with an edge $u - v$ of M^j . For instance, the vertex numbered 7 of M^1 in Figure 2 is associated with the edge 2 - 3 of M^0 . We take \mathcal{F}_i^j to be the vertex indices of M^j in the k -discs of u and v . (The k -disc around a vertex v of a triangulation is defined to be the set of all vertices reachable from v by following k or fewer edges of the triangulation.) Except for Tables

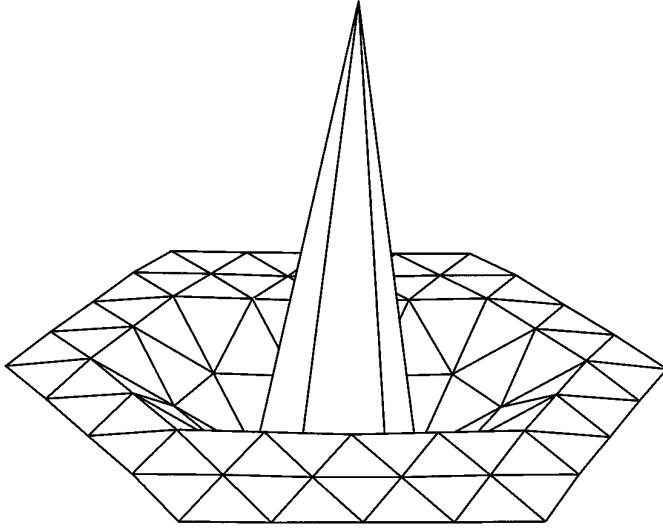


Fig. 7. A polyhedral wavelet centered on a vertex of valence 6.

II and III, all examples in this article were computed using 2-discs. Figure 7 is a plot of a polyhedral wavelet computed using 2-discs.

As the size of the discs increases, the wavelet supports grow, and the wavelets have empirically been observed to rapidly approach semi-orthogonality.

6.3 A Filter Bank Algorithm

Multiresolution analysis on the infinite real line is based on an assumption of spatial invariance—intuitively, every place looks like every other place. This means that the analysis and synthesis filters can be represented by a convolution kernel, that is, by a sequence of real numbers. This is not the case for multiresolution analysis on arbitrary topological domains. The filter coefficients in general must vary over the mesh, so the filters are represented by (hopefully sparse) matrices.

The analysis and synthesis filters can be conveniently expressed using block matrix equations. For any multiresolution analysis the synthesis filters are defined by the relation

$$(\Phi^j(\mathbf{x}) \Psi^j(\mathbf{x})) = \Phi^{j+1}(\mathbf{x}) (\mathbf{P}^j \quad \mathbf{Q}^j), \quad (19)$$

and the analysis filters are obtained from the inverse relation

$$\begin{pmatrix} \mathbf{A}^j \\ \mathbf{B}^j \end{pmatrix} = (\mathbf{P}^j \quad \mathbf{Q}^j)^{-1}. \quad (20)$$

The analysis filters can be used to decompose a surface $S^{j+1}(\mathbf{x})$ in

$V^{j+1}(M^0)$ given by

$$S^{j+1}(\mathbf{x}) = \sum_i v_i^{j+1} \phi_i^{j+1}(\mathbf{x}) \quad (21)$$

into a lower resolution part in $V^j(M^0)$ plus a detail part in $W^j(M^0)$

$$S^{j+1}(\mathbf{x}) = \sum_i v_i^j \phi_i^j(\mathbf{x}) + \sum_i w_i^j \psi_i^j(\mathbf{x})$$

as follows. Let \mathbf{V}^j be as in Equation 5, and let \mathbf{W}^j denote the corresponding matrix of wavelet coefficients w_i^j . We can rewrite Equation 21 in matrix form and substitute the definition of the analysis filters. Thus:

$$\begin{aligned} S^{j+1}(\mathbf{x}) &= \Phi^{j+1}(\mathbf{x}) \mathbf{V}^{j+1} \\ &= (\Phi^j(\mathbf{x}) \Psi^j(\mathbf{x})) \begin{pmatrix} \mathbf{A}^j \\ \mathbf{B}^j \end{pmatrix} \mathbf{V}^{j+1} \\ &= \Phi^j(\mathbf{x}) \mathbf{A}^j \mathbf{V}^{j+1} + \Psi^j(\mathbf{x}) \mathbf{B}^j \mathbf{V}^{j+1} \end{aligned}$$

therefore,

$$\mathbf{V}^j = \mathbf{A}^j \mathbf{V}^{j+1} \quad (22)$$

$$\mathbf{W}^j = \mathbf{B}^j \mathbf{V}^{j+1}. \quad (23)$$

Of course, the analysis filters \mathbf{A}^{j-1} and \mathbf{B}^{j-1} can now be applied to \mathbf{V}^j to yield \mathbf{V}^{j-1} , \mathbf{W}^{j-1} , etc. A similar argument shows that \mathbf{V}^{j+1} can be recovered from \mathbf{V}^j and \mathbf{W}^j using the synthesis filters:

$$\mathbf{V}^{j+1} = \mathbf{P}^j \mathbf{V}^j + \mathbf{Q}^j \mathbf{W}^j. \quad (24)$$

We shall now develop more explicit expressions for the analysis and synthesis filters. It is again convenient to write $\Phi^{j+1}(x)$ in block form as

$$(\mathbb{C}^{j+1}(\mathbf{x}), \mathcal{N}^{j+1}(\mathbf{x})).$$

It then follows from Equation 15 that the synthesis filters can be written in block form as

$$(\mathbf{P}^j \quad \mathbf{Q}^j) = \begin{pmatrix} \mathbf{O}^j & -\mathbf{O}^j \boldsymbol{\alpha}^j \\ \mathbf{N}^j & \mathbf{1} - \mathbf{N}^j \boldsymbol{\alpha}^j \end{pmatrix}, \quad (25)$$

where $\mathbf{1}$ denotes the identity matrix.

The analysis filters are obtained from Equation 20. (Examples of analysis and synthesis matrices for the tetrahedron are shown in Figure 2.) To achieve linear time analysis and synthesis, the analysis and synthesis matrices must be sparse, having only a constant number of nonzero entries

Table I. Time to construct analysis filters for various subdivision methods.

Method	Continuity	Asymptotic run time
Polyhedral	C^0	$O(n)$
Butterfly	G^1	$O(n)$
Loop	G^1	Speed of sparse linear equation solution
Catmull & Clark	G^1	Speed of sparse linear equation solution

in each row. If \mathbf{P}^j and $\boldsymbol{\alpha}^j$ are sparse, then from Equation 25 \mathbf{Q}^j is also sparse. Unfortunately, the analysis filters derived from Equation 20 need not be sparse. For interpolating subdivision schemes such as polyhedral subdivision and the butterfly scheme, the situation is much improved. Such interpolating schemes have the property that \mathbf{O}^j is the identity matrix. In this case the resulting filters are

$$(\mathbf{P}^j \quad \mathbf{Q}^j) = \begin{pmatrix} \mathbf{1} & -\boldsymbol{\alpha}^j \\ \mathbf{N}^j & \mathbf{1} - \mathbf{N}^j \boldsymbol{\alpha}^j \end{pmatrix} \quad \begin{pmatrix} \mathbf{A}^j \\ \mathbf{B}^j \end{pmatrix} = \begin{pmatrix} \mathbf{1} - \boldsymbol{\alpha}^j \mathbf{N}^j & \boldsymbol{\alpha}^j \\ -\mathbf{N}^j & \mathbf{1} \end{pmatrix}.$$

If \mathbf{P}^j and $\boldsymbol{\alpha}^j$ are sparse, then all four filters are also sparse, leading to linear time analysis and synthesis. The situation is less desirable for methods related to B-splines, such as Loop's scheme and Catmull-Clark surfaces. For these subdivision schemes, the synthesis filters are sparse, but the analysis filters are dense. This implies that synthesis is still possible in $O(n)$ time, but that the speed of analysis depends on the time to invert the sparse analysis matrix of Equation 20, or to solve the related sparse linear system. Whether these methods can be made efficient for multiresolution analysis is a topic for future investigation. Table I shows what we currently know about the time required to develop analysis filters for various subdivision methods.

The filter bank process specialized to the case of piecewise linear functions is presented in Section 7.1.

6.4 Stability

One important question concerns the stability of the synthesis and analysis filters. One measure of the stability of a transformation is its L^∞ norm. The L^∞ norm for any matrix T , denoted by $\|T\|_\infty$, is the maximum sum of the absolute values of the elements in any row of T , and indicates the maximum scaling effect of T on any vector that it transforms.

For fully orthogonal wavelet bases over regular grids, the L^∞ norm is bounded by a constant, independent of the number of filtering steps. In this section, we briefly analyze the L^∞ norms of the analysis and synthesis filters our locally supported biorthogonal wavelets.

We first consider repeated application of the synthesis filters. This transformation maps the coarse-level scaling coefficients \mathbf{V}^0 and the wavelet coefficients $\mathbf{W}^0, \dots, \mathbf{W}^{n-1}$ into the fine-level scaling coefficients \mathbf{V}^n .

Expanding equation 24 repeatedly yields

$$\mathbf{V}^n = \left(\prod_{j=0}^{n-1} \mathbf{P}^j \right) \mathbf{V}^0 + \sum_{j'=0}^{n-1} \left(\prod_{j=j'}^{n-1} \mathbf{P}^j \right) \mathbf{Q}^{j'} \mathbf{W}^{j'}.$$

The L^∞ norm of this transformation depends on the L^∞ norm of the product $\prod_{j=0}^{n-1} \mathbf{P}^j$. We claim that this norm is bounded independent of n . By definition, convergence of the subdivision scheme defined by the \mathbf{P}^j implies that the entries of $\prod_{j=0}^{n-1} \mathbf{P}^j$ are also bounded independent of n . Since the number of entries per row of $\prod_{j=0}^{n-1} \mathbf{P}^j$ is independent of n , the L^∞ norm of $\prod_{j=0}^{n-1} \mathbf{P}^j$ is bounded independent of n . To conclude, we note that \mathbf{V}^n is the sum of $n + 1$ transformations with bounded L^∞ norms. Therefore, the norm for the entire transformation is at most $O(n)$. As a common special case, when the subdivision rule encoded by \mathbf{P}^j is a convex combination (such as with piecewise linear subdivision or with the Catmull-Clark scheme), the L^∞ norm of any \mathbf{P}^j is exactly 1, implying that the L^∞ norm of their product is also 1.

We next consider repeated application of the analysis filter. This transformation maps the fine scaling coefficients \mathbf{V}^n into the coarse scaling coefficients \mathbf{V}^0 and the wavelet coefficients $\mathbf{W}^0, \dots, \mathbf{W}^{n-1}$. Expanding equations 22 and 23 yields

$$\mathbf{V}^0 = \prod_{j=0}^{n-1} \mathbf{A}^j \mathbf{V}^n.$$

$$\mathbf{W}^j = \mathbf{B}^j \prod_{j'=j+1}^{n-1} \mathbf{A}^{j'} \mathbf{V}^n.$$

The stability of these transformations depends on the L^∞ norm of $\prod_{j=0}^{n-1} \mathbf{A}^j$. Note that the product of these matrices is the projection operator that maps \mathbf{V}^n into \mathbf{V}^0 . We conjecture that the norm for this transformation is also bounded independent of n .

In practice, the norms for these filters appear to be very small, implying good stability. Table II gives the L^∞ norms for the analysis filter \mathbf{A}^j over a varying number of levels using a varying size support. These \mathbf{A}^j are built using the wavelet approximations described above, for the case of piecewise linear subdivision over the octahedron. In the table, *size* gives the disc size chosen for the support of the wavelet approximations. (The k -disc around a vertex v of a triangulation is defined to be the set of all triangles whose vertices are reachable from v by following k or fewer edges of the triangulation.) All examples in this article were generated using wavelet approximations supported on 2-discs.

Table II. L^∞ Norms for the Analysis Filters \mathbf{A}^j that Arise from Piecewise Linear Subdivision on the Octahedron.

	1-disc	2-disc	3-disc
$\ \mathbf{A}^0\ _\infty$	1.25	1.3125	1.3125
$\ \mathbf{A}^1\ _\infty$	1.27174	1.40771	1.4434
$\ \mathbf{A}^2\ _\infty$	1.27174	1.40859	1.45371
$\ \mathbf{A}^3\ _\infty$	1.27174	1.40859	1.45372
$\ \mathbf{A}^4\ _\infty$	1.27174	1.40859	1.45372

Table III gives the cumulative effect of the analysis filters. It gives the L^∞ norms for each stage of the composition of the \mathbf{A}^j , beginning with \mathbf{A}^4 , down to \mathbf{A}^0 .

The L^∞ norm for the synthesis filters \mathbf{P}^j arising from piecewise linear subdivision is simply 1 for every level, and is therefore not shown in a table.

Note that these results on stability are only numerical. Necessary and sufficient conditions guaranteeing stability in the irregular case are very difficult to establish. Schröder and Sweldens [1995] suggest constraining the resulting wavelets to have a zero order moment (i.e., the integral of the wavelet is zero). In the regular univariate case, this condition is known to be necessary for stability [Daubechies 1992]. However, it is unknown whether this is a necessary condition for stability in the irregular case.

If a zero order moment is necessary for stability, enforcing such a constraint is easy: During the partial orthogonalization process, one simply adds a linear constraint on the unknown coefficients $\alpha_{i,i'}^j$ of the wavelet that forces the wavelet to have a vanishing integral. For more information, Dahmen [1994] gives a much deeper treatment of the stability of multiresolution schemes over irregular grids, and derives some fundamental approximation-theoretic results.

7. WAVELET COMPRESSION OF SURFACES

Multiresolution analysis is widely used for data compression applications. Mallat [1989] and DeVore et al. [1992], among others, use wavelet techniques for efficient image compression. Finkelstein and Salesin [1994] develop a wavelet-based method for approximating B-spline curves within an L^∞ tolerance. Other examples of wavelet-based compression techniques abound [Berman et al. 1994; Chui and Shi 1992; Lucier 1992; Mallat and Hwang 1992].

Using wavelet techniques, lossy compression of a function is typically implemented with a three-stage algorithm:

- (1) *Filter bank decomposition.* The original function, represented by the sequence v^j , is decomposed into a coarse-level approximation v^0 together with wavelet coefficients w^0, \dots, w^{j-1} at the levels from 0 to $j - 1$.

Table III. L^∞ Norms for Successive Products of the \mathbf{A}^j , Beginning at \mathbf{A}^4

Matrix product	1-disc	2-disc	3-disc
\mathbf{A}^4	1.27174	1.40859	1.45372
$\mathbf{A}^3 * \mathbf{A}^4$	1.61574	1.95076	2.04751
$\mathbf{A}^2 * \mathbf{A}^3 * \mathbf{A}^4$	2.02336	2.50179	2.60807
$\mathbf{A}^1 * \mathbf{A}^2 * \mathbf{A}^3 * \mathbf{A}^4$	2.35466	2.78397	2.86811
$\mathbf{A}^0 * \mathbf{A}^1 * \mathbf{A}^2 * \mathbf{A}^3 * \mathbf{A}^4$	2.22687	2.63381	2.75473

- (2) *Selection.* A subset d of the detail is chosen from each sequence w^0, \dots, w^{j-1} , according to some measure of its importance.
- (3) *Reconstruction.* The selected detail d , in the form of scaled wavelets at various levels, is added back to the coarse-level approximation.

Throughout the rest of this section, we will discuss the application of subdivision wavelets to the compression of complex surface data. The common special case of polyhedral decomposition is treated in Section 7.1. Next, Section 7.2 discusses various rules for selecting wavelet coefficients. Section 7.4 discusses the preprocessing step sometimes necessary for converting general input into a form appropriate for compression. Finally, Section 7.5 illustrates the results of compression for two complex polyhedral data sets.

In this section, we exclusively consider the approximation of functions defined over triangular meshes. Quadrilateral methods require separate algorithms which are nevertheless similar in flavor.

7.1 Polyhedral Implementation

Many applications of subdivision wavelets involve the special case of piecewise linear subdivision—an example is polyhedral compression. The techniques outlined in Section 4 apply to polyhedra, but it is possible to exploit the simplicity of piecewise linear subdivision to derive an even more efficient implementation. Moreover, decomposition and reconstruction may be achieved in linear time for polyhedral surfaces without the need for a sparse matrix representation. (However, it is still necessary to solve a small linear system for the local neighborhood of wavelet coefficients.)

The chief property of piecewise linear subdivision that leads to simpler algorithms is tight locality. Unlike more complex subdivision rules, the support of a hat function built around a vertex \mathbf{v} does not extend beyond \mathbf{v} 's neighboring vertices. In this section, we show that this leads to a simpler technique for generating the filters \mathbf{A} and \mathbf{B} .

As is discussed in Section 6.3, the wavelet coefficients \mathbf{W}^j may be found through the matrix product $\mathbf{B}^j \mathbf{V}^{j+1}$. Construction of \mathbf{B}^j is greatly simplified for polyhedral subdivision, where the wavelet coefficient \mathbf{w} associated with a vertex \mathbf{v} (where \mathbf{v} is fine-level vertex added on the edge pq) may be

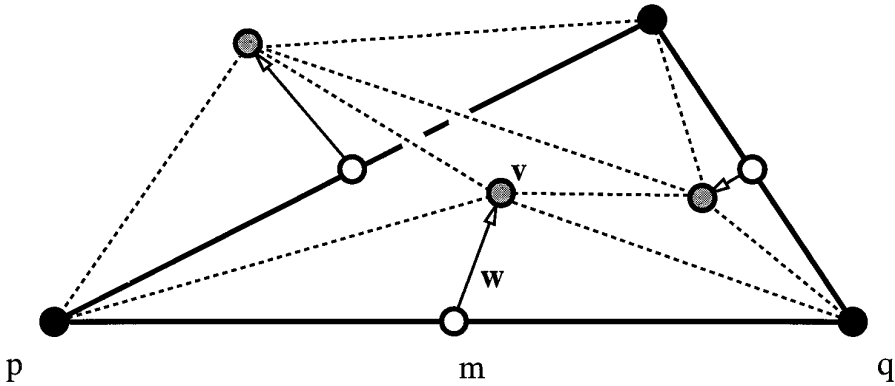


Fig. 8. Determining the wavelet coefficient \mathbf{w} around vertex \mathbf{v} , with parents p and q . The midpoint between p and q is m , and the resulting wavelet coefficient is the difference between m and \mathbf{v} .

derived by the simple rule:

$$\mathbf{w} = \mathbf{v} - \frac{1}{2}(p + q). \quad (26)$$

This equation is shown geometrically in Figure 8. It leads to a very simple algorithm for computing the wavelet coefficients—without the need for implementing sparse matrix multiplication.

Determining all level $j - 1$ wavelet coefficients w^{j-1} using Equation 26 is the first half of the level j to level $j - 1$ decomposition process. It is still necessary to derive the filter \mathbf{A}^{j-1} that gives the scaling functions for the approximation at level $j - 1$. This is not as easy, but one may still take advantage of the simplicity of the piecewise linear representation in order to achieve a procedure that is more efficient than the general case.

Once the wavelet coefficients w_i^{j-1} are derived, the coarser-level approximation at level $j - 1$ may be determined by subtracting from the level j function the effect of every scaled wavelet term $w_i^{j-1} \psi_i^{j-1}$. The problem then reduces to determining the wavelets ψ_i^{j-1} .

The ψ_i^{j-1} may be computed using the techniques of Section 4, but specifically adapted for linear subdivision. In particular, the inner product mask for the linear case is shown in Figure 4. This mask is especially easy to determine for polyhedra, because the central value is simply the valence of the vertex.

7.2 Coefficient Selection

Once the surface has been decomposed via the filter bank, the next step in surface compression is to select appropriate coefficients to add back to the base mesh.

Before examining the wavelet coefficients for selection, it is useful to first *normalize* them. Normalization allows coefficients at differing subdivision

levels to be equitably compared, despite the different domain sizes over which they are defined.

Least-squares (L^2) normalization of the wavelet coefficients assigns a weight to a wavelet coefficient that indicates how much least-squares error occurs when the coefficient is left out of the reconstruction. A wavelet ψ_i^j can be normalized to a “unit length” wavelet $\widehat{\psi}_i^j$ with the following equation:

$$\widehat{\psi}_i^j = \frac{\psi_i^j}{\sqrt{\langle \psi_i^j, \psi_i^j \rangle}}.$$

When the wavelet ψ_i^j has an associated wavelet coefficient w_i^j , the L^2 normalized coefficient \widehat{w}_i^j is therefore assigned the value

$$\widehat{w}_i^j = w_i^j \sqrt{\langle \psi_i^j, \psi_i^j \rangle}.$$

All examples presented in this article use selection based upon this L^2 normalization of the wavelet coefficients.

There are several possible techniques for selecting coefficients. These include:

- Threshold testing.* Perhaps the simplest selection technique is to simply choose all wavelet coefficients whose magnitude is greater than some coefficient ϵ [Donoho 1994]. Although thresholding is quite simple, the results are of remarkable quality, as we will see in Section 7.5. In the reconstruction, areas of high curvature are sampled more densely than relatively flat regions.
- L^2 *progressive refinement.* In certain applications, it is useful to ensure that the most important information is reconstructed first. When the L^2 normalization described above is used, it can be shown that simply supplying the c largest coefficients in decreasing order is sufficient to produce a sequence of approximations, each of which is the best possible least-squares approximation using only c coefficients [Donoho 1994]. The wavelet coefficients must first be sorted by their magnitude, which implies an $O(n \log n)$ run time in the input size.
- Maximum error: L^∞ reconstruction.* An approximation constructed according to the L^2 norm may still contain an arbitrarily large error in a sufficiently small region. An alternative approach uses the L^∞ norm to guarantee that no part of the reconstruction is farther than a user-defined tolerance from its corresponding point on the original input. L^∞ reconstruction begins with the complete, unreduced reconstruction, and attempts to remove successive wavelet coefficients until no more are possible. L^∞ wavelet compression of complex polyhedral surfaces is further studied in Eck et al. [1995].
- Location.* In some applications, it may be useful to compress only a portion of a model while the rest of it is outside the user’s view. For example, a user may be interested in viewing a specific location on a map

of the Earth, but uninterested in viewing geography elsewhere. In this case, the location information associated with a wavelet makes it possible to apply wavelet compression to the region of interest, but to avoid processing irrelevant regions of the model.

Selection by location is usually applied in conjunction with another selection method. Only those coefficients in the relevant region are candidates for the secondary selection technique.

7.3 Reconstruction Algorithms

To review, the result of decomposition is the base mesh and the coefficients of the wavelets at various levels of subdivision. The information at the vertices of the base mesh stores values of a very few scaling functions, providing a very coarse least-squares approximation of the input function. The wavelets are functions that represent the missing detail at each finer-level vertex. In order to build a wavelet approximation of the original input, we can add back to the base mesh a selected subset of these finer-level functions.

Reconstruction of the approximation is done by adding back to the base surface the scaled wavelet associated with each selected coefficient. Details of the reconstruction phase are given explicitly in Lounsbery [1994].

The effect of reconstruction is to produce vertices on the approximation surface that represent the addition of the selected wavelets to the base surface. However, this form is not appropriate for many applications that require an actual surface representation. If the approximation is to be rendered as a polyhedral surface, it is necessary to generate polygons over the approximated surface. A separate top-down recursive triangulation algorithm is required in order to connect the points on the reconstruction into polygonal form.

When c is the number of selected coefficients and j is the subdivision depth of the input function, the asymptotic time for reconstruction and triangulation is at worst $O(jc)$, but in practice usually much closer to $O(c)$. These values are borne out by the empirical evidence of run times for reconstruction.

7.4 Subdivision Connectivity

The compression techniques detailed in this section are useful for applications requiring decomposition of a function in $V^j(M^0)$, where M^0 may be a mesh of any topological type. An implicit assumption is that the connectivity of the input mesh must have the form of a mesh M^j that results from subdividing a simple mesh M^0 j times. We call this property *subdivision connectivity*.

For many applications, such as global illumination or surface editing, producing input with subdivision connectivity is a simpler matter. For other purposes, including multiresolution compression of an arbitrary mesh, an initial preprocessing step is required to convert the input into an approximation with the necessary connectivity. Such a conversion algo-

rithm is presented in Eck et al. [1995] for the general case. The examples presented in this article were converted with an algorithm written for the special case of converting input taken in cylindrical sections into points on a subdivided octahedron.

7.5 Polyhedral Examples

In this section, subdivision wavelets are applied to two compression problems: the compression of a polyhedral model consisting of over 32,000 triangles, and compression of a piecewise linear representation of a color function defined on over one million points on the globe. The data for these examples were resampled from cylindrical sections onto a subdivided octahedron.

7.5.1 Geometric Data. The input for the first example (shown in Figure 9(a)) is a polyhedral mesh consisting of 32,768 triangles whose vertices were resampled from laser range data originally provided through the courtesy of Cyberware, Inc. The triangulation was created by recursively subdividing an octahedron six times. The octahedron therefore serves as the domain mesh M^0 , with the input triangulation considered as a parametric function $S(\mathbf{x})$, $\mathbf{x} \in M^0$ lying in $V^6(M^0)$. More precisely, if v_i^6 denotes the vertices of the input mesh, $S(\mathbf{x})$ can be written as

$$S(\mathbf{x}) = \Phi^6(\mathbf{x})\mathbf{V}^6 \quad \mathbf{x} \in M^0$$

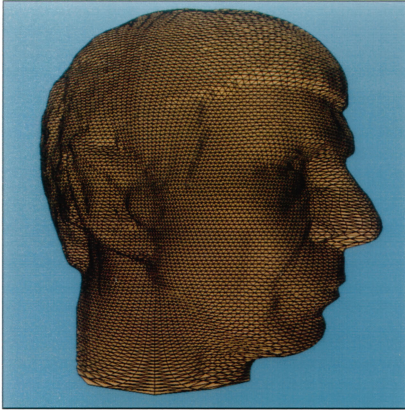
where the scaling functions $\Phi^6(\mathbf{x})$ are the (piecewise linear) functions defined through polyhedral subdivision.

The locally supported wavelet approximations $\psi_i^j(\mathbf{x})$ for this example are chosen to be supported on 2-discs. The filter bank process outlined in Section 6.3 can be applied in linear time to rewrite $S(\mathbf{x})$ in the form

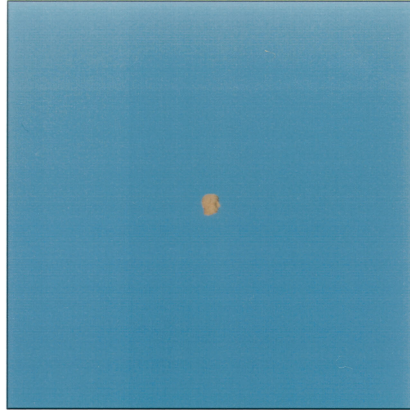
$$S(\mathbf{x}) = \Phi^0(\mathbf{x})\mathbf{V}^0 + \sum_{j=0}^5 \Psi^j(\mathbf{x})\mathbf{W}^j.$$

The first term describes a base shape as the projection of $S(\mathbf{x})$ into $V^0(M^0)$, which in this case is an approximating octahedron with vertex positions given by the six rows of \mathbf{V}^0 . For this data, the decomposition stage of the filter bank runs in about 14 seconds on an SGI Indigo² Extreme.

Approximations to the original mesh $S(\mathbf{x})$ can be easily obtained from the wavelet expansion using coefficients selected according to the threshold rule (see Section 7.2). The models in Figure 10(b), (d), and (f) are compressed to 1%, 13%, and 32%, respectively. Notice that thresholding causes the mesh to refine in areas of high detail, while leaving large triangles in areas of relatively low detail. An implementation of the entire decomposition and reconstruction process that produces Figure 10(d) runs in about 53 seconds from input to output.



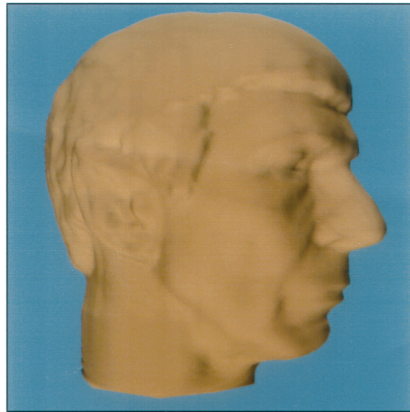
(a) The original Spock mesh.



(b) Distant view.

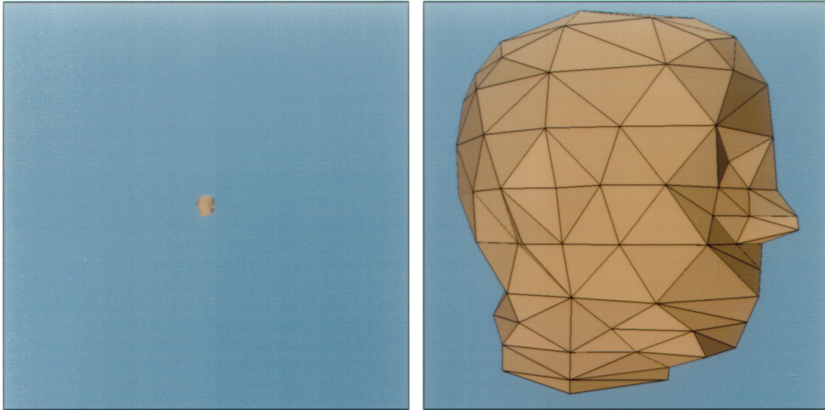


(c) Mid-range view.



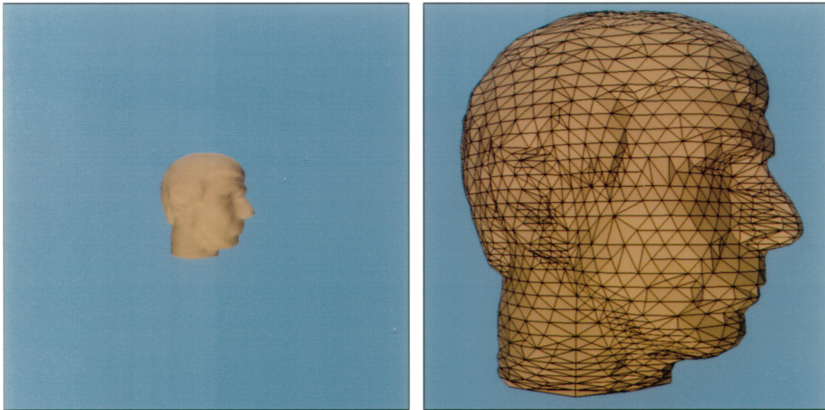
(d) Close-up view.

Fig. 9. The full-resolution Spock polyhedron. Left: the full mesh (16,386 points, 32,768 triangles). Right: Gouraud-shaded views of the full mesh at various distances.



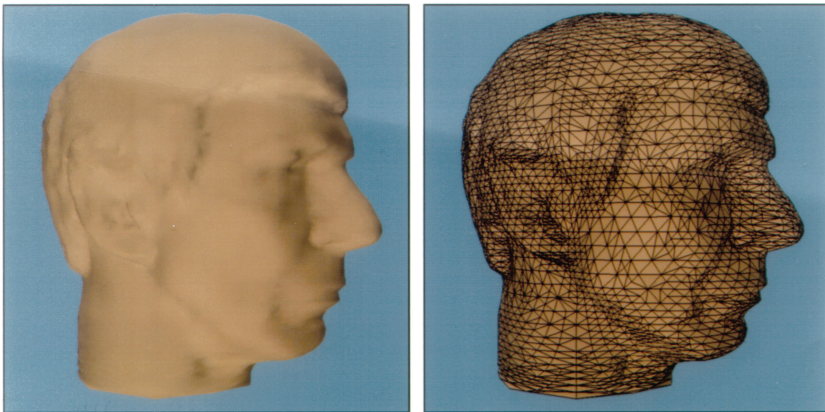
(a) Distant view of mesh in (b).

(b) 105 wavelets, 240 triangles.



(c) Mid-range view of mesh in (d).

(d) 1,966 wavelets, 4,272 triangles.



(e) Close-up view of mesh in (f).

(f) 5,054 wavelets, 10,704 triangles.

Fig. 10. Wavelet approximations of the Spock polyhedron. Left: Gouraud-shaded wavelet approximations. Right: Flat-shaded close-ups showing structure of approximations.

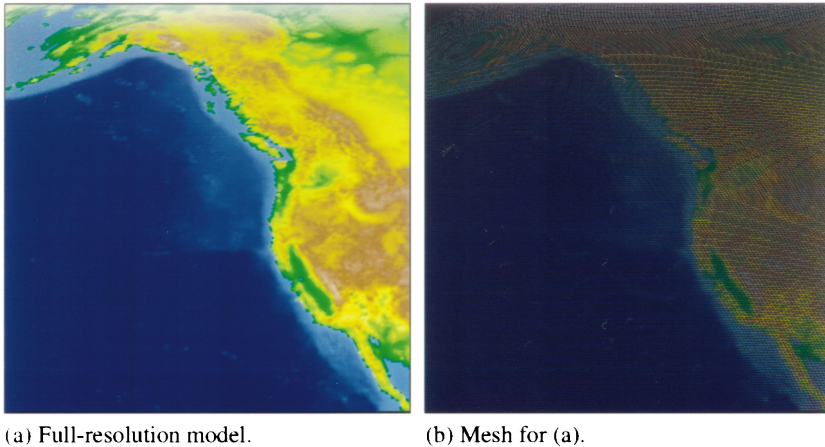


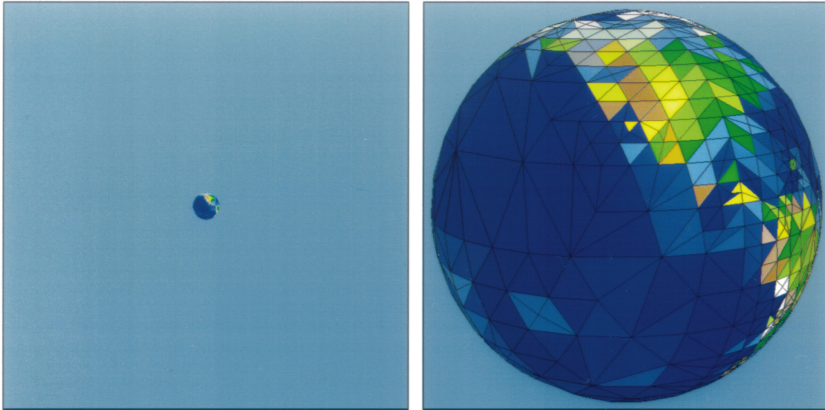
Fig. 11. Close-up of Earth data before compression.

Figure 10 also illustrates the use of wavelet approximations for automatic level-of-detail control in rendering. The original Spock polyhedron is shown in full resolution in Figure 9(a). Views of the full-resolution model from various distances are shown in the rest of Figure 9. When viewing the input polyhedron at these distances, it is inefficient and unnecessary to render all 32,000 triangles. The approximations shown in the left column of Figure 10 may instead be used without significantly degrading image quality.

Suddenly switching between models of different detail in an animation often produces a noticeable jump. This problem is easily mended by using a wavelet expansion where the wavelet coefficients are treated as continuous functions of the viewing distance. This simple technique allows the object geometry to smoothly change its appearance as the viewing distance changes. This method has proved successful shown in a frame-by-frame animation we have produced.

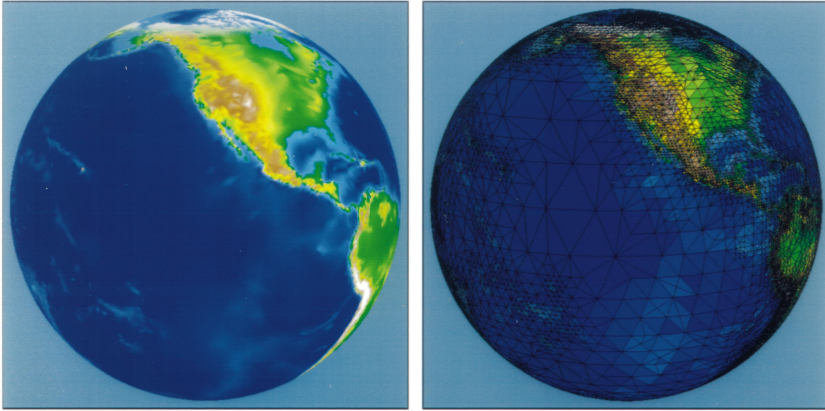
7.5.2 Color Data. Subdivision wavelets may be applied to more general functions over surfaces than geometric data. Figure 12 demonstrates another application—that of compressing a function on the sphere. In this example, elevation and bathymetry data obtained from the U.S. National Geophysical Data Center was used to create a piecewise linear pseudocoloring of the sphere. The resulting color function was represented by 2,097,152 triangles and 1,048,578 vertices. The full-resolution pseudocoloring was too large to be rendered on an SGI Indigo² Extreme with 128 MB, and is therefore not shown in its entirety in Figure 12. Instead, Figure 11 shows a close-up of a region that is compressed in Figure 12. An appreciation for the density of the data can be obtained from Figure 11(b), where even at close range the mesh lines of the original uncompressed data are so dense that the image appears almost completely black.

The approximations shown in Figure 12(a)–(f) were produced using subdivision wavelet compression. Figure 12(a) shows a distant view of the



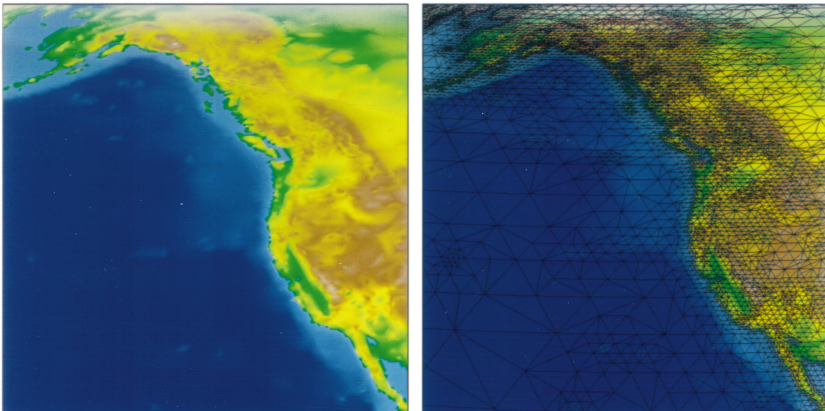
(a) Distant view of (b).

(b) Compression to 0.1%.



(c) Mid-range view of (d).

(d) Compression to 2%.



(e) Close-up view of (f).

(f) Compression to 16%.

Fig. 12. Approximating color as a function over the sphere.

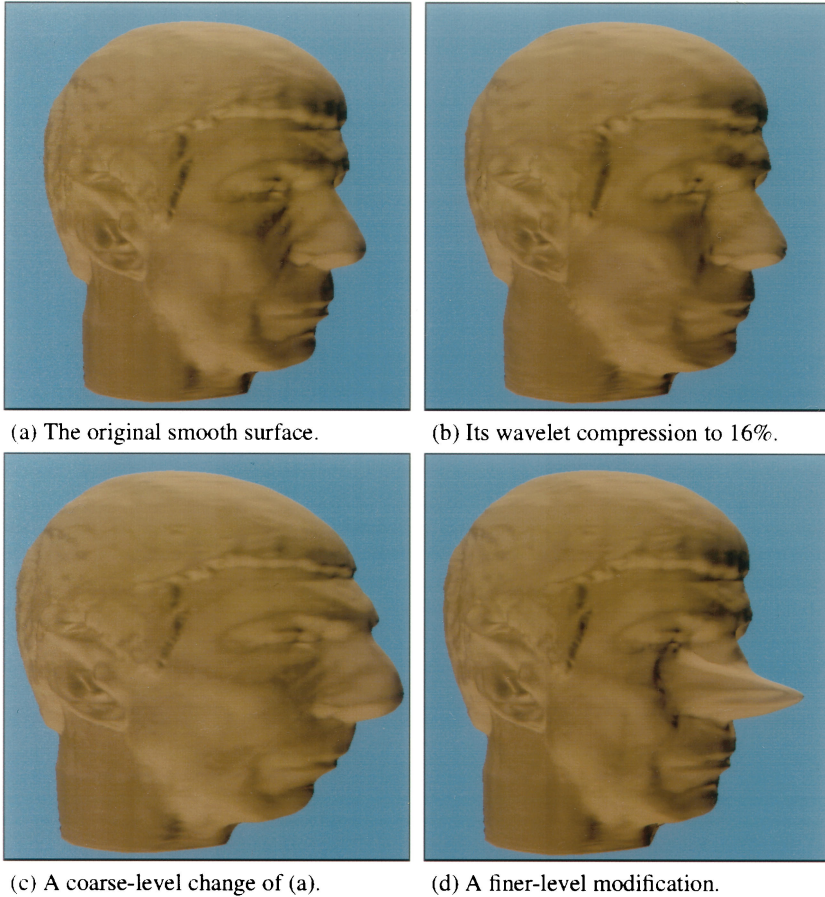


Fig. 13. Compression and multiresolution editing of a smooth surface.

Earth using an approximation of only 0.1% (the mesh is shown in Figure 12(b)). Likewise, Figures 12(c) and (d) show the result of compression to 2% for a medium-range view. At close range the compression to 16% in (e) is nearly indistinguishable from the full-resolution model in Figure 11(a). A comparison of the meshes shown in Figure 11(b) and Figure 12(f) reveals the striking degree of compression achieved in this case.

8. SMOOTH SURFACE MODELING

This section describes how subdivision wavelets can be used to compress and edit smooth subdivision surfaces. Although our examples concentrate on the modeling of smooth surfaces based on the Dyn et al. [1990] butterfly scheme, the techniques described in this section may be applied to general subdivision surfaces, at the cost of potentially quadratic processing time in the case of Loop's method.

In Section 7, we saw examples of polyhedral compression using wavelets. In this section, we examine compression of surfaces defined by smooth subdivision rules.

8.1 Smooth Surface Compression

The result of smooth surface compression is shown in Figure 13. The surface in Figure 13(a) shows the level 6 Spock data set from Section 7.5 after it has been subdivided using the tangent-plane smooth butterfly subdivision rule. Figure 13(b) shows the result of compression using wavelets that were developed from the butterfly scheme. The surface of Figure 13(a) has been compressed to a representation (depicted in Figure 13(b)) that may be stored using only 16% of the original coefficients.

8.2 Multiresolution Editing

Subdivision wavelets may also be used for editing shapes at multiple resolutions. Although we have not fully implemented multiresolution editing, Figures 13(c)–(d) show an example of editing the smooth shape seen in Figure 13(a).

Figure 13(c) shows the effect of changing a single scaling function coefficient on the level 0 base octahedron. Because finer-level vertices in the same region are defined relative to the coarser shape, they move along with the modification. However, the geometry in areas away from the front of the bust is not affected.

It is also possible to locally modify the shape at a finer level by changing the value of a wavelet coefficient at that level. The result of modifying a single level 3 wavelet coefficient is shown in Figure 13(d).

The changes shown in Figure 13(c)–(d) were created by simply modifying a single value in the wavelet representation. It is preferable to make these changes under a more powerful user interface. Finkelstein and Salesin [1994] discuss interfaces that are more appropriate for wavelet editing of curves and surfaces.

9. SUMMARY AND FUTURE WORK

In this article, we have established a theoretical basis for applying multiresolution analysis to surfaces of arbitrary topological type. These results hold for any local, uniformly convergent, continuous, primal subdivision scheme, including polyhedral subdivision, the butterfly scheme [Dyn et al. 1990], Loop's scheme [1987], and Catmull-Clark surfaces [1978]. The results also hold for piecewise smooth subdivision as described in Hoppe [1994] and Hoppe et al. [1994], and for open surfaces possessing boundaries.

There are numerous areas for future work:

—*Linear time sparse matrix inversion.* As explained in Section 6.3, linear-time reconstruction for general subdivision rules depends upon linear-time solution of general sparse linear systems. Hence, finding an $O(n)$

sparse linear system solver is of value for efficiently implementing subdivision wavelets for more general subdivision schemes, such as the G^1 methods of Loop and Catmull-Clark.

- Semiorthogonal wavelets.* The wavelets developed in Section 6 are biorthogonal and linear-time for interpolating subdivision. It would be interesting to determine if linear-time semiorthogonal wavelets also exist.
- Simplifying the topological type.* The surface decomposition developed herein retains the topological type of the input surface. When the input is a relatively simple object with many small holes, it is more often desirable to decompose the input into a “topologically simpler” surface; that is, one with lower genus or fewer boundary curves.

ACKNOWLEDGMENTS

We thank Tom Duchamp, Jean Schweitzer, Peter Schröder, and Wim Sweldens for many productive discussions, and Eric Stollnitz for help with figures.

REFERENCES

- ANDERSSON, L., HALL, N., JAWERTH, B., AND PETERS, G. 1993. Wavelets on closed subsets of the real line. In *Recent Advances in Wavelet Analysis*, L. L. Schumaker and G. Webb, Eds. Academic Press, New York, 1–61.
- BARTLE, R. G. 1964. *The Elements of Real Analysis*. Wiley, New York.
- BERMAN, D., BARTELL, J., AND SALESIN, D. 1994. Multiresolution painting and compositing. In *SIGGRAPH 94 Conference Proceedings*, ACM SIGGRAPH, Addison Wesley, Reading, MA, 85–90.
- BEYLKIN, G., COIFMAN, R., AND ROKHLIN, V. 1991. Fast wavelet transforms and numerical algorithms. *Commun. Pure Appl. Math.* 44, 141–183.
- CARNICER, J. M., DAHMEN, W., AND PEÑA, J. M. 1994. Local decompositions of refinable spaces. Tech. Rep. Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Aachen, Germany.
- CATMULL, E. AND CLARK, J. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Comput. Aided Des.* 10, 6, 350–355.
- CERTAIN, A., POPOVIC, J., DE ROSE, T., DUCHAMP, T., SALESIN, D., AND STUETZLE, W. 1996. Interactive multiresolution surface viewing. In *SIGGRAPH 96 Conference Proceedings*, ACM SIGGRAPH, Addison Wesley, Reading, MA, 91–98.
- CHUI, C. AND QUAK, E. 1992. Wavelets on a bounded interval. In *Numerical Methods of Approximation Theory*, D. Braess and L. L. Schumaker, Eds. Birkhäuser-Verlag, Basel, 1–24.
- CHUI, C. 1992. *Wavelet Analysis and its Applications*, Vol. 1. Academic Press, Boston, MA.
- CHUI, C. K. AND SHI, X. 1992. Wavelets and multiscale interpolation. In *Mathematical Methods in Computer Aided Geometric Design II*, Tom Lyche and Larry L. Schumaker, Eds. Academic Press.
- COHEN, A., DAUBECHIES, I., AND VIAL, P. 1993. Multiresolution analysis, wavelets and fast algorithms on an interval. *Appl. Comput. Harmon. Anal.* 1, 1, 100–115.
- COHEN, A., DAUBECHIES, I., AND FEAUVEAU, J.-C. 1992. Biorthogonal bases of compactly supported wavelets. *Commun. Pure Appl. Math.* XLV, 485–560.
- DAHMEN, W. 1994. Stability of multiscale transformations. Tech. Rep. Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Aachen, Germany.

- DAHMEN, W., KLEEMAN, B., PRÖSSDORF, S., AND SCHNEIDER, R. 1993. A multiscale method for the double layer potential equation on a polyhedron. Tech. Rep. 91, Institut für Geometrie und Praktische Mathematik, Rheinisch-Westfälische Technische Hochschule Aachen, Nov.
- DAHMEN, W. AND MICCHELLI, C. A. 1993. Using the refinement equation for evaluating integrals of wavelets. *SIAM J. Numer. Anal.* 30, 2 (April) 507–537.
- DAUBECHIES, I. 1988. Orthonormal bases of compactly supported wavelets. *Comm. Pure Appl. Math.* 41, 909–996.
- DAUBECHIES, I. 1992. *Ten Lectures on Wavelets*. SIAM, Philadelphia.
- DEVORE, R., JAWERTH, B., AND LUCIER, B. 1992. Image compression through wavelet transform coding. *IEEE Trans. Inf. Theory* 38, 2 (March), 719–746.
- DONOHO, D. L. 1994. Unconditional bases are optimal bases for data compression and for statistical estimation. *Appl. Comput. Harmon. Anal.* 1, 1, 100–115.
- DOO, D. AND SABIN, M. 1978. Behaviour of recursive division surfaces near extraordinary points. *Computer Aided Design*, 10, 6, 356–360.
- DOO, D. W. H. 1978. A recursive subdivision algorithm for fitting quadratic surfaces to irregular polyhedrons. PhD Thesis, Brunel Univ.
- DYN, N., LEVIN, D., AND GREGORY, J. 1990. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans. Graph.* 9, 2, (April) 160–169.
- ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. 1995. Multiresolution analysis of arbitrary meshes. In *SIGGRAPH 95 Conference Proceedings*, ACM SIGGRAPH, Addison Wesley, Reading, MA, 173–182.
- FINKELSTEIN, A. AND SALESIN, D. 1994. Multiresolution curves. In *SIGGRAPH 94 Conference Proceedings*, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 261–268.
- FORSEY, D. AND BARTELS, R. 1988. Hierarchical B-spline refinement. *Comput. Graph.* 22, 4, 205–212.
- HALSTEAD, M., KASS, M., AND DEROSE, T. 1993. Efficient, fair interpolation using Catmull-Clark surfaces. *Computer Graphics Annual Conference Series*, (August) 35–44.
- HOPPE, H. 1994. Surface reconstruction from unorganized points. PhD thesis, TR 94-06-01. Univ. of Washington, Seattle, Washington, June.
- HOPPE, H., DEROSE, T., DUCHAMP, T., HALSTEAD, M., JIN, H., McDONALD, J., SCHWEITZER, J., AND STUETZLE, W. 1994. Piecewise smooth surface reconstruction. *Computer Graphics Annual Conference Series*, (July) 295–302.
- HOPPE, H., DEROSE, T., DUCHAMP, T., McDONALD, J., AND STUETZLE, W. 1993. Mesh optimization. *Computer Graphics Annual Conference Series*, (August) 19–26.
- JIA, R.-Q. AND MICCHELLI, C. A. 1991. Using the refinement equations for the construction of pre-wavelets II: Powers of two. In *Curves and Surfaces*, P. J. Laurent, A. LeMéhauté, and L. L. Schumaker, Eds. Academic Press, 209–246.
- LIU, Z., GORTLER, S. J., AND COHEN, M. E. 1994. Hierarchical spacetime control. *Computer Graphics Annual Conference Series*, (July) 35–22.
- LOOP, C. T. 1987. Smooth subdivision surfaces based on triangles. Master's thesis, Dept. of Mathematics, Univ. of Utah, (August).
- LOUNSBERY, J. M. 1994. Multiresolution analysis for surfaces of arbitrary topological type. PhD thesis, Univ. of Washington, Seattle, WA, Sept.
- LUCIER, B. J. 1992. Wavelets and image compression. In T. Lyche and L. L. Schumaker, Eds., *Mathematical Methods in Computer Aided Geometric Design II*. Academic Press.
- MALLAT, S. 1989. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Patt. Anal. Mach. Intell.* 11, 7 (July) 674–693.
- MALLAT, S. AND HWANG, W. L. 1992. Singularity detection and processing with wavelets. *IEEE Trans. Inf. Theory* 38, 2 (March) 617–643.
- MEYER, Y. 1992. Ondelettes sur l'intervalle. *Rev. Mat. Iberoamericana* 7, 115–133.
- MEYER, Y. 1993. *Wavelets and Operators*. Cambridge University Press.
- MEYERS, D. 1994a. Multiresolution tiling. In *Proceedings of Graphics Interface*, (May) 25–32.
- MEYERS, D. 1994b. Reconstruction of surfaces from planar contours. PhD thesis, Univ. of Washington, Seattle, July.

- PENTLAND, A. P. 1992. Fast solutions to physical equilibrium and interpolation problems. *Visual Comput.* 8, 5–6 (June) 303–314.
- SCHRÖDER, P. AND SWELDENS, W. 1995. Spherical wavelets: Efficiently representing functions on the sphere. In *SIGGRAPH 95 Conference Proceedings*, ACM SIGGRAPH, Addison Wesley, Reading, MA, 161–172.
- SCHROEDER, W. J., ZARGE, J. A., AND LORENSON, W. E. 1992. Decimation of triangle meshes. *Comput. Graph.* 26, 2, 65–70.
- SWELDENS, W. 1994. Construction and applications of wavelets in numerical analysis. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium.
- TURK, G. 1992. Re-tiling polygonal surfaces. *Comput. Graph.* 26, 2, 55–64.