Bicubic Subdivision-Surface Wavelets for Large-Scale Isosurface Representation and Visualization

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

Abstract

We introduce a new subdivision-surface wavelet transform for arbitrary two-manifolds with boundary that is the first to use simple lifting-style filtering operations with bicubic precision. We also describe a conversion process for re-mapping large-scale isosurfaces to have subdivision connectivity and fair parameterizations so that the new wavelet transform can be used for compression and visualization. The main idea enabling our wavelet transform is the circular symmetrization of the filters in irregular neighborhoods, which replaces the traditional separation of filters into two 1-D passes. Our wavelet transform uses polygonal base meshes to represent surface topology, from which a Catmull-Clark-style subdivision hierarchy is generated. The details between these levels of resolution are quickly computed and compactly stored as wavelet coefficients. The isosurface conversion process begins with a contour triangulation computed using conventional techniques, which we subsequently simplify with a variant edge-collapse procedure, followed by an edge-removal process. This provides a coarse initial base mesh, which is subsequently refined, relaxed and attracted in phases to converge to the contour. The conversion is designed to produce smooth, untangled and minimally- skewed parameterizations, which improves the subsequent compression after applying the transform. We have demonstrated our conversion and transform for an isosurface obtained from a high-resolution turbulent-mixing hydrodynamics simulation, showing the potential for compression and level-of-detail visualization.

Keywords: Compression Algorithms, Geometric Modeling, Isosurfaces, Multiresolution Methods, Wavelets

1 Introduction

Modeling complex geometries, like isosurfaces resulting from high-resolution turbulent-mixing hydrodynamics simulated on massively parallel supercomputers, is a challenging problem in scientific visualization. Volumetric data sets are often too large for efficient exploration of isosurfaces, material boundaries, or shock waves. Interactive exploration of families of surfaces, such as time-dependent surfaces or isosurfaces at multiple isovalues, is facilitated by compact surface representations that allow fast and selective access. When storing and transmitting large amounts of data, compression of the geometry and associated fields becomes an important application.

Multiresolution representations, like wavelets, are essential to process large-scale geometries efficiently. They allow the display of a complete surface model at a low resolution, as well as zoomed views at high resolution, using small amounts of computation and transmission time. If a geometric model can be evaluated locally and progressively, it is possible to satisfy demand-driven queries, *i.e.*, providing a maximal amount of detail within a prescribed processing time or providing a minimal amount of detail satisfying a prescribed error bound. Progressive evaluation adaptively increases the resolution of a geometric model while the data is accessed, without processing the same information twice. The need of multiresolution modeling for view-dependent visualization was motivated by Heckbert/Garland [18].

Applications for multiresolution surface modeling include:

- *Computational Physics.* Surfaces in large-scale physics computations include isocontours, shock fronts, and material boundaries derived from volume fractions [3]. Parametric surface representations are ideal for interactive exploration, as they offer efficient access to the geometry and correspond well with the capabilities of current rendering hardware.
- *Medical imaging.* Surface approximations representing bone or vessel structures are obtained from contouring algorithms applied to volumetric data constructed from computer tomography (CT) scans. Smooth surface models can be used for visualizing these structures, identifying diseases and planning surgery.
- *Reverse engineering.* Points sampled from a real model–using, for example, laser range scanners–can be approximated or interpolated by continuous surface models. These models facilitate the design of geometry and subsequent numerical analysis and manufacturing.

In the present paper, we construct a new wavelet transform based on uniform bicubic B-Spline subdivision and fitting rules that are generalized to arbitrary polygonal base meshes representing two-manifold topologies of arbitrary genus. Compared to the previous smooth subdivision-surface wavelet constructions [28, 29], our approach requires only fast and local lifting-style filtering operations rather than global sparse matrix solutions, and thus makes large-data surface compression feasible. Our surface representation satisfies C^2-continuity conditions at any level of resolution, except at extraordinary points (control points with other than four incident edges), where the surface is tangent-plane continuous. Our wavelet construction also includes modifications to support boundary curves and sharp features. Our wavelet transform is structurally similar to Catmull-Clark subdivision [5], has comparable simplicity, and also produces piecewise-bicubic patches.

Besides the wavelet construction, the contribution of this paper is a fitting algorithm that converts isosurfaces efficiently to our surface representation. The fitting algorithm is composed of the following steps:

- **Isosurface extraction:** Using common algorithms, unstructured, triangulated meshes of high resolution are obtained.
- **Topology-preserving mesh simplification:** Coarse meshes are obtained using an edge-collapsing/removing algorithm. These are then used as base meshes providing the domain for our parametric surface representation.

[1]Center for Applied Scientific Computing (CASC), Lawrence Livermore National Laboratory, P.O. Box 808, L-561, Livermore, CA 94551, USA
{mbertram,duchaineau1}@llnl.gov

[2]Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California at Davis, Davis, CA 95616-8562, USA
{bertram,hamann,joy}@cs.ucdavis.edu

- **Regular mesh refinement and shrink wrapping:** The fully detailed isosurface is obtained by an iterative fitting process that provides a smooth, high quality parameterization in the correct form for the wavelet transform.

- **Wavelet decomposition:** our new transform is applied to the shrink-wrapped surface to provide multiple resolutions of surface approximation. The differences between resolution levels are compactly stored so that progressive reconstruction from coarse to fine resolutions is possible.

Since resolution capabilities of scanners as well as performance of supercomputers are improving rapidly, it becomes increasingly important to develop efficient modeling tools for highly complex geometries. B-spline patches alone are no longer sufficient for representing these geometries efficiently. Hierarchical representations are required that progressively provide multiple levels of resolution and are stored in a compressed form. The level of refinement for the surface during display should depend on the magnitude of local details and on the displayed level of resolution. Our wavelet transform provides all these capabilities in a highly efficient way. To demonstrate our method, we have extracted an isosurface from an extremely high-resolution turbulent-mixing hydrodynamics simulation [30] and have visualized it at multiple resolutions.

2 Related Work

Wavelet transforms are used in signal processing and numerical mathematics to analyze functions by separating *details* corresponding to different frequency bands [6, 36]. Details of a certain frequency can efficiently be added or removed to switch between resolutions. These details are represented by wavelet coefficients that typically have small absolute values and can be compressed by arithmetic coding [31]. Wavelets are thus a useful tool the concise, progressive transmission of scientific data [38].

An important tool for wavelet construction is the *lifting scheme*, described by Sweldens [37]. A similar technique was developed earlier by Dahmen [7]. Wavelet lifting makes it possible to design wavelets with certain desirable properties: increased numbers of vanishing moments, efficient transform computation, and the ability to implement integer-arithmetic transforms for lossless compression [4].

Lounsbery [28] showed that wavelets can be constructed for subdivision surfaces on arbitrary polygonal base meshes such that they exactly reproduce certain subdivision rules when assuming zero detail. It is possible to reproduce polynomial splines with certain properties, like interpolation or variational subdivision [22]. Schröder/Sweldens [34] define a variety of subdivision-surface wavelets specialized to spherical domains. Piecewise-constant approaches were further improved by Nielson *et al.* [32] and Bonneau [2]. In contrast to regular subdivision rules, signal processing algorithms for meshes that are irregular on every level of resolution are described by Guskov *et al.* [16].

Unfortunately, only few smooth wavelet constructions are known that work efficiently on domains of arbitrary topology and represent smooth surfaces. Lounsbery *et al.* [29] construct wavelets for Loop's subdivision [27] and Catmull-Clark surfaces [5]. These wavelet constructions, however, require the solution of sparse, global systems of equations and are thus not applicable to large data sets. Our construction is based on local lifting operations that are generalized to irregular mesh domains. The number of operations for computing our wavelet transform is less than twice the number required for Catmull-Clark subdivision.

In order to apply our wavelet transform to irregular base meshes refined by regular subdivision, we need to convert surfaces to a mesh structure that is obtained by recursive subdivision. Therefore, we need to construct base meshes and smooth parameterizations mapping these meshes onto the surface geometry. For surfaces defined by scattered data (unorganized points) in three dimensions, such algorithms were described by Eck/Hoppe [12] and Guo [15]. Both algorithms build base meshes and fit smooth patches to the surface regions. Eck/Hoppe use bicubic B-spline patches minimizing a fairness function and satisfying tangent-plane continuity constraints. Guo's construction is based on C^2-continuous blending surfaces [14]. Another surface fitting scheme for dense polygonal meshes is described by Krishnamurthy/Levoy [24]. Hoppe *et al.* [19] introduce surface reconstruction and fitting with Loop subdivision surfaces, where they extend the Loop rules to include sharp features.

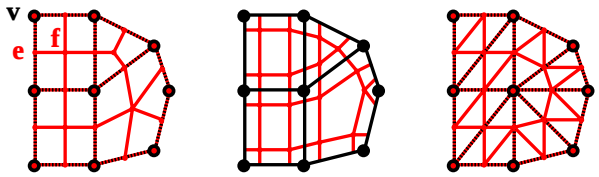


Figure 1: Subdivision schemes. From left to right: Catmull-Clark; Doo-Sabin; Butterfly/Loop.

Surfaces represented by unstructured polygonal meshes can be converted to a regularly subdivided mesh by multiresolution adaptive parameterization of surfaces (MAPS), see Lee *et al.* [25]. MAPS simplifies a given triangle mesh by recursively removing a fraction of all vertices and re-triangulating. During this procedure, the original mesh is mapped onto the coarser triangulation. The coarsest mesh is regularly refined using Loop's subdivision to smooth the parameterization and fitted to the original mesh. An algorithm that *shrink-wraps* a mesh to a given surface is described by Kobbelt *et al.* [23]. We use a similar shrink-wrapping approach for surface fitting. However, we are using quadrilateral rather than triangular subdivision and extend the approach to surfaces of arbitrary genus. To construct a base mesh, we use an edge-collapse algorithm similar to Hoppe's progressive meshes [20]. There exists a variety of mesh-simplification approaches. For an overview of different methods, we refer to Lindstrom/Turk [26].

3 Bicubic Subdivision-Surface Wavelets

In the following, we introduce an index-free notation for certain subdivision rules that we use to define our wavelet transform. We describe our lifted wavelet construction and show how to represent surface boundaries and sharp features. At the end of this section, we outline the most important properties of our wavelet construction.

3.1 Subdivision Rules

Subdivision surfaces are limit surfaces that result from recursive refinement of polygonal base meshes. A subdivision step refines a *submesh* to a *supermesh* by inserting vertices. The positions of all vertices of the supermesh are computed from the positions of the vertices in the submesh, based on certain subdivision rules. Most subdivision schemes converge rapidly to a continuous limit surface, and a mesh obtained from just a few subdivisions is often a good approximation for surface rendering. Subdivision surfaces that reproduce piecewise polynomial patches can be evaluated in a closed form at arbitrary parameter values [35].

There exists a variety of different mesh-subdivision schemes, see Figure 1. Catmull-Clark subdivision [5] generalizes bicubic B-spline subdivision to arbitrary topology. Vertices in the supermesh correspond to faces, edges, or vertices in the submesh. In the following, we denote the corresponding vertex types as $\mathbf{f}$, $\mathbf{e}$, and $\mathbf{v}$,

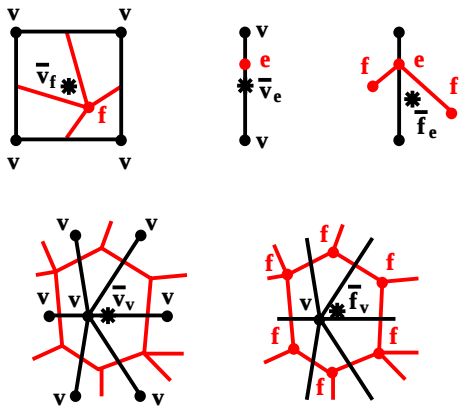


Figure 2: Examples for index-free notation. The point $\overline{\mathbf{v}}_\mathbf{f}$ is the centroid of a face; $\overline{\mathbf{v}}_\mathbf{e}$ is the midpoint of an edge; $\overline{\mathbf{f}}_\mathbf{e}$ is the midpoint of the line segment defined by $\mathbf{f}$ vertices; $\overline{\mathbf{v}}_\mathbf{v}$ is the centroid of all adjacent $\mathbf{v}$ vertices; and $\overline{\mathbf{f}}_\mathbf{v}$ is the centroid of $\mathbf{f}$ vertices.

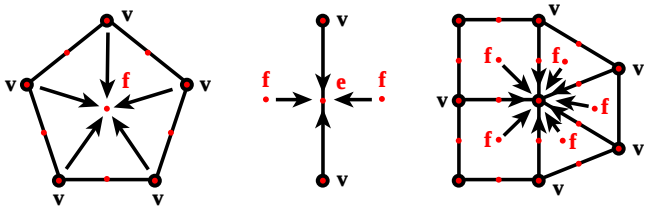


Figure 3: Catmull-Clark subdivision rules (apply left to right).

respectively. All faces produced by Catmull-Clark subdivision are quadrilaterals. An interpolating scheme based on the same mesh structure is described by Kobbelt [21]. Doo-Sabin subdivision [9] generalizes biquadratic B-splines and uses a supermesh without $\mathbf{v}$ vertices that correspond to vertices in a submesh. All vertices produced by Doo-Sabin subdivision are of the same type and have valence four. Loop [27] introduced a smooth subdivision scheme for triangle meshes. It produces $\mathbf{e}$ and $\mathbf{v}$ vertices subdividing each triangle into four. An interpolating scheme that uses the same mesh structure is known as *butterfly scheme* [11, 39].

For our wavelet construction, we use the Catmull-Clark subdivision structure with slightly different subdivision rules. To describe subdivision rules that determine new vertex positions, we introduce an index-free notation. We use the averaging operator $\overline{x}_y$, where x and y can represent $\mathbf{f}$, $\mathbf{e}$, or $\mathbf{v}$. This operator returns for every vertex of type y the arithmetic average of all adjacent vertices of type x. If there are no direct neighbors of type x, then $\overline{x}_y$ returns the average from those vertices of type x that correspond to adjacent primitives, *i.e.*, adjacent vertices or incident edges or faces. Examples for the averaging operator are shown in Figure 2.

To provide an example for our index-free notation, we formulate the Catmull-Clark subdivision rules in algorithmic notation using the $\overline{x}_y$ operator:

$$\begin{aligned} &1.\quad \mathbf{f} \leftarrow \overline{\mathbf{v}}_\mathbf{f} \\ &2.\quad \mathbf{e} \leftarrow \tfrac{1}{2}\left(\overline{\mathbf{v}}_\mathbf{e} + \overline{\mathbf{f}}_\mathbf{e}\right) \\ &3.\quad \mathbf{v} \leftarrow \tfrac{1}{n_\mathbf{v}}\left(\overline{\mathbf{f}}_\mathbf{v} + \overline{\mathbf{v}}_\mathbf{v} + (n_\mathbf{v}-2)\mathbf{v}\right) \end{aligned} \tag{3.1}$$

Here, $n_\mathbf{v}$ denotes the valence of a vertex. The three rules are illustrated in Figure 3. The $\mathbf{v}$ vertices are initially defined by the coordinates given by a submesh. The initial values for $\mathbf{e}$ and $\mathbf{f}$ vertices are irrelevant. The first rule defines each $\mathbf{f}$ vertex to be located at the centroid of its corresponding face. The second rule defines each $\mathbf{e}$ vertex to be the average of its edge midpoint $\overline{\mathbf{v}}_\mathbf{v}$ and the midpoint $\overline{\mathbf{f}}_\mathbf{e}$ of the two adjacent $\mathbf{f}$ vertices. The third rule re-defines the position for each $\mathbf{v}$ vertex as a weighted sum of its neighboring $\mathbf{f}$ vertices, its adjacent submesh vertices $\overline{\mathbf{v}}_\mathbf{v}$, and its own location. This vertex-modification step is performed simultaneously for all vertices.

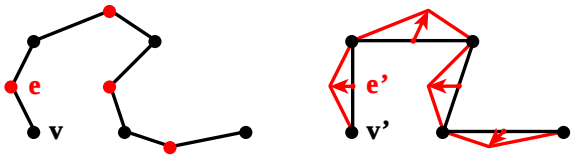


Figure 4: Wavelet decomposition coarsens a control polygon and stores difference vectors in place of removed vertices.

Subdivision rules like these define vertex modifications that are necessary to determine all supermesh coordinates for an individual subdivision step. For the next subdivision step, all vertices become $\mathbf{v}$ vertices again, and the same subdivision rules are applied recursively.

3.2 Lifted One-Dimensional Wavelets

A discrete wavelet transform (DWT) can be defined to represent hierarchical uniform B-splines under dyadic refinement [6, 10, 36]. The principle is to decompose a B-spline control polygon representing a function at a fine resolution into a hierarchy of coarser B-splines and a set of wavelet coefficients representing the details that were reduced in each coarsening step. This process is called *decomposition* or *analysis*. Figure 4 illustrates a decomposition step. Every second vertex is denoted as $\mathbf{v}$ vertex and remains in the coarser control polygon. The coordinates of a $\mathbf{v}$ vertex may be modified, the new point is denoted by $\mathbf{v}'$ (scaling-function coefficient). All other vertices correspond to edges in the coarser polygon and thus are denoted as $\mathbf{e}$ vertices. The latter are transformed into difference vectors $\mathbf{e}'$ (wavelet coefficients) defining the details necessary to reconstruct the finer control polygon.

Decomposition rules for a DWT are defined by two linear operators, a *fitting* operator $\mathbf{F}$ predicting the vertex coordinates for the coarser polygon, and a *compaction-of-difference* operator $\mathbf{C}$ representing the reduced details:

$$\begin{aligned} \mathbf{v}' &= \mathbf{F}(\mathbf{v},\mathbf{e}) \quad \text{and} \\ \mathbf{e}' &= \mathbf{C}(\mathbf{v},\mathbf{e}). \end{aligned} \tag{3.2}$$

Decomposition is recursively applied to a coarse polygon until a base resolution is reached. A DWT thus provides a base polygon and all individual levels of detail that need to be added recursively to reconstruct an original polygon. An inverse DWT is defined by *reconstruction* or *synthesis* rules that invert the decomposition rules. Starting with a base polygon, an inverse DWT applies reconstruction rules recursively in reverse order of decomposition and thus adds more and more detail to a polygon. Reconstruction rules are defined by a *subdivision* operator $\mathbf{S}$ predicting the shape of a next finer control polygon and an *expansion* operator $\mathbf{E}$ providing the missing details:

$$\begin{pmatrix}\mathbf{v}\\ \mathbf{e}\end{pmatrix} = \mathbf{S}(\mathbf{v}') + \mathbf{E}(\mathbf{e}'). \tag{3.3}$$

To obtain a smooth approximating curve, the reconstruction process can be terminated at any level of resolution providing a control polygon at an intermediate level of detail. The curve is obtained by applying the subdivision operator $\mathbf{S}$ *ad infinitum* and assuming zero detail on all finer levels. In the case of B-spline wavelets, the operator $\mathbf{S}$ reproduces in the limit a B-spline curve with uniform knot

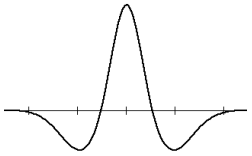


Figure 5: Lifted cubic B-spline wavelet.

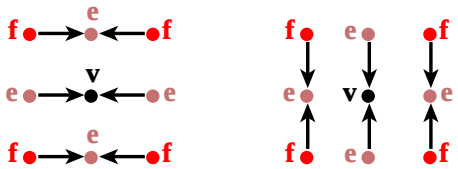


Figure 6: Tensor-product lifting operation on a rectilinear grid. A one-dimensional lifting operation is applied first to the rows and then to the columns of a grid.

vector. A B-spline curve can be computed directly from a control polygon [13].

The reconstruction rules for a cubic B-spline wavelet transform can be defined in index-free notation as follows:

$$\begin{aligned} &1.\quad \mathbf{v} \leftarrow \mathbf{v} - \tfrac{3}{4}\overline{\mathbf{e}}_{\mathbf{v}} \\ &2.\quad \mathbf{e} \leftarrow \mathbf{e} + \overline{\mathbf{v}}_{\mathbf{e}} \\ &3.\quad \mathbf{v} \leftarrow \tfrac{1}{2}\mathbf{v} + \tfrac{1}{2}\overline{\mathbf{e}}_{\mathbf{v}} \end{aligned} \tag{3.4}$$

These three vertex modifications represent *lifting* operations for the DWT. Lifting is used to define the shape of wavelets with certain properties, like vanishing moments. (We use the term lifting throughout this paper to denote local operations for computing the wavelet transform.) The subdivision operator $\mathbf{S}$ is obtained from these reconstruction rules by assuming zero wavelet coefficients $\mathbf{e}'$. The operator $\mathbf{S}$ reproduces dyadic B-spline subdivision [6, 10]. Vertices of type $\mathbf{e}'$ and $\mathbf{v}'$ represent coefficients for wavelets and B-spline *scaling functions*, respectively. A cubic B-spline wavelet obtained from our construction is depicted in Figure 5.

To construct the corresponding decomposition rules we invert the three individual lifting operations in reverse order. The decomposition rules are defined as:

$$\begin{aligned} &1.\quad \mathbf{v} \leftarrow 2\mathbf{v} - \overline{\mathbf{e}}_{\mathbf{v}} \\ &2.\quad \mathbf{e} \leftarrow \mathbf{e} - \overline{\mathbf{v}}_{\mathbf{e}} \\ &3.\quad \mathbf{v} \leftarrow \mathbf{v} + \tfrac{3}{4}\overline{\mathbf{e}}_{\mathbf{v}} \end{aligned} \tag{3.5}$$

3.3 Wavelets on Polygon Meshes

In the special case of a rectilinear mesh, a tensor-product DWT is defined by performing a one-dimensional DWT for all rows and then for all columns of the mesh. The corresponding tensor-product approach for the first lifting operation of the reconstruction rules (3.4),

$$\mathbf{v} \leftarrow \mathbf{v} - \tfrac{3}{4}\overline{\mathbf{e}}_{\mathbf{v}} \tag{3.6}$$

is illustrated in Figure 6. We note that the tensor-product lifting scheme also involves $\mathbf{f}$ vertices and that $\mathbf{e}$ vertices act like $\mathbf{v}$ vertices for exactly one direction.

The fundamental observation that makes our lifting approach possible is that this tensor-product lifting operation can be computed by modifying $\mathbf{e}$ vertices and $\mathbf{v}$ vertices separately, see Figure 7. This formulation does not require vertex valences to be four, and it is thus applicable to arbitrary polygonal base meshes.

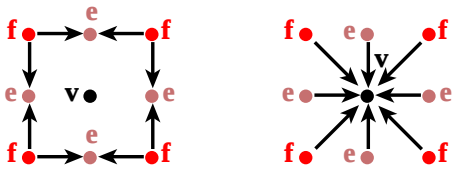


Figure 7: Same tensor-product lifting operation as in Figure 6, applied to arbitrary polygonal meshes.

A generalized tensor-product lifting operation for equation (3.6) is defined by the rules

$$\begin{aligned} \mathbf{e} &\leftarrow \mathbf{e} - \tfrac{3}{4}\overline{\mathbf{f}}_{\mathbf{e}} \\ \mathbf{v} &\leftarrow \mathbf{v} - \tfrac{9}{16}\overline{\mathbf{f}}_{\mathbf{v}} - \tfrac{3}{2}\overline{\mathbf{e}}_{\mathbf{v}} \end{aligned} \tag{3.7}$$

Due to the averaging operators, the total weight added to an individual vertex remains independent of its valence. This property ensures that the shapes of basis functions near extraordinary points are close to the shapes of corresponding basis functions on regular domains.

Analogously to the first reconstruction rule in (3.4) the two remaining rules can be generalized to arbitrary polygonal meshes. This is the entire set of reconstruction rules for the lifted generalized bicubic DWT:

$$\begin{aligned} &1.\quad \mathbf{e} \leftarrow \mathbf{e} - \tfrac{3}{4}\overline{\mathbf{f}}_{\mathbf{e}} \\ &2.\quad \mathbf{v} \leftarrow \mathbf{v} - \tfrac{9}{16}\overline{\mathbf{f}}_{\mathbf{v}} - \tfrac{3}{2}\overline{\mathbf{e}}_{\mathbf{v}} \\ &3.\quad \mathbf{e} \leftarrow \mathbf{e} + \overline{\mathbf{v}}_{\mathbf{e}} \\ &4.\quad \mathbf{f} \leftarrow \mathbf{f} - \overline{\mathbf{v}}_{\mathbf{f}} + 2\overline{\mathbf{e}}_{\mathbf{f}} \\ &5.\quad \mathbf{e} \leftarrow \tfrac{1}{2}\mathbf{e} + \tfrac{1}{2}\overline{\mathbf{f}}_{\mathbf{e}} \\ &6.\quad \mathbf{v} \leftarrow \tfrac{1}{4}\mathbf{v} - \tfrac{1}{4}\overline{\mathbf{f}}_{\mathbf{v}} + \overline{\mathbf{e}}_{\mathbf{v}}. \end{aligned} \tag{3.8}$$

This reconstruction formula defines linear subdivision and expansion operators, given by

$$\begin{pmatrix} \mathbf{v} \\ \mathbf{e} \\ \mathbf{f} \end{pmatrix} = \mathbf{S}(\mathbf{v}') + \mathbf{E}(\mathbf{e}', \mathbf{f}'). \tag{3.9}$$

On a rectilinear grid, the subdivision operator $\mathbf{S}$ reproduces bicubic B-splines (scaling functions) as limit surfaces when no detail is added, *i.e.*, when all wavelet coefficients $\mathbf{e}'$ and $\mathbf{f}'$ are zero. On arbitrary polygonal base meshes, the subdivision scheme behaves similar to Catmull-Clark subdivision.

The corresponding decomposition rules are defined by inverting the rules (3.8) in reverse order:

$$\begin{aligned} &1.\quad \mathbf{v} \leftarrow 4\mathbf{v} + \overline{\mathbf{f}}_{\mathbf{v}} - 4\overline{\mathbf{e}}_{\mathbf{v}} \\ &2.\quad \mathbf{e} \leftarrow 2\mathbf{e} - \overline{\mathbf{f}}_{\mathbf{e}} \\ &3.\quad \mathbf{f} \leftarrow \mathbf{f} + \overline{\mathbf{v}}_{\mathbf{f}} - 2\overline{\mathbf{e}}_{\mathbf{f}} \\ &4.\quad \mathbf{e} \leftarrow \mathbf{e} - \overline{\mathbf{v}}_{\mathbf{e}} \\ &5.\quad \mathbf{v} \leftarrow \mathbf{v} + \tfrac{9}{16}\overline{\mathbf{f}}_{\mathbf{v}} + \tfrac{3}{2}\overline{\mathbf{e}}_{\mathbf{v}} \\ &6.\quad \mathbf{e} \leftarrow \mathbf{e} + \tfrac{3}{4}\overline{\mathbf{f}}_{\mathbf{e}} \end{aligned} \tag{3.10}$$

In analogy to equation (3.2), these decomposition rules define linear fitting and compaction-of-difference operators, given by

$$\begin{aligned} \mathbf{v}' &= \mathbf{F}(\mathbf{v}, \mathbf{e}, \mathbf{f}) \quad \text{and} \\ \begin{pmatrix} \mathbf{e}' \\ \mathbf{f}' \end{pmatrix} &= \mathbf{C}(\mathbf{v}, \mathbf{e}, \mathbf{f}). \end{aligned} \tag{3.11}$$

Our DWT is now applied as follows: a polygonal base mesh defining a surface topology is recursively subdivided using Catmull-Clark subdivision structure until a prescribed resolution is obtained. A mapping between the fine mesh and the geometry that needs to be represented is established. Coordinates for each mesh vertex are estimated so that the fine mesh (and the limit surface obtained by $\mathbf{S}$) approximates the geometry closely. Algorithms for these steps are described in the next sections. The actual input for our DWT is a hierarchical mesh structure with associated vertex coordinates at the finest subdivision level. Decomposition rules (3.10) are applied recursively from fine to coarse until the base mesh is obtained. The latter now contains control points for the geometry at coarsest resolution. The wavelet coefficients corresponding to vertices on any finer subdivision level contain surface details. A control mesh represents, at any resolution, a smooth generalized bicubic B-spline surface.

3.4 Boundary Curves and Feature Lines

Boundary curves and *sharp feature lines* need to be treated differently in the wavelet scheme in order to avoid a large number of non-zero wavelet coefficients and to predict coarser levels of resolution more precisely. Feature lines for subdivision surfaces were described for Loop subdivision by Hoppe *et al.* [19], and for other subdivision schemes by DeRose *et al.* [8] and Zorin *et al.* [40]. Boundaries and features correspond to marked edges in the base-mesh that are subdivided like B-spline curves defined by $\mathbf{v}$- and $\mathbf{e}$ vertices. It is also possible to define *sharp vertices* that cannot be modified by any subdivision rule. (To improve the surface quality in the neighborhood of sharp vertices, adjacent edges are treated like sharp edges.)

To handle sharp edges and vertices correctly, our subdivision operator $\mathbf{S}$ must apply one-dimensional subdivision rules for all $\mathbf{v}$ and $\mathbf{e}$ vertices belonging to sharp edges and it must not modify sharp vertices. Therefore, our reconstruction rules (3.8) need to be modified in the following way:

- For any $\mathbf{e}$ vertex located on a sharp edge or belonging to an edge with a sharp vertex, the first and fifth rules are ignored.
- For any sharp $\mathbf{v}$ vertex, the second and sixth rules are ignored.
- For any $\mathbf{v}$ vertex that has incident sharp edges and that is not sharp itself, the second and sixth rules are replaced by

  $$2.\quad \mathbf{v} \leftarrow \mathbf{v} - \tfrac{3}{4}\overline{\mathbf{e}}_{\mathbf{v}} \quad \text{and} \quad 6.\quad \mathbf{v} \leftarrow \tfrac{1}{2}\mathbf{v} + \tfrac{1}{2}\overline{\mathbf{e}}_{\mathbf{v}}.$$

  For both rules, the average $\overline{\mathbf{e}}_{\mathbf{v}}$ is computed from only those $\mathbf{e}$ vertices that correspond to sharp edges.

The decomposition rules (3.8) are modified analogously.

3.5 Subdivision and Fitting Properties

The four linear operators $\mathbf{S}$, $\mathbf{E}$, $\mathbf{F}$, and $\mathbf{C}$ correspond to non-square matrices that can be multiplied with corresponding sets of control points or wavelet coefficients. Considering that our wavelet transform has an inverse, it holds that $\mathbf{FS}$ and $\mathbf{CE}$ are identity matrices. Since the subdivision operator $\mathbf{S}$ can reproduce every bicubic polynomial on a regular (rectilinear) control mesh, the fitting operator $\mathbf{F}$ has bicubic precision as well. This implies that all wavelet coefficients are zero when representing a bicubic polynomial. We note that $\mathbf{F}$ does not interpolate certain surface points, but that it closely approximates the finer levels of resolution. Interpolation constraints often introduce unwanted "wiggles" to curves and surfaces.

A property of the subdivision operator $\mathbf{S}$ is that it produces limit surfaces with tangent-plane continuity at extraordinary points. This can be shown by exploring a square sub-matrix of $\mathbf{S}$ transforming a local set of control points around an extraordinary vertex into another set of points with the same local mesh topology. Eigenanalysis of this matrix characterizes the limit behavior at the extraordinary point, *i.e.,* the surface limit and its normal can be computed analytically from the eigenvectors. Our subdivision scheme falls into a class of generalized Catmull-Clark schemes for which the eigenvalues and eigenvectors of a local subdivision matrix have been evaluated by Ball/Storry [1]. A proof of tangent-plane continuity is provided by Peters/Reif [33].

Our subdivision scheme generates polynomial patches satisfying C^2-continuity constraints in all regular mesh regions. Except for extraordinary points, all surface regions become regular after a sufficient number of subdivisions. Around every extraordinary point there is an infinite number of smaller and smaller patches. However, it is possible to compute the limit-surface efficiently at arbitrary parameter values based on eigenanalysis of subdivision matrices [35].

4 Surface Fitting

Using the wavelet construction described previously, we can efficiently compute detail coefficients at multiple levels of surface resolution when a base mesh and control points on the finest subdivision level are given. In order for the transform to apply to an arbitrary input surface, the surface must be re-mapped to one with subdivision connectivity. In this section we introduce an efficient algorithm to construct base meshes and to subdivide and fit them to an isosurface. Our approach begins with the high-resolution, unstructured triangulation obtained by conventional contour extraction methods. This triangulation is simplified to form an initial coarse base mesh, which we subsequently subdivide and shrink-wrap to fit the input triangulation with a smoothly-parameterized mapping that helps minimize wavelet coefficient magnitudes and thus improves compression efficiency.

4.1 Constructing Base Meshes

We construct a initial base mesh by performing a variant on edge-collapse simplification due to Hoppe [20], followed by an additional pass that removes some edges while keeping the vertices fixed. This class of simplification method works on triangulations (manifold with boundary), and preserves the genus of the surface, resulting in polygonal meshes. We constrain the simplification to produce polygons with three-, four- and five-sided faces, and vertices of valences from 3 to 8. These constraints result in higher quality mappings and compression efficiency, since very low or high degree vertices and faces cause highly skewed or uneven parameterizations in the subsequent fitting process.

Our variant on edge-collapse simplification uses a fast priority queue implemented using bucket sorting, where within each bucket of priorities the legal edge collapses are listed in first-in-first-out order. Collapses are performed on the highest-priority legal edge, iterating until no legal collapses are possible or a target vertex count is met. We mark an edge as legal to collapse if the rules from [20] and certain additional constraints hold. We ensure that valences on any new vertex are in the range $\{3, \ldots, 8\}$ (valence 2 is allowed for boundary vertices). We disallow collapses on edges with two sharp vertices. The new vertex formed by the edge collapse is located at the edge center if neither edge vertex is sharp. In the case where one edge vertex is sharp, it becomes the new vertex. The general configuration is shown in Figure 8.

Collapsing edge $v_i v_j$ decreases by one the valence of vertices a and b belonging to incident triangles as shown in Figure 8 and replaces v_i and v_j by a single vertex v of valence $N_i + N_j - 4$, where N_i and N_j are the valences of v_i and v_j, respectively. We

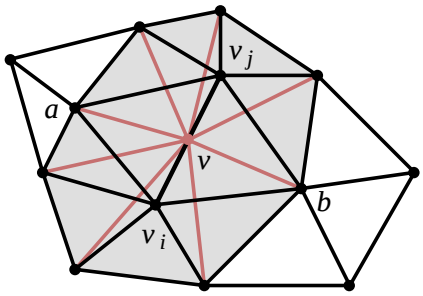


Figure 8: Collapsing edge $v_i v_j$ modifies the triangulation in the shaded region. The error estimates for all edges shown are updated.

check the valences that a, b and v would have after collapse using these formulas to enforce the desired constraints.

The error function used in our mesh-collapse algorithm is simpler than Hoppe's and does not require evaluating shortest distances of a dense sampling of points to the finest mesh every time an edge is updated. It was demonstrated by Lindstrom and Turk [26] that memoryless simplification can provide results at least as good as the methods of Hoppe and others but without expensive metrics computed with respect to the original fine mesh. The priority computation we have developed has some similarity in spirit, but has been re-formulated in response to experience attempting to optimize the subsequent fitting process and resulting mappings. We feel it is important to test the volume-preserving aspect of the Lindstrom-Turk edge collapses in future work, since this has the potential to reduce some artifacts encountered in especially aggressive reductions (*e.g.*, > 50 times reductions for the turbulence contours we examine).

The priority for an edge collapse is computed as follows. Let c_i be the vectors obtained by taking cross products of consecutive vectors associated with the ring of edges adjacent to the new vertex after collapse. Then $n_i = c_i / \|c_i\|_2$ are the unit normals to the triangles forming a ring around the new vertex. The unit average normal is $n = 1/k \sum_{i=0}^{k-1} n_i$. The error introduced by an edge collapse is estimated by

$$\Delta E_0 = \max_{i=0,\ldots,k-1} \left| \frac{d}{2} \cdot n_i \right| \tag{4.1}$$

where $d = v_j - v_i$. In order to allow priority distinctions in skewed/tangled planar neighborhoods, we clamp ΔE_0 to a small positive value by $\Delta E_1 = \max\{\Delta E_0, 10^{-8}\}$. Errors in higher-curvature or tangled neighborhoods are emphasized by a weighting factor, giving the final delta in error energy as

$$\Delta E = \left(2 - \frac{\min c_i \cdot n}{\max \|c_i\|_2}\right)^{16} \Delta E_1 \tag{4.2}$$

We keep an estimate of accumulated error for each edge neighborhood, E_{acc}. The original edges are initialized with zero accumulated error. The total error for an edge is defined as

$$E = E_{\text{acc}} + \Delta E \tag{4.3}$$

When an edge is collapsed, the accumulated error for any edge adjacent to the new vertex is set to the maximum of its previous value E_{acc} and the values E for the five old edges destroyed by the collapse. The priority of a collapse is given by $\log(1/E)$. This value is discretized, for example, to about 10^5 buckets within a maximum expected range of $[\log 10^{-10}, \log 10^{10}]$, to give a bucket index.

Upon collapse, a neighborhood of nearby edges must have their legality markings and priorities updated. These edges are: (a) those adjacent to the new vertex or to the two old vertices remaining from the triangles that collapsed to edges, and (b) the ring of edges formed between consecutive outer endpoints of the edges in (a). The complete stencil of edge updates is shown in Figure 8.

To improve the vertex and face valences of the base mesh, we delete some edges that satisfy certain constraints, in a priority-queue order. An edge is eligible for deletion if its incident vertices both have valences at least four, and if the resulting merged face has no more than five sides. Sharp edges are never eligible for deletion. If the unit normals of the faces on either side of the edge have a dot product less that a specified threshold, for example $.5$, then we also make the edge ineligible for removal. The priority for removal is formulated to be higher when the new face has four sides, when the two faces being combined have similar normals, and when high-valence vertices (valence > 4) are on the ends. Removal priority is lower when the new face has more than four sides, when the two combining faces have disparate normals, and when the edge's vertices have valence 4.

4.2 Isosurface Fitting

Given the initial base mesh from the edge collapse and removal procedures, a refinement fitting procedure is the final step in converting the contour surface to have subdivision-surface connectivity, a fair parameterization, and a close approximation to the original unstructured geometry. Our method is inspired by the *shrink-wrapping* algorithm by Kobbelt *et al*. [23], which models an equilibrium between *attracting forces* pulling control points towards a surface and *relaxing forces* minimizing parametric distortion. We iterate the attraction/relaxation phases a few times at a given resolution, then refine using Catmull-Clark subdivision, repeating until a desired accuracy or resolution is attained. Relaxation is provided by a simple Laplacian averaging procedure, where each vertex is replaced by the average of its old position and the centroid of its neighbor vertices. Relaxation for a vertex adjacent to two sharp edges only weights the adjacent vertices on those edges. Sharp vertices are not relaxed. The remainder of this section describes the attraction method.

For attraction, vertices are moved to the actual isosurface along a line defined by the unit average normal of the faces adjacent to the vertex in the current shrink-wrap mesh. The location chosen along this line is determined by use of a signed-distance field for the original contour surface. We choose the current-mesh normal direction to ensure that samples spread evenly over the surface, especially for high-curvature features. The even spread is facilitated by the mesh relaxation procedure. The signed-distance field is used to help locate the best attraction point because it is a reliable indicator of which way to move and how far, and can help disambiguate between near isosurface locations by selecting the one facing a direction that most agrees with the mesh normal. The scalar field itself, while available a no additional space or time cost, is generally not reliable for these things. We move to the nearest isosurface along the mesh-normal line that has a contour normal facing in the same direction (the dot product of the two normals is positive). If the distance to this location is greater that a specified threshold, then the point is left where it was until further iterations/refinements provide a sensible target location.

We briefly outline the algorithm we use for computing the signed-distance field for the contour. If the scalar field is defined on a regular hexahedral grid, we use the same grid for the signed-distance function. The sign for our distance function can be obtained from the underlying scalar field while estimating the distance to the isosurface involves more work. Our algorithm creates a breadth-first queue of "updated" nodes in the grid, initialized to include the grid nodes for cells containing isosurface, using the the isolevel, scalar value and scalar-field gradient to estimate these initial distances. Each queue entry contains the node index and coordinates of the closest surface point found so far for that node. The

first entry on the queue is removed, and all its neighbors are checked to see if they need to be updated. A neighbor is updated if the removed nodes closest point is closer that its closest point. Updating involves replacing the coordinates of the closest point, placing the neighbor at the end of the queue, and storing the new distance in the distance field for the neighbor's node. The queue processing continues until the queue is empty. Typically each node gets updated only a few times, resulting in very fast computation of the signed-distance field.

We note that an isosurface of a signed-distance function will have slight differences from the original extracted isosurface, hence the fitting process will converge to a slightly different surface than may be desired. This can be optionally corrected after fitting, by moving the vertices in the scalar-gradient direction to the exact isosurface. This is possible after the fitting process because the points are within a fraction of a cell width from the exact surface and the scalar field is reliable when in that proximity.

5 Results

To demonstrate the performance of our algorithm, we have extracted an isosurface from a block of a high-resolution turbulent-mixing hydrodynamics simulation [30], converted it into our surface representation and displayed different levels of resolution. Starting with a block of $256 \times 256 \times 384$ samples, we have constructed an isosurface mesh composed of 976,321 vertices. This mesh has been simplified to a base mesh with only 19,527 vertices. We have used three subdivision steps for the shrink-wrapping process and obtained a fine-resolution mesh composed of 1,187,277 vertices, which corresponds to the total number of control points and wavelet coefficients. We obtained computation times of 12 minutes for base mesh generation, about one minute for the shrink-wrapping step, and 30 seconds for computing the wavelet transform on a 250 MHz MIPS R10000 processor.

No. of coefficients	Percent of full resolution	RMSE [%]
237,490	20.0	7.6
118,728	10.0	20.5
59,364	5.0	41.7
19.527	1.6	71.3

Table 1: Root mean square errors in percent of edge length of volume cell for reconstructions from subsets of coefficients.

Assuming that the control points of the shrink-wrapped mesh interpolate the isosurface, we can estimate the root mean square error (RMSE) for a mesh reconstructed from only a subset of coefficients by using differences between control points at finest resolution. Error estimates are shown in Table 1. The errors are computed in percent of the edge length of one volume cell. The main diagonal of the entire block is about 528 edge lengths. The coarsest resolution possible can be obtained by reconstruction from the base mesh, which corresponds to 1.6 percent of the full resolution. We note that every wavelet coefficient is a vector-valued quantity with three components.

The color plate shows the base mesh with interpolating control points and different levels of resolution from two different points of view. All figures are rendered at a mesh resolution corresponding to three subdivision levels (same resolution as obtained from shrink-wrapping) using flat shading.

6 Future Work

We have introduced a powerful multiresolution modeling tool for highly detailed surfaces of arbitrary topology and described an algorithm to convert large-scale isosurfaces into this parametric representation. Future work will be directed at constructing a wider range of lifted biorthogonal wavelets, including smooth interpolating schemes that are efficiently generated by generalizing local lifting operations to arbitrary base mesh topology. It is also desirable to incorporate local topology changes into the wavelet transform, which would provide more flexibility in the choice of base meshes and increase the number of resolution levels. We plan to improve the shrink-wrapping algorithm to detect local topology changes and to handle them appropriately.

Acknowledgements

We thank Chuck Hansen for his many suggestions on an early draft of this paper. This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. It was also supported by the National Science Foundation under contracts ACI 9624034 and ACI 9983641 (CAREER Awards), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Office of Naval Research under contract N00014-97-1-0222; the Army Research Office under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; and the North Atlantic Treaty Organization (NATO) under contract CRG.971628 awarded to the University of California, Davis. We also acknowledge the support of ALSTOM Schilling Robotics, Chevron, Silicon Graphics, Inc. and ST Microelectronics, Inc. We thank the members of the Visualization Thrust at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis.

References

[1] A.A. Ball, D.J.T. Storry, *Conditions for tangent plane continuity over recursively generated B-spline surfaces*, ACM Transactions on Graphics, Vol. 7, No. 2, April 1988, pp. 83–102.

[2] G.-P. Bonneau, *Multiresolution analysis on irregular surface meshes*, IEEE Transactions on Visualization and Computer Graphics, Vol. 4, No. 4, IEEE, Oct.-Dec. 1998, pp. 365–378.

[3] K.S. Bonnell, M.A. Duchaineau, D.R. Schikore, B. Hamann, and K.I. Joy, *Constructing Material Interfaces from Data Sets with Volume-fraction Information*, Proceedings of Visualization '00, IEEE Computer Society Press, 2000.

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

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

[6] C.K. Chui, *An Introduction to Wavelets*, Academic Press, San Diego, California, 1992.

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

[8] T. DeRose, M. Kass, and T. Truong, *Subdivision surfaces in character animation*, Computer Graphics (Proceedings of SIGGRAPH '98), ACM, 1998, pp. 85–94.

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

[10] M.A. Duchaineau, *Dyadic Splines*, Ph.D. Thesis, Department of Computer Science, University of California, Davis, 1996. `http://graphics.cs.ucdavis.edu/~duchaine/dyadic.html`

[11] N. Dyn, D. Levin, and J.A. Gregory, *A butterfly subdivision scheme for surface interpolation with tension control*, ACM Transactions on Graphics, Vol. 9, No. 2, April 1990, pp. 160–169.

[12] M. Eck and H. Hoppe, *Automatic reconstruction of B-spline surfaces of arbitrary topological type*, Computer Graphics (Proceedings of SIGGRAPH '96), ACM, 1996, pp. 325–334.

[13] *G. Farin, Curves and Surfaces for CAGD*, Fourth edition, Academic press, San Diego, California, 1997.

[14] C.M. Grimm and J.F. Hughes, *Modeling surfaces of arbitrary topology using manifolds*, Computer Graphics (Proceedings of SIGGRAPH '95), ACM, 1995, pp. 359-368.

[15] B. Guo, *Surface reconstruction: from points to splines*, Computer Aided Design, Vol. 29, No. 4, April 1997, pp. 269–277.

[16] I. Guskov, W. Sweldens, and P. Schröder, *Multiresolution signal processing for meshes*, Computer Graphics (Proceedings of SIGGRAPH '99), ACM, 1999, pp. 325–334.

[17] M. Halstead, M. Kass, and T. DeRose, *Efficient, fair interpolation using Catmull-Clark surfaces*, Computer Graphics (Proceedings of SIGGRAPH '93), ACM, 1993, pp. 35–44.

[18] P.S. Heckbert, M. Garland, *Multiresolution modeling for fast rendering.*, Proceedings Graphics Interface '94, Canadian Inf. Process. Soc, 1994, pp. 43–50, 246.

[19] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, *Piecewise Smooth Surface Reconstruction*, Computer Graphics (Proceedings of SIGGRAPH '94), ACM, 1994, pp. 295–302.

[20] H. Hoppe, *Progressive meshes*, Computer Graphics (Proceedings of SIGGRAPH '96), ACM, 1996, pp. 99–108

[21] L. Kobbelt, *Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology*, Computer Graphics Forum 15, Proceedings of Eurographics '96, Blackwell Publishers, 1996, pp. 409–420.

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

[23] L.P. Kobbelt, J. Vorsatz, U. Labsik, H.-P. Seidel, *A shrink wrapping approach to remeshing polygonal surfaces* Computer Graphics Forum, Vol. 18, Proceedings of Eurographics '99, Blackwell Publishers, 1999, pp. 119–129.

[24] V. Krishnamurthy and M. Levoy, *Fitting smooth surfaces to dense polygon meshes*, Computer Graphics (Proceedings of SIGGRAPH '96), ACM, 1996, pp. 313–324.

[25] A.W.F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin, *MAPS: multiresolution adaptive parameterization of surfaces*, Computer Graphics, Proceedings of SIGGRAPH '98, ACM 1998, pp. 95–104.

[26] P. Lindstrom, G. Turk, *Evaluation of memoryless simplification*, IEEE Transactions on Visualization and Computer Graphics, Vol. 5, No. 2, 1999.

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

[28] J.M. Lounsbery, *Multiresolution analysis for surfaces of arbitrary topological type*, Ph.D. Thesis, Department of Mathematics, University of Washington, 1994.

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

[30] A.A. Mirin, R.H. Cohen, B.C. Curtis, W.P. Dannevik, A.M. Dimits, M.A. Duchaineau, D.E. Eliason, D.R. Schikore, S.E. Anderson, D.H. Porter, P.R. Woodward, L.J. Shieh and S.W. White, *Very High Resolution Simulation of Compressible Turbulence on the IBM-SP System*, Supercomputing 99 Conference, Portland, Oregon, November 1999.

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

[32] G.M. Nielson, I.-H. Jung, and J. Sung, *Haar wavelets over triangular domains with applications to multiresolution models for flow over a sphere*, Proceedings of Visualization '97, IEEE Computer Society Press, 1997, pp. 143–150.

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

[34] P. Schröder and W. Sweldens, *Spherical wavelets: efficiently representing functions on the sphere*, Computer Graphics (Proceedings of SIGGRAPH '95 Proc), ACM, 1995, pp. 161–172.

[35] J. Stam, *Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values*, Computer Graphics (Proceedings of SIGGRAPH '98 Proc), ACM, 1998, pp. 395–404.

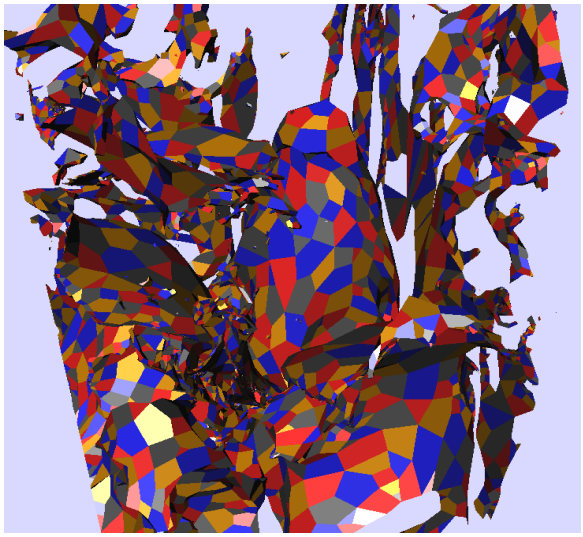
[36] E.J. Stollnitz, T.D. DeRose, and D.H. Salesin, *Wavelets for Computer Graphics–Theory and Applications*, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1996.

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

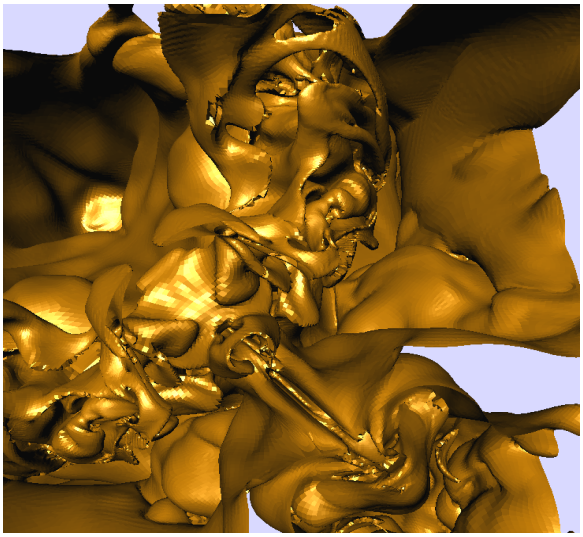
[38] H. Tao and R.J. Moorhead, *Progressive transmission of scientific data using biorthogonal wavelet transform*, in : R.D. Bergeron and A.E. Kaufman, eds., Proceedings of Visualization '94, IEEE Computer Society Press, 1994, pp. 93–99.

[39] D. Zorin, P. Schröder, and W. Sweldens, *Interpolating Subdivision for Meshes with Arbitrary Topology*, Computer Graphics (Proceedings of SIGGRAPH '96), ACM, 1996, pp. 189–192.

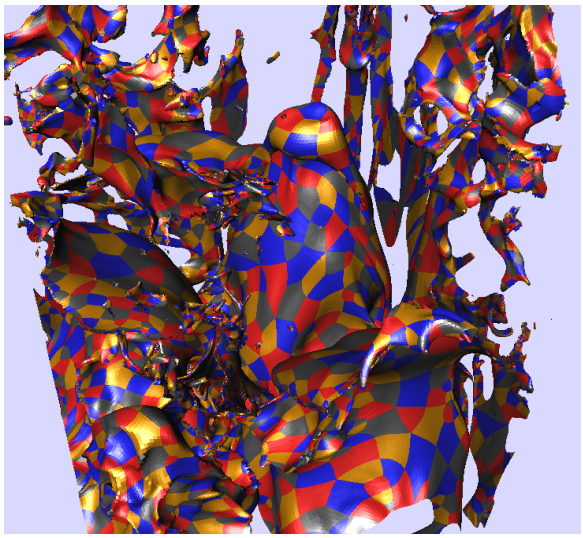
[40] D. Zorin, H. Biermann, A. Levin, *Piecewise Smooth Subdivision Surfaces with Normal Control*, Technical Report No. 781, Courant Institute, New York University, 1999.



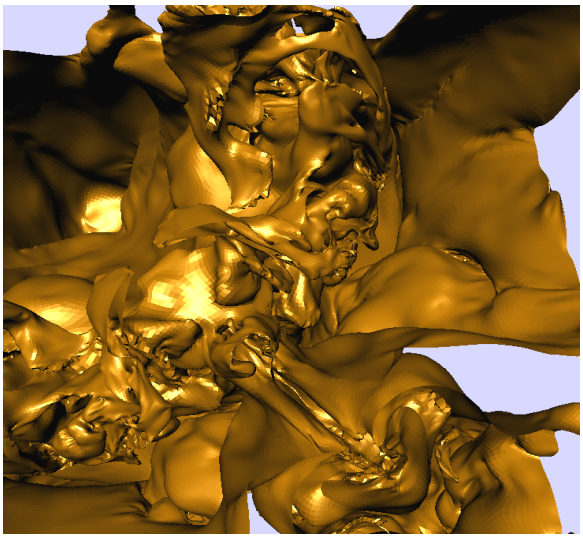
(a) Base mesh (side view)



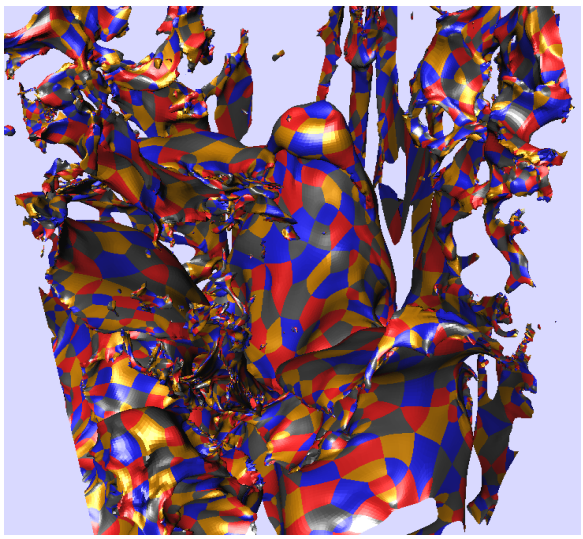
(b) Full resolution (bottom view)



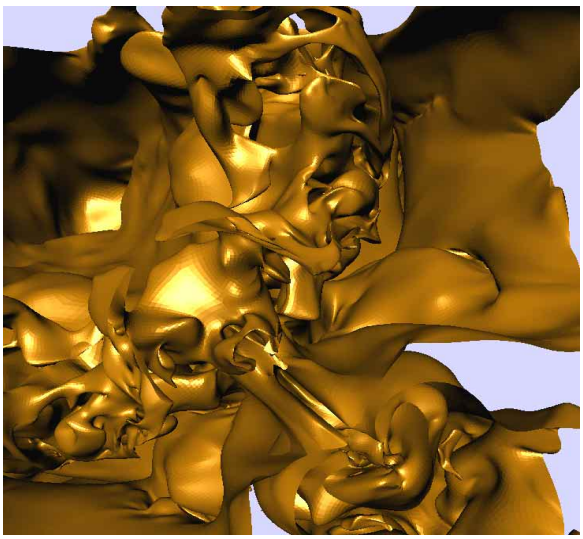
(c) Full resolution



(d) 5% of coefficients used



(e) 5% of coefficients used



(f) 1.6% of coefficients used