WAVELETS FOR ADAPTIVELY REFINED $\sqrt[3]{2}$-SUBDIVISION MESHES

L. Linsen*† B. Hamann† K.I. Joy†

* Department of Mathematics and Computer Science
Ernst-Moritz-Arndt-Universität Greifswald
Friedrich-Ludwig-Jahn-Str. 15a, 17487 Greifswald, Germany

† Institute for Data Analysis and Visualization (IDAV)
Department of Computer Science
University of California, Davis
One Shields Avenue, Davis, CA 95616-8562, U.S.A.

ABSTRACT

For view-dependent visualization, adaptively refined volumetric meshes are used to adapt resolution to given error constraints. A mesh hierarchy based on the $\sqrt[3]{2}$-subdivision scheme produces structured grids with highest adaptivity. Downsampling filters reduce aliasing effects and lead to higher-quality data representation (in terms of lower approximation error) at coarser levels of resolution. We present a method for applying wavelet-based downsampling filters to adaptively refined meshes. We use a linear B-spline wavelet lifting scheme to derive narrow filter masks. Using these narrow masks, the wavelet filters are applicable to adaptively refined meshes without imposing any restrictions on the adaptivity of the meshes, i. e., all wavelet filtering operations can be performed without further subdivision steps. We define rules for vertex dependencies in wavelet-based adaptive refinement and resolve them in an unambiguous manner. We use the wavelet filters for view-dependent visualization in order to demonstrate the functionality and the benefits of our approach. When using wavelet filters the approximation quality is higher at each resolution level. Thus, less polyhedra need to be traversed by a visualization method to meet certain error bounds / quality measures.

KEY WORDS

View-dependent visualization, multiresolution modeling, B-spline wavelets, lifting, downsampling filter, data approximation.

*linsen@uni-greifswald.de
†{hamann,joy}@cs.ucdavis.edu

1 Introduction

Due to substantial improvements in computing power and imaging and sensor technologies in recent years, today's data-intensive applications are generating huge amounts of data in shorter and shorter time frames. Simulating three-dimensional phenomena, measuring scalar fields in a three-dimensional environment, or scanning with three-dimensional devices lead to large-scale volume data, possibly varying over time. In-core data exploration and visualization tools cannot be applied to such data sets at highest resolution. Using multiresolution approaches, downsampling can be used to reduce data sets to manageable sizes, and a multiresolution hierarchy is employed to represent data at various levels of resolution.

Applying in-core visualization tools to an appropriate level of resolution can lead to rather low-quality images. A well-known concept in computer graphics and visualization used to improve image quality in a hierarchical setting is view-dependent refinement: Instead of visualizing an entire data set at the same level of resolution, resolution is adapted to viewing parameters in the three-dimensional scene. Regions close to the viewpoint and the line of sight should be given higher priority. In such regions, data should be available at "sufficiently" high resolution, whereas other regions can remain at lower resolutions. What is considered to be sufficient, depends on the application, the user, the resolution of the output device, and frame-rate requirements.

To obtain better approximations of a data set at coarse levels of resolution, downsampling filters can be applied when generating a multiresolution hierarchy. In [15, 16], we showed that filters based on linear B-spline wavelets significantly improve approximation quality. However, downsampling filters based on non-constant B-spline wavelets can only be applied efficiently when defined over structured (rectilinear, tensor-product type) grids permitting the use of a regular downsampling / refinement scheme. On the other hand, view-dependent visualization can be performed more successfully when the refinement scheme supports high adaptivity. Thus, we use the $\sqrt[3]{2}$-subdivision scheme, a regular refinement scheme with finest granularity supporting high adaptivity. The splitting steps of the $\sqrt[3]{2}$-subdivision scheme are equivalent to longest-edge bisection applied to a tetrahedral mesh. We describe multiresolution hierarchy construction based on $\sqrt[3]{2}$ subdivision in Section 3.

Non-constant B-spline wavelets have the property that the computation of the wavelet coefficient at a vertex $\mathbf{p}$ is not only based on the neighbors of $\mathbf{p}$ but also on vertices farther away. Larger filters reduce adaptivity of a multiresolution representation. Lifting schemes with narrow filters can be used to overcome this problem. In Section 4, we describe a lifting scheme for linear B-spline wavelets.

The application of the lifting scheme to an adaptive setting is not straightforward. In an adaptive setting, vertices are represented at different levels of resolution, whereas the lifting scheme requires all of them to belong to the same level. In Section 5, we discuss how the wavelet downsampling filters can be applied to an adaptive setting using the lifting scheme. In Section 6, we show how this wavelet-based adaptive setting is used for view-dependent visualization.

2 Related work

Adaptive refinement of meshes and view-dependent visualization techniques were developed when intensive research on terrain rendering started, about a decade ago. To date, many view-dependent approaches exist for heightfield-like surfaces [4, 2, 9, 12, 36] and also for more general polygonal surfaces [8, 13, 14, 21, 3]. For scalar-valued volume data, one can extract multiresolution hierarchies of isosurfaces [5, 35]. However, these approaches are not suitable for large-scale volume visualization, since storing all hierarchies of all possibly important isosurfaces would require too much storage.

Recent techniques use multiresolution volume representation and extract view-dependent isosurfaces from adaptively refined volume data [7, 19, 20, 28]. These approaches are typically based on structured grids and regular refinement schemes [18, 22, 24, 26, 30, 34, 37] for establishing a multiresolution hierarchy, since vertex positions and mesh connectivity are implicitly defined for structured grids, leading to faster data access and loading, which is crucial for interactivity.

One major drawback of approaches based on structured grids is low adaptivity, which is of high importance for view-dependent visualization. The multiresolution hierarchy with highest adaptivity is a tetrahedral mesh hierarchy based on longest-edge bisection [6, 7, 24, 37]. The splitting step of the $\sqrt[3]{2}$-subdivision scheme is equivalent to longest-edge bisection for tetrahedral meshes. In fact, for implementation purposes we have used tetrahedral meshes, since they can be supported by existing visualization tools. The derivations and descriptions of the techniques are more easily explained when using the (more general) $\sqrt[3]{2}$-subdivision scheme for (initially) hexahedral meshes.

3 Multiresolution with $\sqrt[3]{2}$ subdivision

A multiresolution hierarchy based on $\sqrt[3]{2}$ subdivision is constructed by starting with the coarsest resolution of a given mesh and iteratively applying $\sqrt[3]{2}$-subdivision steps. The subdivision steps can be performed simultaneously for all mesh elements.

The splitting step of the $\sqrt[n]{2}$-subdivision scheme was described by Cohen and Daubechies [1] for dimension $n = 2$ and Maubach [23] for arbitrary dimension n. Figure 1 illustrates four splitting steps of a $\sqrt{2}$ subdivision ($n = 2$). To split the quadrilateral Q, we compute its centroid $\mathbf{c}$ and connect $\mathbf{c}$ to the four vertices of Q. The "old" edges of the mesh are removed (except for the edges determining the mesh/domain boundary). Velho and Zorin [33] completed the $\sqrt{2}$-subdivision scheme by adding an averaging step to the splitting step. They showed that the produced surfaces are C^4-continuous at regular and C^1-continuous at extraordinary vertices.

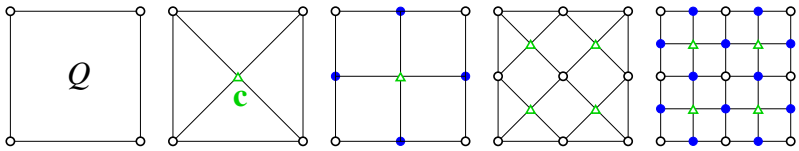


Figure 1. Four steps of $\sqrt{2}$ subdivision applied to quadrilateral Q.

This subdivision scheme can be generalized to arbitrary dimension [15, 16, 25]. The splitting step of the $\sqrt[n]{2}$ subdivision is executed by inserting the centroid of the n-dimensional geometrical shapes and adjusting vertex connectivity. The averaging step applies to every old vertex $\mathbf{v}$ the linear update rule

$$\mathbf{v} = \alpha\mathbf{v} + (1 - \alpha)\mathbf{w} \ ,$$

where $\mathbf{w}$ is the centroid of the adjacent new vertices and $\alpha \in [0, 1]$.

Figure 2 shows three $\sqrt[3]{2}$-subdivision splitting steps ($\alpha = 1$) for structured rectilinear volume data. Three kinds of polyhedral shapes arise, shown in Figure 3.

With respect to the start configuration (first picture of Figure 2), the three subdivision steps can be described in the following way: The first step inserts the centroid of the cuboid (second picture of Figure 2); the second step inserts the centers of the faces of the original cuboid (third picture of Figure 2); and the third step inserts the midpoints of the edges of the original cuboid (fourth picture of Figure 2).

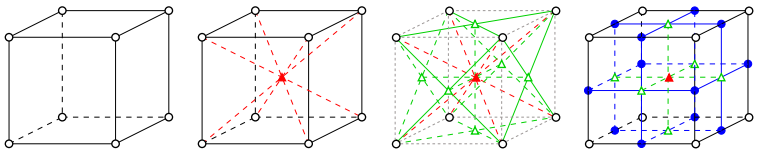


Figure 2. Three steps of $\sqrt[3]{2}$ subdivision applied to a cube.

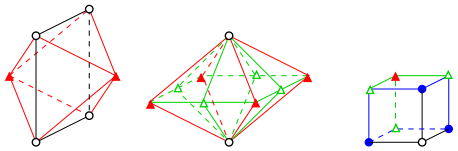


Figure 3. Polyhedral shapes created by $\sqrt[3]{2}$ subdivision: octahedron, octahedron with split faces, and cuboid.

Thus, three $\sqrt[3]{2}$-subdivision steps produce the same result as one octree refinement step. The two steps in between lead to finer granularity and thus higher adaptivity.

4 Wavelet lifting scheme

When downsampling volume data in a regular fashion, data is not "grouped" due to variation in the data. Thus, aliasing artifacts occur and important details may be missing at coarse levels of resolution. We overcome this problem by using downsampling filters. In image processing, such downsampling filters are commonly employed with wavelets.

A family of filters can be derived by using B-splines of various degrees for wavelet generation. (For an introduction to B-spline techniques, we refer to [27].) However, when using non-constant B-splines, the size of the wavelet filters is not limited to adjacent vertices. Localization is desirable when we want to apply wavelet filters to adaptive refinement and out-of-core visualization techniques. Lifting schemes, as introduced by Sweldens [31], decompose wavelet computations into several steps, but they assert narrow filters. We review the idea of lifting, and lifting of B-spline wavelets for $\sqrt[3]{2}$ subdivision in particular.

The idea of a lifting scheme is shown in Figure 4, using the example of linear B-spline wavelets. For downsampling, the vertices of a level of resolution $\mathcal{L}_n$ are split into two groups: the ones that belong to the next coarser level of resolution $\mathcal{L}_{n-1}$ (often referred to as *even* vertices) and the ones that belong to $\mathcal{L}_n \setminus \mathcal{L}_{n-1}$ (often referred to as *odd* vertices). Instead of applying a large downsampling filter to the vertices in $\mathcal{L}_{n-1}$, the lifting scheme decomposes the large filter into two narrow ones and executes two steps. First, one narrow filter (*w-lift*) is applied to the vertices in $\mathcal{L}_n \setminus \mathcal{L}_{n-1}$. Second, the other narrow filter (*s-lift*) is applied to the vertices in $\mathcal{L}_{n-1}$. This process is usually referred to as *encoding*, and the values at the vertices in $\mathcal{L}_n \setminus \mathcal{L}_{n-1}$ are called *wavelet coefficients*. The *decoding* step inverts the two encoding steps and reconstructs level $\mathcal{L}_n$ from level $\mathcal{L}_{n-1}$ using the wavelet coefficients.

The lifting filters can be described by *masks*. For example, the one-dimensional B-spline wavelet lifting filters are given by

$$\text{s-lift}(a,b): \quad \begin{pmatrix} a & b & a \end{pmatrix} \text{ and} \tag{1}$$

$$\text{w-lift}(a,b): \quad \begin{pmatrix} a & b & a \end{pmatrix}. \tag{2}$$

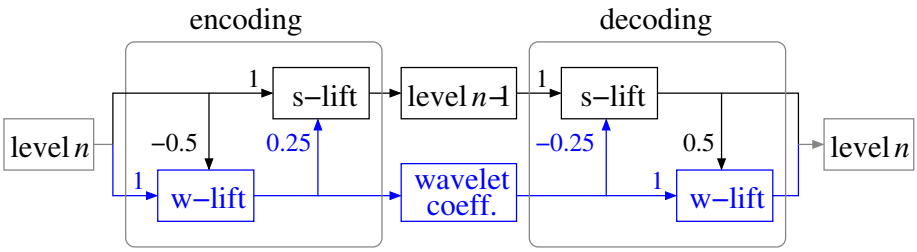


Figure 4. One-dimensional linear B-spline wavelet lifting: By consecutively applying w-lift and s-lift operation, a coarser resolution level $\mathcal{L}_{n-1}$ and corresponding wavelet coefficients are computed from a finer resolution level $\mathcal{L}_n$. Inverse lifting, i. e., computing level $\mathcal{L}_n$ from level $\mathcal{L}_{n-1}$ and the corresponding wavelet coefficients, is executed by applying the inverse lifting opertaions in inverse order.

Using the s-lift and w-lift masks, a linear B-spline wavelet encoding step is defined by sequentially executing the two operations

$$\text{w-lift}(-\tfrac{1}{2}, 1) \quad \text{and}$$
$$\text{s-lift}(\tfrac{1}{4}, 1)\ .$$

A linear B-spline wavelet decoding step is defined by executing the inverse operations in reverse order. They are

$$\text{s-lift}(-\tfrac{1}{4}, 1) \quad \text{and}$$
$$\text{w-lift}(\tfrac{1}{2}, 1)\ .$$

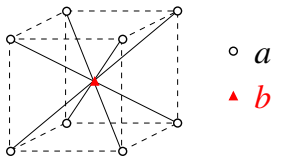
The 1D filters in Equations (1) and (2) can be generalized to 2D filters for quadrilateral meshes (representing tensor-product surfaces) in a quadtree-like setting and to 3D filters for hexahedral meshes in an octree-like setting by convolution of the 1D masks in the two or three coordinate directions, respectively. For example, one would get the 2D mask by

$$\begin{pmatrix} a & b & a \end{pmatrix} * \begin{pmatrix} a \\ b \\ a \end{pmatrix} = \begin{pmatrix} a^2 & ab & a^2 \\ ab & b^2 & ab \\ a^2 & ab & a^2 \end{pmatrix}\ .$$

Wavelets for general dilation matrices were discussed by Riemenschneider and Shen [29], who used a box-spline approach. Kovačević and Vetterli [11] and, more recently, Uytterhoeven [32] and Kovačević and Sweldens [10] developed lifting schemes that can be applied to $\sqrt[n]{2}$-subdivision mesh hierarchies. Uytterhoeven's method only addresses the two-dimensional case, Kovačević and Sweldens' approach deals with the two- and three-dimensional cases. The filters used in [10] are not narrow enough for our purposes. We use the 3D masks for $\sqrt[3]{2}$-subdivision developed in [15].

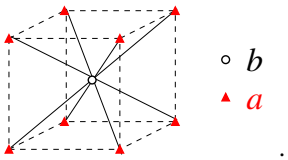
In a $\sqrt[3]{2}$-subdivision hierarchy, three different kinds of polygonal shapes appear. They are shown in Figure 3. Therefore, three different kinds of masks must be defined for the lifting filters:

1. We start with the situation shown in the second picture of Figure 2. The masks w-lift(a, b) are of the form



.

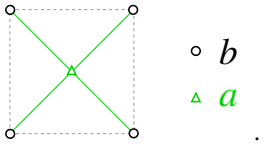
The masks s-lift(a, b) are of the form



.

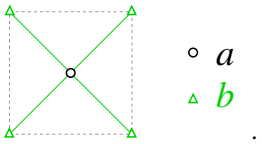
The masks are derived from the masks for octree-based hierarchies by linearly interpolating the values at the non-existing vertices. For a detailed derivation and the exact values for a and b we refer to [15].

2. In the situation shown in the third picture of Figure 2, the masks reduce to 2D masks when ensuring that we do not violate the linear-interpolation assumptions made in the above case. The masks w-lift(a, b) are of the form



.

The masks s-lift(a, b) are of the form



.

3. In the situation shown in the fourth picture of Figure 2, the masks reduce to 1D masks when ensuring that we do not violate the linear-interpolation assumptions. The masks w-lift(a, b) and s-lift(a, b) of Equations (1) and (2) can be applied.

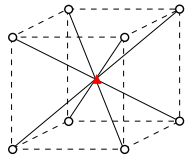
All masks are as narrow as they can be and thus appropriate for adaptive refinement. Moreover, the scheme naturally covers boundary faces and boundary edges of a mesh.

5 Wavelet lifting in adaptive setting

In an adaptive setting, the vertices of a mesh should not all belong to the same level of resolution. Thus, if a vertex is inserted, we cannot simply apply the update rules described in the previous section, but we have to make sure that the neighbor vertices (the ones belonging to the support of the update masks) are at the right levels of resolution. We may have to "raise" neighbor vertices to the appropriate level of resolution first.

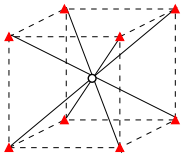
Moreover, when applying a local subdivision step to a polyhedron P, we cannot simply apply one mask to update the vertices of P, but we must apply the update rules for all adjacent polyhedra that share an updated vertex with P. When inserting a new vertex, we must update the values according to the lifting scheme, i.e., we first must execute s-lift operations for the neighbors of the new vertex and then w-lift operations for the new vertex. Again, we have to distinguish three cases, depending on the kind of vertex being inserted:

1. When inserting a vertex ▲ we must apply the following w-lift mask:



.

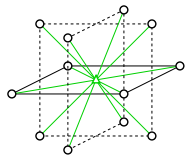
For linear B-spline wavelets, we obtain $a = \frac{1}{8}$ and $b = 1$. Let $\mathcal{N}_{\mathbf{p}}$ be the set of neighbor vertices of a vertex $\mathbf{p}$, which belong to the support of the update mask applied to $\mathbf{p}$. For example, if $\mathbf{p}$ is the vertex ▲, $\mathcal{N}_{\mathbf{p}}$ is the set of vertices ○. According to the lifting scheme, we first update the vertices ○ before updating ▲. We apply the following s-lift mask:



.

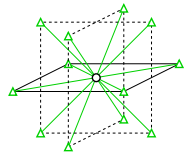
For linear B-spline wavelets, we obtain $a = -\frac{1}{64}$ and $b = 1$.

2. When inserting a vertex △ we must apply the two-dimensional w-lift masks from the previous section to all possible directions. These masks can be combined into a single w-lift mask, given as



.

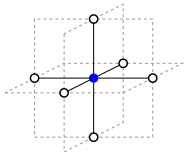
For linear B-spline wavelets, we obtain $a' = \frac{a}{3} = \frac{1}{12}$ and $b' = b = 1$. Again, we first update the vertices ○ before updating △. We combine the two-dimensional s-lift masks from the previous section in a single s-lift mask:



.

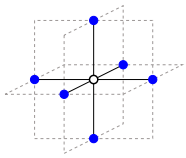
For linear B-spline wavelets, we obtain $a' = \frac{a}{3} = -\frac{1}{48}$ and $b' = b = 1$.

3. When inserting a vertex • we must apply the one-dimensional w-lift masks from the previous section to all directions. We combine them into a single w-lift mask, given as



.

For linear B-spline wavelets, we obtain $a' = \frac{a}{3} = \frac{1}{6}$ and $b' = b = 1$. We update the vertices ○ before updating •. We combine the one-dimensional s-lift masks from the previous section in a single s-lift mask:



For linear B-spline wavelets, we obtain $a' = \frac{a}{3} = -\frac{1}{48}$ and $b' = b = 1$.

When removing a vertex we apply the inverse masks in inverse order. The inverse masks have the same structure, but different values a and b.

The dependencies between vertices can be defined and resolved as follows: Initially, we assign a uniquely defined index n to every vertex in the multiresolution hierarchy, indicating to which level $\mathcal{L}_n$ a vertex belongs (see Section 4). The indexing starts with zero for the vertices belonging to the coarsest resolution level $\mathcal{L}_0$.

To illustrate how to resolve dependencies, we discuss the example shown in Figure 5. The first picture shows the start configuration with all vertices in $\mathcal{L}_0$. In the second picture, a first vertex ▲ is inserted, which must be in $\mathcal{L}_1$. The third picture shows that the values at the vertices ∘ are updated first by executing an s-lift operation (their indices are raised by one), before the value at vertex ▲ is updated by executing a w-lift operation, shown in the fourth picture (its index is being initialized).

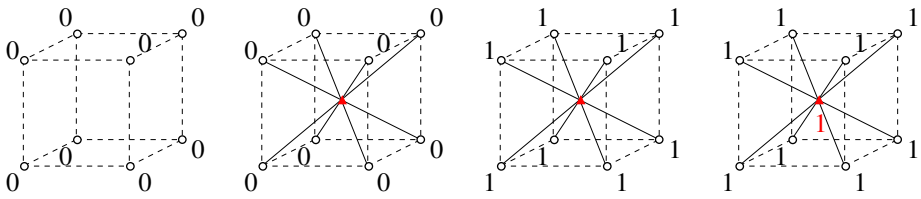


Figure 5. Wavelet lifting when inserting vertex ▲: Before updating vertex ▲, surrounding vertices need to be lifted to appropriate resolution level.

In general, when a vertex $\mathbf{p}$ with index n is inserted, a w-lift operation must be applied to update the value at that vertex. The execution of the w-lift requires that all vertices in $\mathcal{N}_\mathbf{p}$ are active and have index n. The subdivision scheme already ensures that all vertices in $\mathcal{N}_\mathbf{p}$ are active, before it inserts the vertex $\mathbf{p}$ [7].

We must ensure that the vertices in $\mathcal{N}_\mathbf{p}$ have index n. The appropriate s-lift operations are executed. Each execution of an s-lift operation raises the index by one. For the execution of an s-lift operation at a vertex $\mathbf{q} \in \mathcal{N}_\mathbf{p}$, the vertices in $\mathcal{N}_\mathbf{q}$ do not need to be active. In fact, they are never active, since the vertex $\mathbf{q}$ would already have been updated, if any vertex in $\mathcal{N}_\mathbf{q}$ had been inserted earlier. Moreover, the vertices in $\mathcal{N}_\mathbf{q}$ are not updated before they are selected for insertion and become active. At that point, the increased index of vertex $\mathbf{q}$ indicates that $\mathbf{q}$ has already been updated before.

We conclude that the wavelet filters do not reduce the adaptivity of the entire system, since no additional vertices have to be inserted for applying the wavelet lifting masks.

6 Wavelet-based view-dependent visualization

To validate our adaptive wavelet lifting scheme, we have applied it to view-dependent volume visualization. In a view-dependent setting, resolution should be high next to the viewpoint (or the focus of attention) and decrease with increasing distance from the viewpoint.

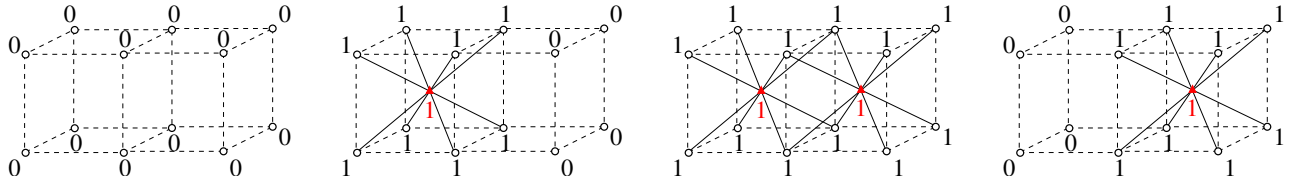


Figure 6. Reversing subdivision steps: All vertex dependencies must be checked when applying inverse lifting operations. Only to vertices with no dependencies inverse lifting can and (for consistency) has to be applied.

We define an approximation error $\mathcal{E}(P)$ for every polyhedron P according to an error metric. Following the approach described in [17], P is subdivided when its error $\mathcal{E}(P)$ is beyond a threshold, where the threshold increases with increasing distance from the viewpoint. Let $d(P)$ be the distance from P to the viewpoint, d_{max} the maximum distance from the viewpoint (or the range of sight), and $\mathcal{E}_{max}$ the maximum approximation error. P is subdivided when

$$\mathcal{E}(P) > \frac{d(P)}{d_{max}}\,\mathcal{E}_{max}\ .$$

The parameters d_{max} and $\mathcal{E}_{max}$ are application-specific and user-controlled. For fly-through data exploration, one can restrict subdivision steps to regions within the view frustum, which is defined by the range of sight d_{max} and a maximum deviation angle from the line of sight. Note that the adaptive resolution of the mesh is solely determined by the view-dependent visualization application. We do not perform a wavelet compression. The wavelet filters are used to improve the approximation quality for a given resolution and not to determine the adaptive resolution of the mesh.

For the definition of the approximation error, we use a simple and thus efficient error metric. Given the original function F at the discrete sample values, the error for a polyhedron P is defined as

$$\mathcal{E}(P) = \sqrt{\frac{1}{|P|}\sum_{\mathbf{x}\in P}\big(F(\mathbf{x}) - f(\mathbf{x})\big)^2}\ ,$$

where $|P|$ denotes the volume spanned by P and $f(\mathbf{x})$ the value at $\mathbf{x}$ linearly interpolated from the values at the vertices of P. We add the approximation errors at all vertices $\mathbf{x}$ of the finest resolution level that lie in P. One can define a screen-space error by projecting $\mathcal{E}(P)$ onto the screen. For some applications, we have used a more sophisticated, data-dependent error metric that also takes a chosen isovalue into account, see [6].

When a polyhedron P is subdivided, the vertex inserted by the subdivision step is updated (using a w-lift operation), in addition to all the vertices the inserted vertex is dependent on (using an s-lift operation). The latter are not necessarily vertices of P. When a subdivision step is reversed, the same vertices are updated again using the inverse lifting operations in reverse order. Some of the s-lift operations must not be reversed. For example, Figure 6 shows a mesh with two adjacent cuboids, where in a first step the first cuboid is subdivided, in a second step the second cuboid is subdivided, and in a third step the first subdivision step is reversed. To some of the vertices $\circ$, to which an s-lift operation has been applied in the first subdivision step (their new index is 1), we must not apply the reverse s-lift operation, since a new dependency has been generated by the second subdivision step.

Following these rules, the index of a vertex in an adaptively refined mesh is always uniquely defined and consistent with its neighborhood.

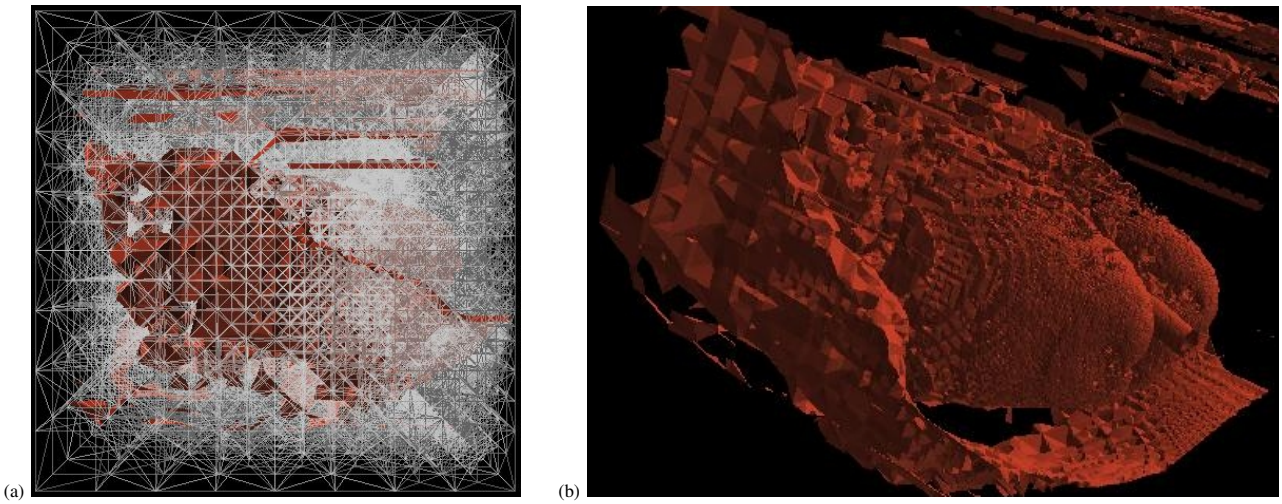


Figure 7. View-dependent visualization of Lung data set using adaptively refined $\sqrt[3]{2}$-subdivision mesh. Lifting schemes have been used to apply linear B-spline wavelet filters to the adaptive mesh hierarchy. (a) Adaptively refined $\sqrt[3]{2}$-subdivision mesh. (b) Visualization of isosurface, which is shown from the side to illustrate the decreasing resolution level with increasing distance from the focus located on the right hand side. (Data set courtesy of E. Wisner, Department of Surgical and Radiological Sciences, University of California, Davis)

7 Results

Figure 7 shows a view-dependent visualization of a primate lung data set. The size of the shown CT data set is $512 \times 512 \times 266$, and the range of the values is $[0, 255]$. For visualization purposes, we have extracted an isosurface (for isovalue 86) from an adaptively refined $\sqrt[3]{2}$-subdivision hierarchy, where linear B-spline wavelet filters have been used for generating the multiresolution hierarchy. The wavelet filters have been applied to the adaptive setting using the described lifting scheme. Figure 7(a) shows the adaptively refined mesh, where the viewpoint is located at the center of the "right" quadrilateral face of the bounding box. Figure 7(b) illustrates how view-dependent visualization works by showing the extracted isosurface from a point that does not coincide with the viewpoint used for view-dependent isosurface extraction. It can be seen clearly how the resolution changes from fine to coarse in the adaptively refined mesh.

In [15], we showed the improvement in quality when comparing the results obtained with wavelet filters to those obtained without wavelet filters in a non-adaptive setting. Typically, approximation error is 10 to 15% lower when applying wavelet filters, and error reduction is higher when considering coarser resolutions. For the view-dependent setting, we have observed the same behavior. Thus, in order to meet a certain error bound (as defined in the last section), we need to process less polyhedra and, when applying isosurface extraction, need to display less triangles. The amount of polyhedra and triangles we save is proportional to (and in the same range as) the error reduction.

In Figure 8, we provide a visual comparison of view-dependent visualization with and without wavelet filters. The data set is a CT scan of a Bonsai tree. The size of the data set is 256^3, and the range of the values is $[0, 255]$. Figure 8(a) shows an isosurface (for isovalue 42) extracted from an adaptively refined $\sqrt[3]{2}$-subdivision hierarchy without wavelet filters; Figure 8(b) shows the same isosurface extracted from an adaptively refined $\sqrt[3]{2}$-subdivision hierarchy with wavelet filters. Both hierarchies satisfy the same error bound. The hierarchy with wavelets filters requires 868 polyhedra less to meet the error criterion. The

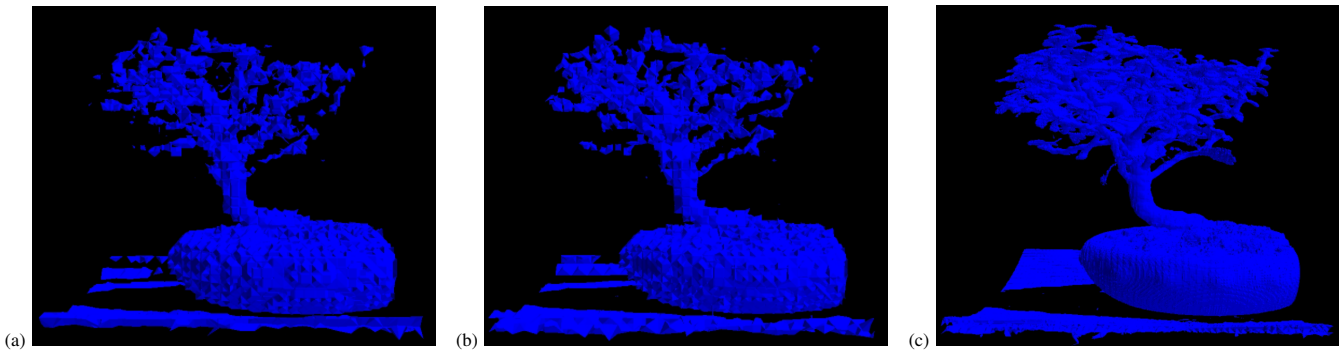


Figure 8. Comparing isosurfaces extracted from adaptively refined $\sqrt[3]{2}$-subdivision hierarchy without (a) and with (b) wavelets filters for Bonsai data set. The visualization with wavelet filters requires less geometry to meet same error criterion / approximation quality. (c) High-quality visualization of an adaptively refined $\sqrt[3]{2}$-subdivision hierarchy with wavelets filters. Adaptive refinement allows us to traverse less than 30% of original amount of geometry while producing images of same quality. (Data set courtesy of S. Roettger, Abteilung Visualisierung und Interaktive Systeme, University of Stuttgart, Germany)

extracted isosurface from the hierarchy with wavelets filters consists of 2454 fewer triangles. Thus, during both isosurface extraction and rendering computation time can be saved.

Figure 8(c) illustrates the benefits of using adaptive refinement techniques. The shown isosurface (for isovalue 42) is "visually equal" to the one extracted from the highest-resolution representation of the data set, but it is based on fewer than 30% of the original number of polyhedra. For adaptive refinement, we have used the data-dependent error metric described in [6]. The center of the data set has been used as focus of attention.

8 Conclusion

We have demonstrated how wavelet filters can be applied to adaptive refinement of 3D meshes. The used underlying mesh hierarchy is based on $\sqrt[3]{2}$ subdivision, which is a regular refinement scheme leading to structured grids with highest adaptivity. Since the splitting steps of the $\sqrt[3]{2}$-subdivision scheme are equivalent to longest-edge bisection steps applied to tetrahedral meshes, our techniques can be applied to both kinds of mesh hierarchies.

We have used a linear B-spline wavelet lifting scheme to derive narrow filter masks, which can be applied to adaptively refined $\sqrt[3]{2}$-subdivision meshes. We have shown that the application of the wavelet masks does not impose restrictions on the adaptivity of the meshes. All wavelet filtering operations can be performed without further subdivision steps. Moreover, when following the described rules to define and resolve vertex dependencies, every vertex is always represented at a uniquely defined level of resolution.

We have applied our wavelet filters to view-dependent visualization and have shown that our methods are functional and practical, that wavelet filters reduce approximation error in adaptively refined mesh hierarchies, and that wavelet-based adaptive refinement can reduce the amount of polyhedra to be processed for visualization purposes.

Acknowledgments

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award) and ACI 0222909, 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 National Institute of Mental Health and the National Science Foundation under contract NIMH 2 P20 MH60975-06A2; and the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159. We also acknowledge the support of ALSTOM Schilling Robotics and SGI. We thank the members of the Visualization and Graphics Research Group at the Institute for Data Analysis and Visualization (IDAV) at the University of California, Davis.

References

[1] Albert Cohen and Ingrid Daubechies. Nonseparable bidimensional wavelet bases. *Rev. Mat. Iberoamericana*, 9(1):51–137, 1993.

[2] Mark Duchaineau, Murray Wolinsky, David E. Sigeti, Mark C. Mille, Charles Aldrich, and Mark B. Mineev-Weinstein. Roaming terrain: Real-time optimally adapting meshes. In R. Yagel and H. Hagen, editors, *Proceedings of IEEE Conference on Visualization 1997*, pages 81–88. IEEE, IEEE Computer Society Press, 1997.

[3] Jihad El-Sana and Eitan Bachmat. Optimized view-dependent rendering for large polygonal datasets. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of IEEE Conference on Visualization 2002*, pages 77–84. IEEE, IEEE Computer Society Press, 2002.

[4] Leila De Floriani, Paola Magilla, and Enrico Puppo. Variant: A system for terrain modeling at variable resolution. *Geoinformatica*, 4(3):287–315, 2000.

[5] Marcel Gavriliu, Joel Carrance, David E. Breen, and Alan H. Barr. Fast extraction of adaptive multiresolution meshes with guaranteed properties from volumetric data. In Thomas Ertl, Ken Joy, and Amitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2001*, pages 295–302. IEEE, IEEE Computer Society Press, 2001.

[6] Jevan T. Gray, Lars Linsen, Bernd Hamann, and Kenneth I. Joy. Adaptive multi-valued volume data visualization using data-dependent error metrics. In Greg Turk, Jarke J. van Wijk, and Robert Moorhead, editors, *Proceedings of IEEE Conference on Visualization 2003*. IEEE, IEEE Computer Society Press, 2003.

[7] Benjamin Gregorski, Mark A. Duchaineau, Peter Lindstrom, Valerio Pascucci, and Kenneth I. Joy. Interactive view-dependent rendering of large isosurfaces. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of the IEEE Conference on Visualization 2002*, pages 475–482. IEEE, IEEE Computer Society Press, 2002.

[8] Hugues Hoppe. View-dependent refinement of progressive meshes. In Gordon Cameron, editor, *Proceedings of SIGGRAPH 1997*, Computer Graphics Proceedings, Annual Conference Series, pages 189–198. ACM, ACM Press / ACM SIGGRAPH, 1997.

[9] Hugues Hoppe. Smooth view-dependent level-of-detail rendering using cached geometry. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *Proceedings of IEEE Conference on Visualization 1998*, pages 35–42. IEEE, IEEE Computer Society Press, 1998.

[10] Jelena Kovačević and Wim Sweldens. Wavelet families of increasing order in arbitrary dimensions. *IEEE Transactions on Image Processing*, 9(3):480–496, 1999.

[11] Jelena Kovačević and Martin Vetterli. Nonseparable multidimensional perfect reconstruction filter banks and wavelet bases for $\mathbf{r}^n$. *IEEE Transactions on Information Theory*, 38(2):533–555, 1992.

[12] Joshua Levenberg. Fast view-dependent level-of-detail rendering using cached geometry. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of IEEE Conference on Visualization 2002*, pages 259–265. IEEE, IEEE Computer Society Press, 2002.

[13] Peter Lindstrom. Out-of-core simplification of large polygonal models. In Kurt Akeley, editor, *Proceedings of SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 259–262. ACM, ACM Press / ACM SIGGRAPH, 2000.

[14] Peter Lindstrom and Claùdio T. Silva. A memory insensitive technique for large model simplification. In Thomas Ertl, Ken Joy, and Amitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2001*, pages 121–126. IEEE, IEEE Computer Society Press, 2001.

[15] Lars Linsen, Jevan T. Gray, Valerio Pascucci, Mark A. Duchaineau, Bernd Hamann, and Kenneth I. Joy. Hierarchical large-scale volume representation with $\sqrt[3]{2}$ subdivision and trivariate b-spline wavelets. In Guido Brunnett, Bernd Hamann, Heinrich Müller, and Lars Linsen, editors, *Geometric Modeling for Scientific Visualization*. Springer-Verlag, Heidelberg, Germany, 2004.

[16] Lars Linsen, Valerio Pascucci, Mark A. Duchaineau, Bernd Hamann, and Kenneth I. Joy. Hierarchical representation of time-varying volume data with $\sqrt[4]{2}$ subdivision and quadrilinear b-spline wavelets. In Coquillart, Shum, and Hu, editors, *Proceedings of Tenth Pacific Conference on Computer Graphics and Applications – Pacific Graphics 2002*. IEEE, IEEE Computer Society Press, 2002.

[17] Lars Linsen and Hartmut Prautzsch. Fan clouds – an alternative to meshes. In T. Asano, R. Klette, and Ch. Ronse, editors, *Geometry, Morphology, and Computational Imaging, (Proceedings of Dagstuhl Seminar 02151 on Theoretical Foundations of Computer Vision)*, LNCS 2616 Theoretical Foundations of Computer Vision. IEEE, Springer-Verlag, 2003.

[18] L. Lippert, M. H. Gross, and C. Kurmann. Compression domain volume rendering for distributed environments. In *Proceedings of the Eurographics '97*, volume 14, pages 95–107. COMPUTER GRAPHICS Forum, 1997.

[19] Zhiyan Liu, Adam Finkelstein, and Kai Li. Progressive view-dependent isosurface propagation. In D. Ebert, J. M. Favre, and R. Peikert, editors, *Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization (VisSym-01)*, pages 223–232. Springer-Verlag, 2001.

[20] Yarden Livnat and Charles Hansen. View-dependent isosurface extraction. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *Proceedings of IEEE Conference on Visualization 1998*, pages 175–180. IEEE, IEEE Computer Society Press, 1998.

[21] David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In Gordon Cameron, editor, *Proceedings of SIGGRAPH 1997*, Computer Graphics Proceedings, Annual Conference Series, pages 199–208. ACM, ACM Press / ACM SIGGRAPH, 1997.

[22] Donald Maegher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19:129–147, 1982.

[23] Joseph M. Maubach. Local bisection refinement for n-simplicial grids generated by reflection. *SIAM J. Scientific Computing*, 16:210–227, 1995.

[24] Mario Ohlberger and Martin Rumpf. Hierarchical and adaptive visualization on nested grids. *Computing*, 59:365–385, 1997.

[25] Valerio Pascucci. Slow growing subdivision (sgs) in any dimension: towards removing the curse of dimensionality. In *Proceedings of Eurographics 2002*. COMPUTER GRAPHICS Forum, 2002.

[26] Dmitriy Pinskiy, Erie Brugger, Henry R. Childs, and Bernd Hamann. An octree-based multiresolution approach supporting interactive rendering of very large volume data sets. In H. Arabnia, R. Erbacher, X. He, C. Knight, B. Kovalerchuk, M. Lee, Y. Mun, M. Sarfraz, J. Schwing, and H. Tabrizi, editors, *Proceedings of the 2001 International Conference on Imaging Science, Systems, and Technology (CISST 2001), Volume 1*, pages 16–22. Computer Science Research, Education, and Applications Press (CSREA), Athens, Georgia, 2001.

[27] Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-spline Techniques*. Springer-Verlag, Heidelberg, Germany, 2002.

[28] Vijaya Ramachandran, Xiaoyu Zhang, and Chandrajit Bajaj. Paralel and out-of-core view-dependent isocontour visualization. In David Ebert, Pere Brunet, and Isabel Navaz, editors, *Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization (VisSym-02)*. Springer-Verlag, 2002.

[29] Sherman D. Riemenschneider and Zuowei Shen. Wavelets and pre-wavelets in low dimensions. *Journal Approximation Theory*, 71:18–38, 1992.

[30] Raj Shekhar, Elias Fayyad, Roni Yagel, and J. Fredrick Cornhill. Octree-based decimation of marching cubes surfaces. In Roni Yagel and Gregory M. Nielson, editors, *Proceedings of IEEE Conference on Visualization 1997*, pages 335–342. IEEE, IEEE Computer Society Press, 1996.

[31] Wim Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet constructions. In A. F. Laine and M. Unser, editors, *Wavelet Applications in Signal and Image Processing III*, pages 68–79. Proceedings of SPIE 2569, 1995.

[32] Geert Uytterhoeven. *Wavelets: Software and Applications*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 1999.

[33] Luiz Velho and Denis Zorin. 4-8 subdivision. *Computer-Aided Geometric Design*, 18(5):397–427, 2001.

[34] Rüdiger Westermann, Leif Kobbelt, and Thomas Ertl. Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Visual Computer*, pages 100–111, 1999.

[35] Zoë J. Wood, Mathieu Desbrun, Peter Schröder, and David Breen. Semi-regular mesh extraction from volumes. In Thomas Ertl, Bernd Hamann, and Abitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2000*, pages 275–282. IEEE, IEEE Computer Society Press, 2000.

[36] Julie C. Xia and Amitabh Varshney. Dynamic view-dependent simplification for polygonal models. In Roni Yagel and Gregory M. Nielson, editors, *Proceedings of IEEE Conference on Visualization 1996*, pages 335–344. IEEE, IEEE Computer Society Press, 1996.

[37] Yong Zhou, Baoquan Chen, and Arie E. Kaufman. Multiresolution tetrahedral framework for visualizing regular volume data. In Roni Yagel and Hans Hagen, editors, *Proceedings of IEEE Conference on Visualization 1997*, pages 135–142. IEEE, IEEE Computer Society Press, 1997.

Authors

Lars Linsen is an assistant professor of computer science at the Department of Mathematics and Computer Science of the Ernst-Moritz-Arndt-Universität Greifswald, Germany. He received a B.S. and an M.S. (Diplom) in computer science from the Universität Karlsruhe (TH), Germany, as well as a Ph.D. in 2001. He spent three years as a post-doctoral researcher and lecturer at the Institute for Data Analysis and Visualization (IDAV) and the Department of Computer Science of the University of California, Davis, U.S.A. He joined the Ernst-Moritz-Arndt-Universität Greifswald in October 2004. His research interests are in the areas of scientific and information visualization, multiresolution methods, computer graphics, and geometric modeling.

Bernd Hamann serves as associate vice chancellor for research and is full professor of computer science at the University of California, Davis. His main research interests are visualization, geometric modeling and computer-aided geometric design, computer graphics, and virtual reality. Bernd Hamann received a B.S. in computer science, a B.S. in mathematics, and an M.S. in computer science from the Technical University of Braunschweig, Germany. He received a Ph.D. in computer science from Arizona State University in 1991. He was awarded a 1992 National Science Foundation Research Initiation Award and a 1996 National Science Foundation CAREER Award. In 1995, he received a Hearin-Hess Distinguished Professorship in Engineering by the College of Engineering at Mississippi State University.

Kenneth I. Joy is a professor of computer science at the University of California at Davis, and co-director of the Institute for Data Analysis and Visualization (IDAV). He joined UC Davis in 1980 in the Department of Mathematics and was a founding member of the Computer Science Department in 1983. He is a faculty computer scientist at the Lawrence Berkeley National Laboratory, and a participating guest researcher at the Lawrence Livermore National Laboratory. His primary research interests are in the areas of visualization, Multiresolution representation of data, geometric modeling and computer graphics. Professor Joy serves on the editorial board of the IEEE Transactions on Visualization and Computer Graphics and served as a papers co-chair and proceedings co-editor for the IEEE Visualization conferences in 2001 and 2002.