Hierarchical Large-scale Volume Representation with $\sqrt[3]{2}$ Subdivision and Trivariate B-spline Wavelets

Lars Linsen[1], Jevan T. Gray[1], Valerio Pascucci[2], Mark A. Duchaineau[2], Bernd Hamann[1], and Kenneth I. Joy[1]

[1] Center for Image Processing and Integrated Computing (CIPIC)
Department of Computer Science
University of California, Davis
http://graphics.cs.ucdavis.edu/
llinsen@ucdavis.edu, {grayj, hamann, joy}@cs.ucdavis.edu
[2] Center for Applied Scientific Computing (CASC)
Data Science Group
Lawrence Livermore National Laboratory, Livermore
http://www.llnl.gov/casc/
{pascucci1, duchaineau1}@llnl.gov

Summary. Multiresolution methods provide a means for representing data at multiple levels of detail. They are typically based on a hierarchical data organization scheme and update rules needed for data value computation. We use a data organization that is based on what we call $\sqrt[n]{2}$ subdivision, where n is the dimension of the data set. The main advantage of $\sqrt[n]{2}$ subdivision, compared to quadtree ($n = 2$) or octree ($n = 3$) organizations, is that the number of vertices is only doubled in each subdivision step instead of multiplied by a factor of 2^n, i. e., four or eight, respectively. To update data values we use n-variate B-spline wavelets, which yield better approximations for each level of detail. We develop a lifting scheme for $n = 2$ and $n = 3$ based on the $\sqrt[n]{2}$-subdivision scheme. We obtain narrow masks that provide a basis for out-of-core techniques as well as view-dependent visualization and adaptive, localized refinement.

1 Introduction

Multiresolution schemes are used in computer graphics mainly for editing and rendering curves and surfaces at multiple levels of resolution. While most existing schemes could, in principle, be generalized for higher-dimensional data, only a few have been extended to data (or functions) defined over three- or even higher-dimensional domains. The combined subdivision-wavelet scheme we are describing in this paper is driven by the need to represent trivariate data (or functions) at multiple resolution levels for scientific visualization.

Representing volume data hierarchically is important in the context of "volume modeling" and visualizing volume data, e.g., scalar or vector fields defined over volumetric domains. Visualizing inherently trivariate phenomena

often requires one to apply rendering operations to volumetric data - examples being volume slicing via a cutting plane, isosurface extraction through marching-cubes-like algorithms, and ray casting. The multiresolution approximation approach we describe in this paper provides an elegant means of hierarchically organizing volume data, and we can use the resulting hierarchy to apply to its various levels volume data visualization methods.

We combine the $\sqrt[n]{2}$-subdivision hierarchy with update rules using n-variate B-spline wavelets to gain an n-dimensional multiresolution data representation. Multiresolution schemes have been studied extensively over the past decade. A survey of the main multiresolution approaches, considering also topological constraints, is given by Kobbelt [11]. These approaches can, for example, be used for a multiresolution representation of isosurfaces. However, when considering (bio-)medical imaging data, for instance, we must be able to switch quickly between isosurfaces corresponding to different isovalues, and when considering, for example, numerically simulated time-dependent hydrodynamics data, we even have to deal with isosurfaces changing over time. It is undesirable to store every single isosurface for all possibly important isovalues at different resolutions and load them during visualization. Instead, we devise a multiresolution volume data representation. We first develop a bivariate B-spline wavelet scheme for $\sqrt{2}$ subdivision and then generalize it to a trivariate B-spline wavelet scheme for $\sqrt[3]{2}$ subdivision. We have applied our techniques to bivariate as well as volumetric data.

For large-scale multiresolution representation, one should use regular rather than irregular data structures, since grid connectivity is implicit and data access simple for regular data. To overcome regular data structures' disadvantage of coarse granularity, we have developed the $\sqrt[n]{2}$-subdivision scheme we discuss in Section 3. Every $\sqrt[n]{2}$-subdivision step only doubles the number of vertices, which is a factor of $\sqrt[n]{2}$ in each of the n dimensions.

When using a wavelet scheme, the data value at a vertex $\mathbf{p}$ is updated when changing the level of detail, and thus the value varies with varying level of detail. On a coarse level, the value represents the value at $\mathbf{p}$ itself as well as an average value of a certain region around $\mathbf{p}$. This approach leads to better approximations on coarser levels. Wavelets based on the $\sqrt[n]{2}$-subdivision scheme unfortunately have the disadvantage of creating over- and undershoots. For example, for isosurface extraction ($n = 3$) this characteristic can cause creation of isosurfaces (or isosurface components) that are not existent in the full resolution. Therefore, we use n-variate B-spline wavelets, which do not create over- and undershoots, and adjust them to the $\sqrt[n]{2}$-subdivision scheme.

B-spline wavelets have the property that they do not only influence the neighbors of a vertex $\mathbf{p}$. Thus, when using out-of-core techniques to operate on or visualize large-scale data, a lot of data must be loaded from external memory with low I/O-performance. Furthermore, the adaptivity for view-dependent refinement techniques is restricted. Lifting schemes with narrow

filters can be used to overcome this problem. We review and generalize the lifting scheme from [3] in Section 4. In Sections 5 and 7, we develop a similar lifting scheme for the $\sqrt[n]{2}$-subdivision scheme for $n = 2$ and $n = 3$. We provide results in Sections 6 and 8.

2 Related work

Multiresolution volume representation is based on a 3D hierarchical data organization of irregular or regular type. Irregular data structures, see [4, 8, 7], use non-uniform refinement steps, which makes them highly adaptive. On the other hand, grid information must be stored and data access is not straightforward. Especially for large-scale data, additional memory requirements and memory organization needs are a major disadvantage of irregular structures.

For regular data organizations, octrees, see [15, 17, 20, 24, 27, 34]; and tetrahedral grids, see [20], are common. For regular structures, grid connectivity is implicit and data is easily and quickly accessed. However, the refinement steps have to conform to the topological constraints, which makes regular structures less adaptive. To overcome this disadvantage, we use the $\sqrt[3]{2}$-subdivision scheme, a regular data organization supporting finer granularity. While, for example, an octree refinement step doubles the number of vertices in every dimension, which leads to a factor of eight, a $\sqrt[3]{2}$-subdivision step only doubles the overall number of vertices. Therefore, $\sqrt[3]{2}$ subdivision will, in general, require less vertices than octrees to satisfy specified image quality error bounds. Since finer granularity leads to higher adaptivity this fact still holds when using adaptive refinement techniques.

The splitting step of the $\sqrt[n]{2}$-subdivision scheme was introduced by Cohen and Daubechies [5] for $n = 2$ and Maubach [18] for arbitrary n. It can be described by using triangular as well as quadrilateral meshes ($n = 2$) or their counterparts for higher dimensions. In the following, we will consider the quadrilateral case and its generalization.

The refinement step of the approaches described in [6, 22, 35] is a longest-edge bisection applied to tetrahedral meshes. This step is equivalent to the splitting step of the $\sqrt[3]{2}$-subdivision scheme. However, these approaches do not represent a full subdivision scheme, since the averaging step is missing. Thus, these schemes are restricted to structured-rectilinear grids, where eight cuboids share a common vertex, and the cuboids have the same size. The $\sqrt[3]{2}$-subdivision scheme also applies to structured-curvilinear grids, where hexahedra of arbitrary shape (but with linear edges) are used instead of cuboids. The scheme can even handle extraordinary vertices, see [21].

Recently, Velho and Zorin [32] introduced $\sqrt{2}$-subdivision surfaces ($n = 2$) by adding an averaging step. They showed that the produced surfaces are C^4-continuous at regular and C^1-continuous at extraordinary vertices. (For an introduction to subdivision methods, we refer to [33].)

The main advantage of wavelet schemes is the fact that they provide a means to generate good approximations in a multiresolution hierarchy. Stollnitz et al. [28] described how to generate wavelets for subdivision schemes. However, $\sqrt[n]{2}$-subdivision wavelets can lead to over- and undershoots, see Figure 7(b), which are especially disturbing when extracting isosurfaces from different levels of approximation. They can even cause topological changes of isosurfaces when changing the level of resolution. Therefore, we have decided to generate B-spline wavelets for the $\sqrt[n]{2}$-subdivision scheme, which are known to produce good approximations. (For an introduction to B-spline techniques, we refer to [25].)

The computation of wavelet coefficients at a certain vertex for wavelets with good approximation quality like B-spline wavelets is not limited to using only adjacent vertices. Localization, however, is strongly desirable when we want to apply the wavelet scheme to adaptive refinement and to out-of-core visualization techniques. Lifting schemes as introduced by Sweldens [29] decompose wavelet computations into several steps, but they assert narrow filters, see Figure 6. Bertram et al. [2, 3] defined a lifting scheme for 1D and 2D B-spline wavelets using a quadtree organization of the vertices.

Wavelets for general dilation matrices go back to Riemenschneider and Shen [26] who used a box-spline approach. Kovačević and Vetterli [13] and, more recently, Uytterhoeven [31] and Kovačević and Sweldens [12] developed lifting schemes that can be applied to $\sqrt[n]{2}$-subdivision data structures. Uytterhoeven [31] only addressed the 2D case. In [12], the filters that produce good approximations are not narrow enough for our purposes. On the other hand, the update rule for the narrow filters in [12] is the identity, which does not support the creation of good approximations.

Another main difference between our approach and the non-separable filters used in [31] and [12] is the update rule. For example, we update the vertices in a $\sqrt[3]{2}$-subdivision scheme by applying first the 3D, then the 2D, and finally the 1D update rules. This approach automatically includes the boundary cases, which are not sufficiently addressed in [31] and [12]. Moreover, the generalization to arbitrary dimension n is straight-forward.

3 The $\sqrt[n]{2}$-subdivision scheme

We first describe the case $n = 2$. For a $\sqrt{2}$-subdivision step of a quadrilateral Q, we compute its centroid $\mathbf{c}$, and connect $\mathbf{c}$ to all four vertices of Q. The "old" edges of the mesh are removed (except for the edges determining the mesh/domain boundary). Figure 1 illustrates four $\sqrt{2}$-subdivision steps.

The mask used for the computation of the centroid $\mathbf{c}$ is given in Figure 2(a). Figure 2(b) shows the mask of the averaging step according to [32]. A $\sqrt{2}$-subdivision step is executed by first applying the mask shown in Figure 2(a), which inserts the new vertices, and then (after the topological mesh

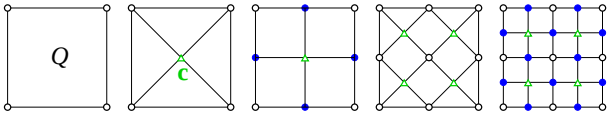


Fig. 1. $\sqrt{2}$ subdivision.

modifications) applying the mask shown in Figure 2(b), which repositions the old vertices.

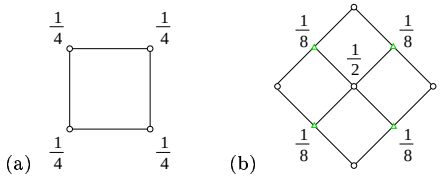


Fig. 2. Masks of $\sqrt{2}$-subdivision step: (a) inserting centroid; (b) repositioning old vertices.

This subdivision scheme for quadrilaterals is analogous to the $\sqrt{3}$-subdivision scheme of Kobbelt [10] for triangles. Therefore, we call it $\sqrt{2}$ subdivision.

We now generalize the subdivision scheme to $\sqrt[n]{2}$ subdivision for arbitrary dimension n. The splitting step is executed by inserting the centroid and adjusting vertex connectivity. The averaging step applies to every old vertex $\mathbf{v}$ the update rule

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

where $\mathbf{w}$ is the centroid of the adjacent new vertices.

We are especially interested in the case $n = 3$. Little research has been done to date concerning 3D (volumetric) subdivision. One example is the work described in [16]. The $\sqrt[n]{2}$-subdivision scheme in this general setting is discussed in [21]. The literature currently provides no analysis of averaging steps for dimensions larger than two. Thus, at present, we cannot provide a solution for the choice of α used in the update rule.

However, when applying the $\sqrt[3]{2}$-subdivision scheme to large volumetric data sets, we usually deal with structured-rectilinear grids, especially when considering imaging data sets. For structured-rectilinear grids, the update rule does not change the position of the vertices regardless of the specific α value, but it only affects the values at the vertices. In Section 6, we show that the $\sqrt{2}$-subdivision wavelets are not appropriate for our purposes, and we replace them by B-spline wavelets. Thus, we do not need to choose a value for α.

In Figure 3, three $\sqrt[3]{2}$-subdivision steps are shown. In each step, the centroids of the polyhedral shapes are inserted, and the connectivity is adjusted. Three kinds of polyhedral shapes arise. They are shown in Figure 4.

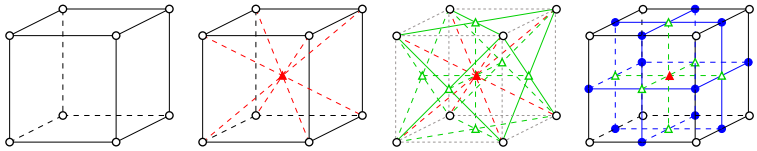


Fig. 3. $\sqrt[3]{2}$ subdivision.

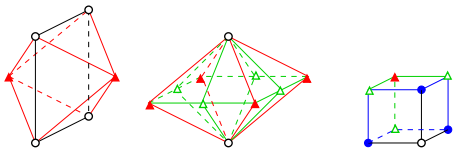


Fig. 4. Polyhedral shapes created by $\sqrt[3]{2}$ subdivision: octahedron, octahedron with split faces, and cuboid.

In the first step, each cuboid (first picture of Figure 3 / third picture of Figure 4) is subdivided by inserting the cuboid's centroid and connecting the centroid to all old vertices (second picture of Figure 3). In the second step, each octahedron (first picture of Figure 4) is subdivided by inserting the octahedron's centroid and connecting the centroid to all old vertices, while all old edges, except the edges inserted in the last subdivision step, are deleted (third picture of Figure 3). In the third step, each octahedron with split faces (second picture of Figure 4) is subdivided by inserting its centroid and connecting the centroid to all old vertices, except the vertices inserted in the next-to-the-last subdivision step (▲), while all old edges, except the edges between the vertices inserted in the next-to-the-last subdivision step (▲) and the vertices inserted in the last step (△), are deleted (fourth picture of Figure 3).

The three subdivision steps can also be described in the following way: The first step inserts the centroid of the cuboid, the second step inserts the centers of the faces of the original cuboid, and the third step inserts the midpoints of the edges of the original cuboid. Three $\sqrt[3]{2}$-subdivision steps produce the same result as one octree refinement step. Hence, for multiresolution purposes, we obtain a much finer granularity through $\sqrt[3]{2}$ subdivision, which reduces the complexity of the scenes to be rendered. If, for example, the resolution in the second picture of Figure 3 suffices to meet a certain screen-space error bound, a $\sqrt[3]{2}$-subdivision hierarchy can provide this resolution, whereas an octree-hierarchy would have to use four times the amount of data (as in the fourth picture of Figure 3). Thus, using the $\sqrt[3]{2}$-subdivision approach, one must render much less data to obtain a desired image quality. The additional effort spent when using the octree approach does not lead to better results, since the improvements projected onto screen are in subpixel range.

4 The B-spline wavelet lifting scheme

In this section, we review and define masks for the 1D lifting scheme of [2] and generalize them to the 2D and 3D cases. In the following sections, we will adjust the 2D lifting scheme to $\sqrt{2}$ subdivision and the 3D lifting scheme to $\sqrt[3]{2}$ subdivision.

The 1D B-spline wavelet lifting scheme makes use of two operations that are defined by the following two masks, called s-lift and w-lift:

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

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

The s-lift mask is applied to the old vertices ∘ and their new neighbors •, whereas the w-lift mask is applied to the new vertices • and their neighbors ∘, see Figure 5(a). For a detailed derivation of the lifting scheme that we use as a basis for this paper, as well as for its analysis (smoothness, stability, approximation order, and zero moments), we refer to [1].

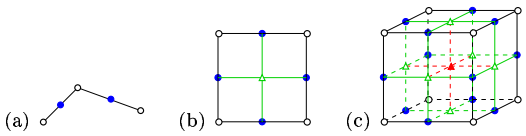


Fig. 5. Refinement step for 1D, 2D, and 3D meshes.

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

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

A linear B-spline wavelet decoding step is defined by sequentially executing the two operations

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

Figure 6 illustrates the 1D lifting scheme.

When applying 2D B-spline wavelets to a quadtree-organized set of vertices, two kinds of new vertices are obtained when executing a refinement step, namely the new vertices inserted at the midpoints • of old edges and the new vertices inserted at the centers △ of old faces, see Figure 5(b). Therefore, we have two different masks. We derive the needed 2D masks by convolution of the 1D masks in the two coordinate directions. This leads to:

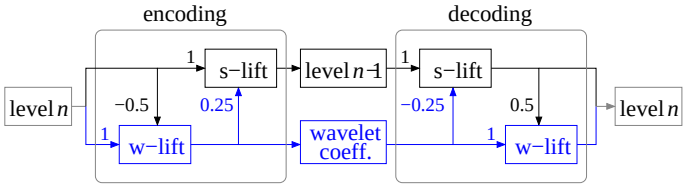


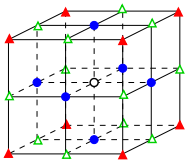
Fig. 6. 1D linear B-spline wavelet lifting scheme.

$$\text{s-lift}(a,b): \quad \begin{pmatrix} a^2 & ab & a^2 \\ ab & b^2 & ab \\ a^2 & ab & a^2 \end{pmatrix}, \quad \begin{pmatrix} a & b & a \end{pmatrix}, \tag{3}$$

$$\text{w-lift}(a,b): \quad \begin{pmatrix} a^2 & ab & a^2 \\ ab & b^2 & ab \\ a^2 & ab & a^2 \end{pmatrix}, \quad \begin{pmatrix} a & b & a \end{pmatrix}. \tag{4}$$

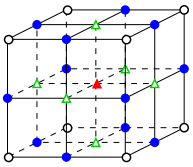
The 1D masks defined by (3) and (4) are applied in both directions. The masks in (3) as well as the masks in (4) are applied simultaneously.

When applying 3D B-spline wavelets to an octree-organized set of vertices, three kinds of new vertices are obtained when executing a refinement step, namely the new vertices inserted at the midpoints • of old edges, the new vertices inserted at the centers △ of old faces, and the new vertices inserted at the centroids ▲ of old cubes, see Figure 5(c). Therefore, we have three different masks. For 3D masks, we show the structure of the mask and separately define the values for the vertices ∘, •, △, and ▲. We derive the needed 3D masks by convolution of the 1D masks in all three coordinate directions. The s-lift(a,b) masks are defined by this depiction:



$$\begin{matrix} \circ\ b^3 \\ \bullet\ ab^2 \\ \triangle\ a^2b \\ \blacktriangle\ a^3 \end{matrix}, \quad \begin{pmatrix} a^2 & ab & a^2 \\ ab & b^2 & ab \\ a^2 & ab & a^2 \end{pmatrix}, \quad \begin{pmatrix} a & b & a \end{pmatrix}. \tag{5}$$

The 1D, 2D, and 3D masks are applied simultaneously to update the vertices ∘, •, and △, respectively. The w-lift(a,b) masks are defined by this depiction:



$$\begin{matrix} \circ\ a^3 \\ \bullet\ a^2b \\ \triangle\ ab^2 \\ \blacktriangle\ b^3 \end{matrix}, \quad \begin{pmatrix} a^2 & ab & a^2 \\ ab & b^2 & ab \\ a^2 & ab & a^2 \end{pmatrix}, \quad \begin{pmatrix} a & b & a \end{pmatrix}. \tag{6}$$

5 A lifting scheme for $\sqrt{2}$ subdivision

Using $\sqrt{2}$ subdivision instead of a quadtree-based scheme, we only obtain new vertices at the centers △ of old faces when executing a subdivision step;

at the midpoints • of old edges, no vertices are inserted, see second picture in Figure 1 and compare to Figure 5(b). Thus, no data is available at the positions of the vertices •, and we must adjust the 2D masks in (3) and (4).

For encoding with linear B-spline wavelets, the w-lift operation is executed first. Since we have no values at the positions • required for mask (4), we linearly interpolate the values at the vertices ∘. Linear interpolation is appropriate, since we are using linear wavelets. This approach changes mask (4) to

$$\text{w-lift}_{encode}(a,b): \qquad \begin{pmatrix} a^2+ab & & a^2+ab \\ & b^2 & \\ a^2+ab & & a^2+ab \end{pmatrix} . \tag{7}$$

Next, the s-lift operation is executed. Again, we have entries at the positions • in mask (3). However, the w-lift operation has (theoretically) executed the 1D mask in (4), and we assumed that the values at the vertices • were linear interpolations of the values at the vertices ∘; therefore, the values at the vertices • have vanished. Mask (3) changes to

$$\text{s-lift}_{encode}(a,b): \qquad \begin{pmatrix} a^2 & & a^2 \\ & b^2 & \\ a^2 & & a^2 \end{pmatrix} . \tag{8}$$

For decoding, we first execute the s-lift operation. Prior to executing the s-lift operation of the encoding, the values at the vertices • have vanished, but the s-lift operation (theoretically) executed the 1D mask in (3). Hence, the values at the vertices • are now given by linear interpolation of the values at the neighbor vertices △ multiplied by the factor $2a$ of the 1D mask in (3). We rename the factor a to $\overline{a}$ and derive from mask (3) the new mask

$$\text{s-lift}_{decode}(a,b): \qquad \begin{pmatrix} a^2+2\overline{a}ab & & a^2+2\overline{a}ab \\ & b^2 & \\ a^2+2\overline{a}ab & & a^2+2\overline{a}ab \end{pmatrix} . \tag{9}$$

Finally, the w-lift operation is executed again. The s-lift decoding operation has (theoretically) applied the 1D mask in (3). Since the 1D mask in (3) applied by the s-lift decoding operation is the inverse of the 1D mask in (3) applied by the s-lift encoding operation, the values at the vertices • are the same as before the execution of these two s-lift operations, i. e., they vanish. These considerations define a new mask derived from mask (4), given by

$$\text{w-lift}_{decode}(a,b): \qquad \begin{pmatrix} a^2 & & a^2 \\ & b^2 & \\ a^2 & & a^2 \end{pmatrix} . \tag{10}$$

In the 2D case, the masks are as narrow as they can be.

6 Results for the 2D case

In Figure 7, we provide an example for $\sqrt{2}$ subdivision and 2D wavelets. The original surface shown in Figure 7(a) results from sampling a 2D Gaussian function at 64^2 vertices. The surface is encoded and decoded again. In Figure 7(b), we show a coarse level of detail obtained by $\sqrt{2}$-subdivision wavelets. In Figure 7(c), we show the same level of detail obtained when combining bilinear B-spline wavelets and $\sqrt{2}$ subdivision in the way described in the previous section.

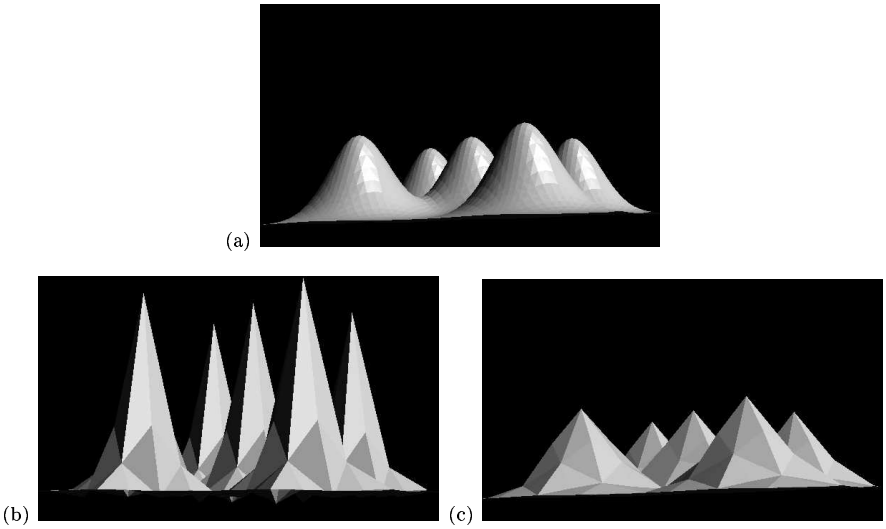


Fig. 7. (a) $\sqrt{2}$-subdivision surfaces; (b) encoded and decoded by $\sqrt{2}$-subdivision wavelets; and (c) bilinear B-spline wavelets.

In Figure 7(b), the over- and undershoots caused by the $\sqrt{2}$-subdivision wavelets can be recognized. No over- and undershoots are visible when combining $\sqrt{2}$ subdivision with linear B-spline wavelets, see Figure 7(c).

We also have developed a lifting scheme for cubic B-spline wavelets, but the masks are not as narrow as in the linear case, three instead of two lifting steps are required, see [2], and, most importantly, over- and undershoots appear again. Since linear B-spline wavelets, contrary to cubic ones, have interpolating scaling functions, interpolating refinement filters are guaranteed, see [12], i. e., no over- and undershoots can appear.

For progressive visualization, e. g. when generating images progressively by loading data from slow external memory or via Internet, the storage of values can be reorganized as shown in Figure 8. Progressive visualization starts by using the upper left block in the right picture, then adding the upper right block, and, finally, adding the lower block. Reordering ensures

that data can be read in a continuous stream without reading data multiple times.

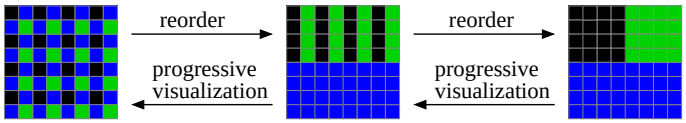


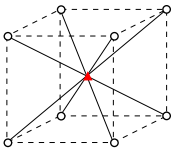
Fig. 8. Reordering data for progressive visualization.

7 A lifting scheme for $\sqrt[3]{2}$ subdivision

In this section, we generalize the ideas of Section 5 to the 3D case. Recalling the steps of a $\sqrt[3]{2}$-subdivision scheme depicted in Figure 3, after the execution of the different steps different kinds of polyhedral shapes arise, see Figure 4. Therefore, we have to distinguish between the different steps. The following description starts with the situation shown in the second picture of Figure 3 (*volume case*), proceeds with the situation shown in the third picture of Figure 3 (*face case*), and finally treats the situation shown in the fourth picture of Figure 3 (*edge case*), which is topologically equivalent to the situation shown in the first picture of Figure 3.

The volume case. To perform linear B-spline wavelet encoding in the situation shown in the second picture of Figure 3, we first execute a w-lift operation. Therefore, we apply masks being similar to masks in (6), subject to the constraint that no values are available at the vertices • and △.

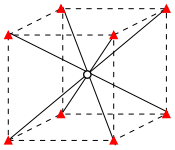
Regarding the structures of masks (6), we assume that the value at a vertex • is defined by linear interpolation of the values at the two vertices ∘ (with which the vertex • shares an edge), and that the value at a vertex △ is defined by bilinear interpolation of the values at the four vertices ∘ (with which the vertex △ shares a face). One obtains the mask w-lift$_{encode}(a, b)$, depicted as



$\circ\ a^3 + \frac{3}{2}a^2b + \frac{3}{4}ab^2$

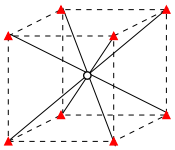
$\blacktriangle\ b^3$

The masks being analogous to the 2D and 1D masks in (6) are only "applied theoretically." However, since the values at the vertices • are assumed to be linear interpolations of the values at the vertices ∘, and since the values at the vertices △ are assumed to be bilinear interpolations of the values at the vertices ∘, the values at the vertices • and △ vanish. Therefore, the mask for the next s-lift operation, which is an analogue of the 3D mask in (5), reduces to the mask s-lift$_{encode}(a, b)$, depicted as

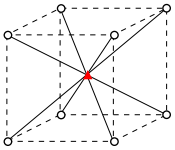


Again, the analogous versions of the 2D and 1D masks in (5) are only applied theoretically.

For the decoding step, we start with the s-lift operation, i.e., we adjust mask (5). Having (theoretically) applied the 2D and 1D masks in (5) with vanishing values at the vertices • and △, the values at the vertices △ are linear interpolations of the values at the neighbor vertices ▲, multiplied by the factor $2a$, and the values at the vertices • are bilinear interpolations of the values at the neighbor vertices ▲, multiplied by the factor $4a^2$. By renaming the factor a to $\overline{a}$, we obtain the mask s-lift$_{decode}(a, b)$, depicted as



Again, the analogous versions of the 2D and 1D masks in (5) are only applied theoretically. Since the masks (5) of this s-lift operation are the inverse masks of the masks (5) of the encoding s-lift operation, the vertices • and △ have their former values assigned again, i.e., the values vanish. Hence, the mask for the final w-lift operation, which is the mask being analogous to the 3D mask in (6), reduces to the mask



In the 3D case, the masks are as narrow as they can be.

The face case. When applying linear B-spline wavelet encoding to the situation depicted in the third picture of Figure 3, we have to make sure that we do not violate the assumptions made for the volume case. We assume that the values at the vertices △ are bilinear interpolations of the values at the neighbor vertices ∘. Thus, when the values at the vertices △ are available, their values should be computed only from the values at the vertices ∘. This insight leaves us with the 2D case, and we can apply masks (7) – (10) of Section 5.

The edge case. When applying linear B-spline wavelet encoding to the situation illustrated in the fourth picture of Figure 3, we must not violate the assumption that the values at the vertices • are linear interpolations of the values at the neighbor vertices ∘. When the values at the vertices • are available, their values should be computed only from the values at the vertices

o. This insight leaves us with the 1D case, and we can apply masks (1) and (2) of Section 4.

It is a significant advantage of our scheme that the face and edge cases cover naturally boundary faces and boundary edges of the domain.

8 Results for the 3D case

In this section, we compare the results obtained by applying a $\sqrt[3]{2}$-subdivision multiresolution scheme with and without trilinear B-spline wavelet encoding. Since we want to show how our wavelets improve image quality at a low resolution, all examples are provided at a coarse level of detail.

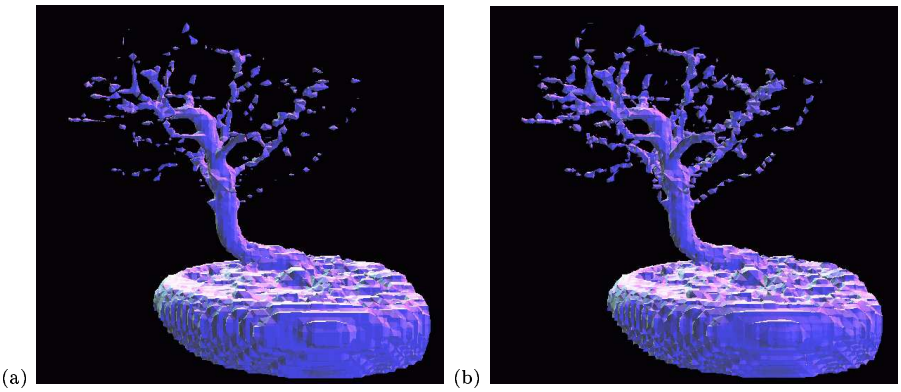


Fig. 9. Comparing $\sqrt[3]{2}$-subdivision hierarchy without (a) and with (b) trilinear B-spline wavelets. Shown is the same isosurface extracted from the level of detail with downsampling ratios 2^6. (Data set courtesy of S. Roettger, Abteilung Visualisierung und Interaktive Systeme, University of Stuttgart, Germany)

The data set used in Figure 9 is a 256^3 uniform rectilinear grid, and at every vertex one scalar value between 0 and 255 is given. The data set represents a "bonsai tree solid." It was obtained by computer tomography. For the visualization of the bonsai tree, we extracted and rendered the isosurface corresponding to the value 80, which was generated by the marching-tetrahedra algorithm described in [9]. All the polyhedral shapes in Figure 4 have a unique subdivision into tetrahedra according to the longest-edge bisection refinement. Thus, all visualization methods based on tetrahedra, including the more sophisticated isosurface-extraction methods in [6, 22], could be applied.

Figure 9(a) shows the isosurface extracted from a coarse level of detail of a $\sqrt[3]{2}$-subdivision hierarchy without using wavelets. Figure 9(b) shows the same isosurface extracted from the same level of detail, where a $\sqrt[3]{2}$-subdivision

Table 1. Root-mean-square errors for three examples at different levels of resolution without (w/o) and with (w/) trilinear B-spline wavelets.

$\mathcal{E}_{RMS}$	Figure 9		Figures 10 and 11		Figure 12		Figures 9-12
dr	w/o	w/	w/o	w/	w/o	w/	improvement
2^3	1.79%	1.59%	2.84%	2.58%	1.77%	1.57%	11.80%
2^6	3.63%	3.13%	3.84%	3.46%	4.29%	3.90%	12.32%
2^9	6.02%	5.05%	5.07%	4.61%	7.60%	6.98%	12.69%
2^{12}	8.99%	7.51%	6.84%	6.25%	10.94%	9.90%	13.22%
2^{15}	12.26%	9.64%	9.23%	8.72%	14.35%	12.71%	15.31%

hierarchy was combined with the trilinear B-spline wavelet scheme described in the previous section. The resolution is not high enough to represent the finest details, like branches and twigs, but the averaging steps of a wavelet encoding clearly leads to better approximations.

To quantify the improvement in approximation quality, we computed an approximation error for each coarser level of approximation by comparing it to the original, highest resolution level. Given the original function F discretely by sample values at locations $\mathbf{x_i}$, $\mathbf{i} \in [1, n_x][1, n_y][1, n_z]$, we used the root-mean-square error

$$\mathcal{E}_{RMS} = \sqrt{\frac{1}{n_x n_y n_z} \sum_{\mathbf{i}} \left(F(\mathbf{x_i}) - f(\mathbf{x_i})\right)^2} \ ,$$

where $f(\mathbf{x_i})$ denotes the approximated function value obtained by trilinear interpolation applied to a "cell" in the coarser level of resolution: If f is defined at corner locations $\mathbf{y_j} = (y_{\mathbf{j},x}, y_{\mathbf{j},y}, y_{\mathbf{j},z})$, and if $\mathbf{x_i}$ is inside the interval $[y_{\mathbf{j},x}, y_{\mathbf{j}+\mathbf{e}_1,x})[y_{\mathbf{j},y}, y_{\mathbf{j}+\mathbf{e}_2,y})[y_{\mathbf{j},z}, y_{\mathbf{j}+\mathbf{e}_3,z})$, the approximated function value $f(\mathbf{x_i})$ results from trilinear interpolation of the eight corner values $f(\mathbf{y_j}), \ldots, f(\mathbf{y_{j+1}})$.

Table 1 lists the root-mean-square errors of the shown examples at various levels of resolution. We scaled the root-mean-square error to the interval $[0, 1]$. The "downsampling ratio" (dr) is defined as the original number of vertices divided by the number of vertices at the used coarser resolution. For all examples and all resolutions, we obtained smaller root-mean-square errors when using trilinear B-spline wavelets. The last row of the table quantifies the "improvement" by listing the average error reduction for each downsampling ratio. We recognize that the improvement increases for coarser resolutions.

Figure 10 shows a biomedical example. The data set represents a human brain. It is given as 753 slices, and each slice has a resolution of 1050×970 points, where 24-bit RGB-color information is stored. The original data set was preprocessed with a segmentation algorithm described in [30] to eliminate noise. We applied the wavelet scheme to each color channel independently and, after conversion, used the value V of the HSV color model for isosurface extraction.

Since the data was too large to be stored in main memory, we used out-of-core techniques. Due to the narrow masks of our lifting scheme, at most three slices were used simultaneously.

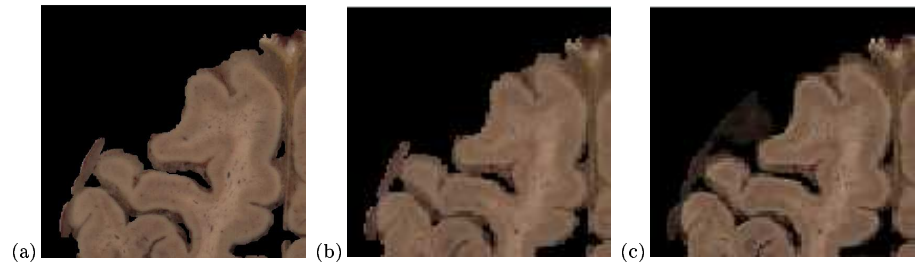


Fig. 10. (a) Slice through 3D brain data set at full resolution; (b) slice at level of detail with downsampling ratio 2^6 without and (c) with B-spline wavelets on a $\sqrt[3]{2}$-subdivision scheme. (Data set courtesy of A. Toga, Ahmanson-Lovelace Brain Mapping Center, University of California, Los Angeles)

For Figure 10, we used an interactive progressive slicing visualization tool, see [23], to generate an arbitrary cutting plane through the brain data set. Figure 10(a) shows the slice at the highest resolution, Figure 10(b) after downsampling with $\sqrt[3]{2}$ subdivision (downsampling ratio 2^6), and Figure 10(c) after downsampling with $\sqrt[3]{2}$ subdivision (downsampling ratio 2^6) and trilinear B-spline wavelets.

Compared to Figure 10(b), the contours of the brain in Figure 10(c) are much smoother. Moreover, the slice in Figure 10(c) does not only contain information of the slice in Figure 10(a) but also of the full-resolution slices next to it. Without the wavelet averaging, some detailed information of the neighbored slices might get lost.

Figure 11 shows an isosurface for the value 78 extracted from the same data set at the level of detail with downsampling ratio 2^9. For Figure 11(a), we used a $\sqrt[3]{2}$-subdivision hierarchy without using wavelets, and, for Figure 11(b), we combined the $\sqrt[3]{2}$-subdivision hierarchy with trilinear B-spline wavelets. Figure 11(b) exhibits much more detail information than Figure 11(a).

In Figure 12, we applied our techniques to numerically simulated hydrodynamics data. The data set is the result of a 3D simulation of the Richtmyer-Meshkov instability and turbulent mixing in a shock tube experiment, see [19]. For each vertex of a 1024^3 structured-rectilinear grid (one time step considered only), an entropy value between 0 and 255 is stored. The figure shows the isosurface corresponding to the value 225 extracted from three different levels of resolution of one time step. Again, we compared the results of the $\sqrt[3]{2}$-subdivision hierarchy without (left column) and with (right column) trilinear B-spline wavelets, partially computed out-of-core.

Considering the example shown in Figure 12, when using the wavelet approach low-resolution visualizations suffice to understand where the turbulent mixing takes place. For example, Figure 12(c) shows clearly the big "bubble" rising in the middle of the data set. The bubble can hardly be seen in Figure 12(a).

9 Conclusions

We have introduced $\sqrt[n]{2}$ subdivision combined with n-variate B-spline wavelets for n-dimensional multiresolution data representation. Visualization of biomedical imaging data and numerically simulated hydrodynamics data, for example, require efficient methods of isosurface extraction. For this purpose, a 3D multiresolution framework is desirable. We first have established a bivariate B-spline wavelet scheme for $\sqrt{2}$ subdivision and have generalized it to a trivariate B-spline wavelet scheme for $\sqrt[3]{2}$ subdivision. The provided examples document the value of our approach for surface and volume modeling and visualization.

By using $\sqrt[n]{2}$ subdivision, instead of using quad- or octrees, a multiresolution hierarchy can be generated that provides much more levels of detail, since, in each subdivision step, the number of vertices is only doubled instead of multiplied by a factor of four or eight, respectively. In the context of view-dependent and adaptive refinement and visualization, this characteristic supports a higher level of adaptivity. Furthermore, $\sqrt[n]{2}$ subdivision does not only work for structured-rectilinear grids, but also for more general structured-curvilinear grids, and even for arbitrary grids, i. e., grids with extraordinary vertices.

By integrating a wavelet scheme into the subdivision approach, we obtain, in general, much better approximations on each level of detail. We have chosen n-variate B-spline wavelets and have developed lifting schemes for $n = 2$ and $n = 3$, which use narrow masks. These narrow masks allow us to utilize the wavelet scheme for view-dependent, adaptive multiresolution visualization of large-scale data.

The wavelet encoding only reorganizes data and does not require additional memory. The $\sqrt[n]{2}$-subdivision scheme also does not require us to store additional connectivity information. Thus, our approach, as a whole, requires no additional storage.

Since the masks of our lifting scheme are of constant size and the number of iterations for our lifting scheme is constant, our algorithms run in linear time with respect to the number of original data. Since the masks are narrow and only two iterations are needed, the run-time constants are small. Considering the examples shown, we conclude that our approach provides a valuable tool for the interactive exploration of volumetric data at multiple level of resolution.

Acknowledgments

This work was supported by the National Science Foundation under contract ACI 9624034 (CAREER Award), 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; the Army Research Office under contract ARO 36598-MA-RIP; 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 Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis, and the members of the Data Science Group at the Center for Applied Scientific Computing (CASC) at the Lawrence Livermore National Laboratory. We especially thank Peer-Timo Bremer for supplying us with an implementation of the $\sqrt{2}$-subdivision wavelets.

References

1. Martin Bertram. *Multiresolution Modeling for Scientific Visualization.* PhD thesis, Department of Computer Science, University of California, Davis, California, 2000.
2. Martin Bertram, Mark A. Duchaineau, Bernd Hamann, and Kenneth I. Joy. Bicubic subdivision-surface wavelets for large-scale isosurface representation and visualization. In Thomas Ertl, Bernd Hamann, and Abitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2000*, pages 389–396. IEEE, IEEE Computer Society Press, 2000.
3. Martin Bertram, Daniel E. Laney, Mark A. Duchaineau, Charles D. Hansen, Bernd Hamann, and Kenneth I. Joy. Wavelet representation of contour sets. In Thomas Ertl, Kenneth I. Joy, and Amitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2001*, pages 303–310. IEEE, IEEE Computer Society Press, 2001.
4. Paolo Cignoni, Claudio Montani, Enrico Puppo, and Roberto Scopigno. Multiresolution modeling and visualization of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 3(4):352–369, 1997.
5. Albert Cohen and Ingrid Daubechies. Nonseparable bidimensional wavelet bases. *Rev. Mat. Iberoamericana*, 9(1):51–137, 1993.
6. Thomas Gerstner and Renato Pajarola. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In Thomas Ertl, Bernd Hamann, and Abitabh Varshney, editors, *Proceedings of IEEE Conference on Visualization 2000*, pages 259–266. IEEE, IEEE Computer Society Press, 2000.
7. Roberto Grosso and Günther Greiner. Hierarchical meshes for volume data. In *Proceedings of CGI '98, Hanover, Germany*, 1998.
8. Roberto Grosso, Christoph Lürig, and Thomas Ertl. The multilevel finite element method for adaptive mesh optimization and visualization of volume data. In R. Yagel and H. Hagen, editors, *Proceedings of IEEE Conference on Visualization 1997*, pages 135–142. IEEE, IEEE Computer Society Press, 1997.
9. André Guéziec and Robert Hummel. Exploiting triangulated surface extraction using tetrahedral decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 1(4):328–342, 1995.
10. Leif Kobbelt. $\sqrt{3}$-subdivision. In Kurt Akeley, editor, *Proceedings of SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 103–112. ACM, ACM Press / ACM SIGGRAPH, 2000.
11. Leif Kobbelt. Multiresolution techniques. In Farin, Hoschek, and Kim, editors, *Handbook of Computer Aided Geometric Design.* Elsevier Science Publishing, Amsterdam, The Netherlands, 2002.
12. Jelena Kovačević and Wim Sweldens. Wavelet families of increasing order in arbitrary dimensions. *IEEE Transactions on Image Processing*, 9(3):480–496, 1999.
13. 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.
14. 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.

15. 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.
16. Ron A. MacCracken and Kenneth I. Joy. Free-form deformations with lattices of arbitrary topology. In Holly Rushmeier, editor, *Proceedings of SIGGRAPH 1996*, Computer Graphics Proceedings, Annual Conference Series, pages 181–188. ACM, ACM Press / ACM SIGGRAPH, 1996.
17. Donald Maegher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19:129–147, 1982.
18. Joseph M. Maubach. Local bisection refinement for n-simplicial grids generated by reflection. *SIAM J. Scientific Computing*, 16:210–227, 1995.
19. Arthur A. Mirin, Ron H. Cohen, Bruce C. Curtis, William P. Dannevik, Andris M. Dimits, Mark A. Duchaineau, D. E. Eliason, Daniel R. Schikore, S. E. Anderson, D. H. Porter, and Paul R. Woodward. Very high resolution simulation of compressible turbulence on the ibm-sp system. In Sally Howe, editor, *Proceedings of Supercomputing '99*. IEEE, IEEE Computer Society Press, 1999.
20. Mario Ohlberger and Martin Rumpf. Hierarchical and adaptive visualization on nested grids. *Computing*, 59:365–385, 1997.
21. Valerio Pascucci. Slow growing subdivision (sgs) in any dimension: towards removing the curse of dimensionality. In *to appear in: Proceedings of Eurographics 2002*. COMPUTER GRAPHICS Forum, 2002.
22. Valerio Pascucci and Chandrajit Bajaj. Time critical adaptive refinement and smoothing. In Roger Crawfis and Daniel Cohen-Or, editors, *Proceedings of the ACM/IEEE Volume Visualization and Graphics Symposium 2000, Salt Lake City, Utah*, pages 33–42. ACM/IEEE, 2000.
23. Valerio Pascucci and Randall J. Frank. Global static indexing for real-time exploration of very large regular grids. In *Supercomputing 2001*. ACM, ACM Press, 2001.
24. 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.
25. Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-spline Techniques*. Springer-Verlag, Heidelberg, Germany, 2002.
26. Sherman D. Riemenschneider and Zuowei Shen. Wavelets and pre-wavelets in low dimensions. *Journal Approximation Theory*, 71:18–38, 1992.
27. 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.
28. Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, Brian A. Barsky (series editor), Morgan Kaufmann Publishers, San Francisco, U.S.A., 1996.
29. Wim Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet constructions. In Andrew F. Laine and Michael Unser, editors, *Wavelet Applications in Signal and Image Processing III*, pages 68–79. Proceedings of SPIE 2569, 1995.
30. Ikuko Takanashi, Eric Lum, Kwan-Liu Ma, Jörg Meyer, Bernd Hamann, and Arthur J. Olson. Segmentation and 3d visualization of high-resolution human brain cryosections. In Robert F. Erbacher, Philip C. Chen, Matti Gröhn, Jonathan C. Roberts, and Craig M. Wittenbrink, editors, *Proceedings of SPIE Visualization and Data Analysis 2002*, pages 55–61. Proceedings of SPIE 4665, 2002.
31. Geert Uytterhoeven. *Wavelets: Software and Applications*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 1999.
32. Luiz Velho and Denis Zorin. 4-8 subdivision. *Computer Aided Geometric Design*, 18(5):397–427, 2001.
33. Joe Warren and Henrik Weimer. *Subdivision Methods for Geometric Design*. Morgan Kaufmann Publishers, San Francisco, U.S.A., 2002.
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. 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.

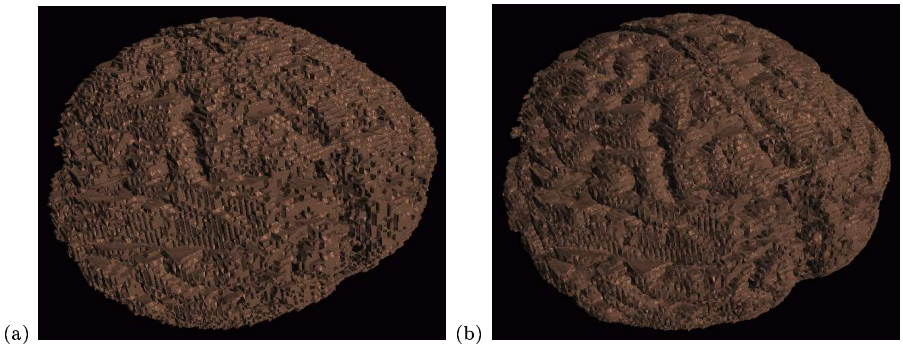


Fig. 11. Hierarchical visualization of brain data set, (a) based on $\sqrt[3]{2}$-subdivision without and (b) with B-spline wavelets. (Data set courtesy of A. Toga, Ahmanson-Lovelace Brain Mapping Center, University of California, Los Angeles)

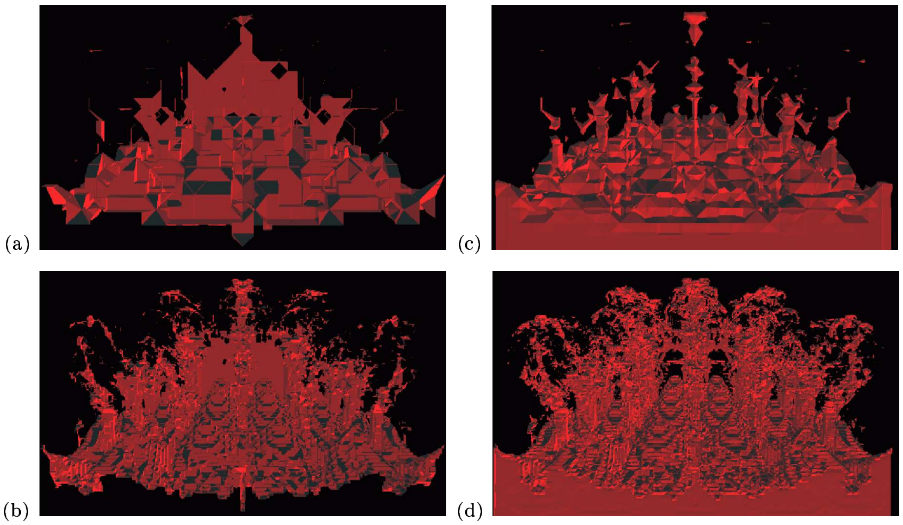


Fig. 12. Entropy in a 3D simulation of Richtmyer-Meshkov instability, visualized by isosurface extraction from a $\sqrt[3]{2}$-subdivision hierarchy without (left column) and with (right column) B-spline wavelets (downsampling ratios 2^{15} and 2^9).