# MATERIAL INTERFACE RECONSTRUCTION

Kathleen S. Bonnell[1]
Mark A. Duchaineau[2]
Daniel R. Schikore[3]
Bernd Hamann[4]
Kenneth I. Joy[5]

## Abstract

This paper presents an algorithm for material interface reconstruction for data sets where fractional material information is given as a percentage for each element of the underlying grid. The reconstruction problem is transformed to a problem that analyzes a dual grid, where each vertex in the dual grid has an associated barycentric coordinate tuple that represents the fraction of each material present. Material boundaries are constructed by analyzing the barycentric coordinate tuples of a tetrahedron in material space and calculating intersections with Voronoi cells that represent the regions where one material dominates. These intersections are used to calculate intersections in the Euclidean coordinates of the tetrahedron. By triangulating these intersection points one creates the material boundary. The algorithm can treat data sets containing any number of materials. The algorithm can also create non-manifold boundary surfaces if necessary. By clipping the generated material boundaries against the original cells, one can examine the error in the algorithm. Error analysis shows that the algorithm preserves volume fractions within an error range of 0.5% per material.

Keywords: Eulerian ¤ow, material boundaries, material interfaces, £nite elements, barycentric coordinates, volume fraction, isosurface extraction.

[1]P.O. Box 808, L-312, Lawrence Livermore National Laboratory, Livermore, CA 94551, USA; e-mail: ksbonnell@ucdavis.edu

[2]Center for Advanced Scienti£c Computing (CASC), Lawrence Livermore National Laboratory, Livermore, CA 94551, USA; e-mail: duchaine@llnl.gov

[3]Computational Engineering International, Morrisville, NC 27560, USA; e-mail: schikore@ceintl.com

[4]Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California, Davis, CA 95616-8562, USA; e-mail: hamann@cs.ucdavis.edu

[5]Corresponding Author, Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California, Davis, CA 95616-8562, USA; e-mail: joy@cs.ucdavis.edu

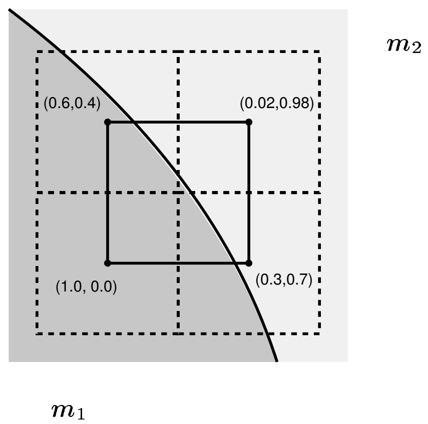


FIGURE 1: Original grid and dual grid. The original grid (dashed lines) is replaced by a dual grid (solid lines), obtained by connecting the centers of the original grid elements. Barycentric coordinates tuples are associated with each vertex of the dual grid.

## 1. INTRODUCTION

In many applications it is necessary to reconstruct or track the boundary surfaces (or "interfaces") between multiple materials that commonly result from £nite-element simulations. Multi-¤uid Eulerian hydrodynamics calculations, for example, require geometric approximations of ¤uid interfaces to form the equations of motion to advance interfaces over time. Typically, the grid cells (£nite elements) contain fractional information for each of the materials. Each cell $C$ of a grid $\mathcal{S}$ has an associated tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$ that represents the portions of each of $m$ materials in the cell, *i.e.*, $\alpha_i$ represents the fractional part of material $i$. It is assumed that $\alpha_1+\alpha_2+\cdots+\alpha_m = 1$ and $\alpha_i \geq 0$. Given the fractions for each cell, we wish to £nd a crack-free piecewise-de£ned (potentially non-manifold) surface approximating the boundary surfaces between the various materials.

To solve this problem, we consider the dual grid constructed from the original grid, as shown in Figure 1. In the dual grid, each cell is represented by a point (typically the center of the cell), and each point has an associated tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$, where $m$ is the number of materials present in the data set. Thus, the boundary surface reconstruction problem reduces to de£ning the material interfaces (boundaries) for a (dual) grid where each vertex has an associated barycentric coordinate tuple representing the fractional parts of each material present at the vertex. We split the dual cells into simplices creating an unstructured grid where each point of an simplex has an associated barycentric coordinate.

We assume that the data set is given as a regular grid of three-dimensional hexahedral cells. In this case, the dual grid is also hexahedral and regular. Each hexahedral cell of the dual grid is split into six tetrahedra, see Nielson [1]. Each vertex of each tetrahedron is associated with a barycentric coordinate tuple.

Given a data set containing $m$ materials, each vertex of a simplex generated from the dual grid can be written and understood as a set of $3 + m$ coordinates, of which three represent the position of the vertex in Euclidean space and $m$ represent the barycentric coordinate tuple associated with the vertex. To £nd the intersection points with material boundaries, we embed the simplex in $m$-space, using only the $m$ coordinates de£ning the fractional material information for each vertex. In $m$-space (material space), we calculate intersections of the embedded tetrahedron with the edges of the Voronoi diagram [2] of the $m$-simplex. The Voronoi cells represent regions where one material "dominates" the other materials locally. In $m$ space, the intersections generate barycentric coordinates that can be used to obtain a boundary approximation in the original Euclidean space.

Section 2 describes previous work dealing with reconstruction of material boundary surfaces. Section 3 describes the algorithms for material interface reconstruction. Section 3.1 describes the two-material case, which can be viewed as a simple extension of a isosurface extraction technique. [3, 4, 5]. Section 3.2 describes the three-material case. Here, intersections are calculated in material space and mapped back to determine intersections in physical space. The general $m$-material case is described in Section 3.3. In this case, intersections are calculated in a barycentric $m$-simplex and mapped back to Euclidean coordinates of the simplices of the data set. Section 5 presents results for various data sets, and Section 6 provides an analysis of the error.

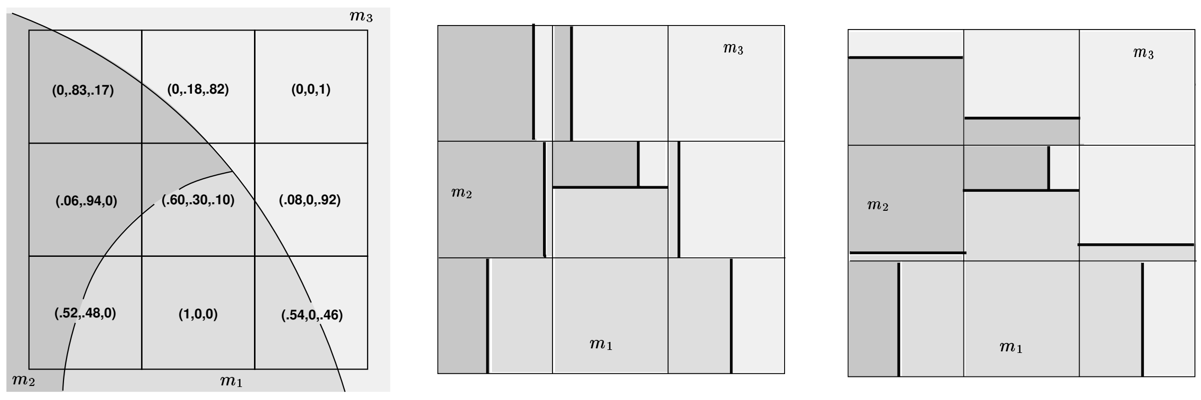
## 2. RELATED WORK

Most research in material interface reconstruction has been conducted in computational ¤uid dynamics (CFD) and hydrodynamics, where researchers are concerned with the movement of material boundaries during a simulation. The *Simple Line Interface Calculation* (SLIC) algorithm by Noh and Woodward [6] is one of the earliest algorithms, describing a method for geometric approximation of ¤uid interfaces. Their algorithm is used in conjunction with hydrodynamics simulations to track the advection of ¤uids. It produces an interface consisting of line segments, constructed parallel or perpendicular to a coordinate axis. Multi-¤uid cells can be handled by grouping ¤uids together, calculating the interface between the groups, subdividing the groups, and iterating this process. Since this algorithm only uses line segments that are parallel to the coordinate axes, the resulting interfaces are generally discontinuous.

In determining the direction of the line segment, cells to the left and right (in the appropriate coordinate direction) of the current cell are considered, and classi£ed according to the ¤uid index. The ¤uid index indicates the presence (1) or absence (0) of a material. Mixed-¤uid cells have multiple ¤uid indices, one per material. Fluids with the same ¤uid index are grouped together so that only two types may be treated at one time.

Consider a two-¤uid 2D cell consisting of materials $A$ (30%) and $B$ (70%), and an x-direction pass of the algorithm, if the left neighbor contains only material $A$ and the right neighbor contains only material $B$, the algorithm will generate an interface in the mixed ¤uid cell approximated by a vertical line dividing the cell into two regions, 30% $A$ on the left and 70% $B$ on the right. If that same cell has both left and right neighbors consisting entirely of material $A$, then the interface in the mixed-material cell would consist of two vertical lines dividing the cell into three parts: 15% material $A$ on the left, 70% material $B$ in the center, and 15% material $A$ on the right.

Consider a three-¤uid cell containing 20% $A$, 45% $B$, and 35% $C$, again with an x-direction pass. If the left neighbor consists entirely of material $A$ and the right neighbor consists entirely of material $B$, then the interface would be represented by two vertical lines, dividing the cell into three zones, with material $A$ on the left, $C$ in



(a) (b) (c)

FIGURE 2: Example data set (a). The speci£c volume fractions are listed for each cell. The approximations generated by the SLIC algorithm are shown in (b) and (c), using an x-pass and y-pass, respectively.

the middle, and $B$ on the right. If the left neighbor contains both $A$ and $B$ and the right neighbor contains $A$ and $C$, then a horizontal line segment is used to £rst construct the zone with material $A$, then the remaining portion of the cell is divided by a vertical line with $B$ on the left and $C$ on the right. These simple rules can be used to generate material interfaces for two-dimensional rectilinear grids. Figure 2 shows an approximation generated by the SLIC algorithm to a sample data set, with both an x-coordinate and y-coordinate pass. Although the interface is discontinuous, the volume fractions are preserved for each cell.

The algorithm of Youngs [7] also operates on two-dimensional grids and uses line segments to approximate interfaces. In this algorithm, the line segments are not necessarily perpendicular or parallel to a coordinate axis. Instead, the neighbor cells of a cell $C$ are used to determine the slope of a line segment approximating an interface in $C$. The exact location of the line segment is adjusted to preserve volume fractions. Multiple materials are treated by grouping materials and determining interfaces on a two-material basis. Again, the resulting interfaces are generally discontinuous.

Since this algorithm treats only two materials (or groups of materials) at a time, one of the materials is used to determine the slope of the interface line. This is done by using neighboring fractions of this material, and the Pythagorean theorem. Figure 3 demonstrates the neighbors of a cell containing materials $A$ and $B$. The cell is treated as a unit-square for this calculation. The slope is de£ned as $\sqrt{(\delta_A - \gamma_A)^2 + (\beta_A - \alpha_A)^2}$, where $\alpha_A$, $\beta_A$, $\gamma_A$, and $\delta_A$ are the fractions of material $A$ present in the neighbor cells[1] Once the slope is determined, the line segment is positioned in the cell such that the volume fractions are preserved. For more than two materials, the user determines in which order interfaces are calculated. Different interfaces will result depending on the chosen ordering/grouping. Figure 4 demonstrates the results of applying Youngs' algorithm.

The algorithm of Gueyf£er [8] is similar to that of Youngs in that it requires an estimate of the normal vector to

[1]We use neighboring "corner" cells to determine the slope if this equation fails. It is possible that one must use the "negative" square root to determine the correct slope.

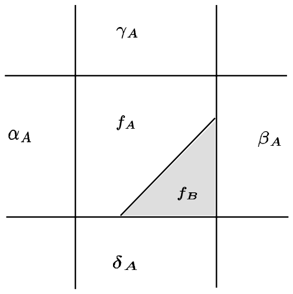
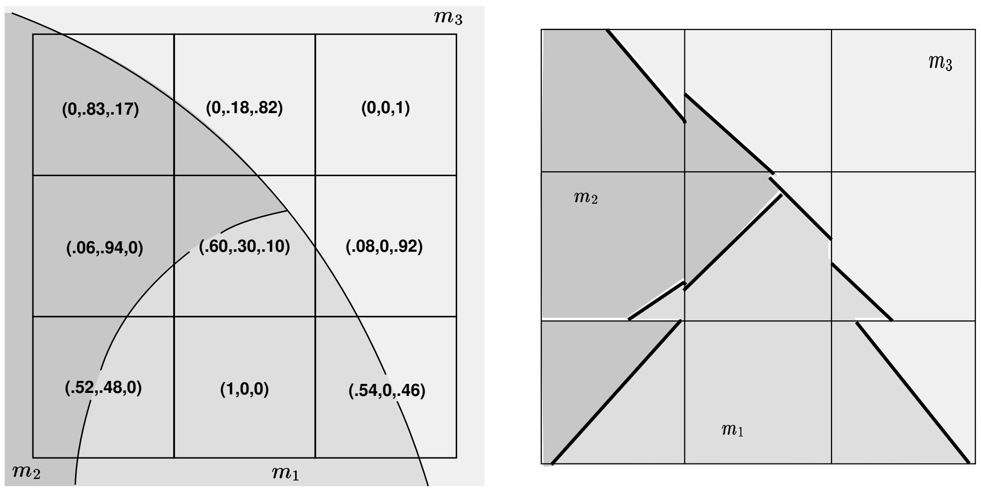


FIGURE 3: Two-material cell with material fractions $f_A$ and $f_B$. Neighbor cells have material fractions $\alpha_A$, $\beta_A$, $\gamma_A$, and $\delta_A$ for material $A$. The slope of the line is determined by the values of $\alpha_A$, $\beta_A$, $\gamma_A$, and $\delta_A$. The position of the line is de£ned by $f_A$ and $f_B$



(a) (b)

FIGURE 4: Result of Youngs' algorithm applied to the data shown in (a). The material interface representation (b) is discontinuous.

the interface in order to reconstruct the interface. Geuyff£er's method utilizes £nite-differencing or least-squares methods to determine this normal, depending upon the order of accuracy (£rst- or second-order) desired. In 2D, a line segment representing the boundary surface is constructed perpendicular to the interface normal. The line segment is positioned in the cell such that it divides the cell into appropriately proportioned areas. In the 3D case, a cutting plane is computed whose normal is the interface normal. Again, the cutting plane is positioned in the cell so that volume fractions are preserved. It is unclear how this algorithm would handle multiple materials.

Pilliod and Puckett [9] compare various volume-of-¤uid interface reconstruction algorithms, including SLIC, noting differences in the surfaces reconstructed and demonstrating £rst-order or second-order accuracy. Their goal is to develop an algorithm that accurately reproduces a linear material interface, allowing discontinuous interfaces if the material boundary is not linear.

Nielson and Franke [10] have presented a method for calculating a separating surface in an unstructured grid where each vertex of the grid is associated with one of several possible classes. Their method generalizes the marching-cubes (or marching-tetrahedra) algorithm, but instead of using a strict binary classi£cation of vertices, it allows any number of classes. Edges in tetrahedral grids whose endpoints have different classi£cations are intersected by the separating surface. Similarly, the faces of a tetrahedron whose three vertices are classi£ed differently, are assumed to be intersected by the surface in the middle of the face. When all four vertices of a tetrahedron have different classi£cations, the boundary surface intersects in the interior of the tetrahedron. The resulting "mid-edge," "mid-face," and "mid-tetrahedron" intersections are triangulated to linearly approximate the surface.

Using the dual-grid representation, Figure 5 shows how Nielson and Franke's algorithm might be applied to the example data set. As this algorithm requires classi£cation of vertices, the greater $\alpha_i$ value in the volume fraction tuple for each cell was chosen as the classi£er. The dual grid has also been triangulated so that the algorithm can be applied. Mid-edge intersections are made half-way between two endpoints that have different classi£cations. Mid-face intersections are assumed when all three vertices of the triangle have different classi£cations.

This paper is an expansion of our work discussed in [11], and generalizes the above schemes. It utilizes a dual-grid approach, where each vertex of the grid has an associated barycentric coordinate tuple. This allows the generation of material boundaries directly from intersections calculated in "material space." The algorithm handles multiple materials and can reconstruct layers and non-manifold interfaces. The algorithm does not rely on application-speci£c knowledge of hydrodynamics or other simulation codes, but solves the problem from a purely mathematical viewpoint.

## 3. MATERIAL INTERFACE CONSTRUCTION

We assume that we are dealing with a grid containing $m$ materials, where the grid cells contain fractional information for each of the materials. To generate the boundaries of material regions, we consider the dual grid constructed from the original grid. In the dual grid, each cell is represented by a point and each point has an

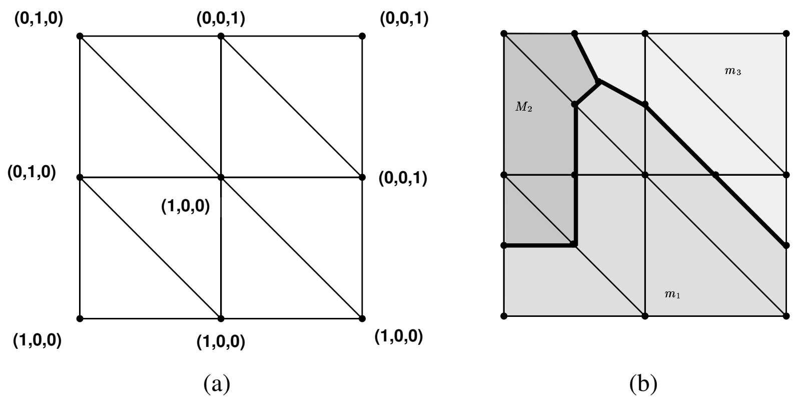


FIGURE 5: Applying the Nielson-Franke algorithm. The test data set (a) represents only "classes" of data, The boundary reconstruction result is shown in (b)

associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$. We split each cell of the dual grid into tetrahedra (triangles in the two-dimensional case), creating an unstructured grid $\mathcal{S}$ where the vertices of each tetrahedron (triangle) have the form $(\mathbf{p}, \alpha)$, where $\mathbf{p}$ denotes the coordinate of the vertex in Euclidean space and $\alpha$ is the associated $m$-material barycentric coordinate tuple. Thus, the boundary surface reconstruction problem reduces to generating the material interfaces for an unstructured simplicial grid, where the vertices of each tetrahedron (triangle) have associated barycentric coordinate tuples.

In most cases, the associated barycentric coordinate tuples of the vertices of a tetrahedron (triangle) $T$ will indicate that only one material present, *i.e.*, for some $k$, $\alpha_k = 1$ and $\alpha_i = 0$ for all $i \neq k$ for each barycentric coordinate associated with the vertices of $T$. We call this case the *one-material* case. In this case, we assume that the entire tetrahedron is £lled with only one material and no boundary is present.

## 3.1. The Two-material Case

Consider an unstructured grid $\mathcal{S}$ containing $m$ materials, *i.e.*, each vertex of $\mathcal{S}$ has the form $(\mathbf{p}, \alpha)$. A tetrahedron (triangle) $T$ of $\mathcal{S}$ contains *two materials* if there are two indices $i_1$ and $i_2$, such that the associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2 ..., \alpha_m)$ of each vertex of $T$ has the property that $\alpha_i = 0$ for $i \neq i_1, i_2$.

If $T$ contains two materials, then, without loss of generality, we can assume that each vertex of $T$ has an associated barycentric coordinate tuple that can be represented by a two-tuple $\alpha = (\alpha_1, \alpha_2)$, where $\alpha_1 + \alpha_2 = 1$. Given two vertices $(\mathbf{p}_1, \alpha^{(1)})$ and $(\mathbf{p}_2, \alpha^{(2)})$, $\alpha^{(1)}$ and $\alpha^{(2)}$ lie on the line connecting the points $(1, 0)$, and $(0, 1)$ in material space, as is shown in Figure 6. We assume that the set of points where $\alpha_1 = \alpha_2 = \frac{1}{2}$ corresponds to the boundary between the two materials in material space.[2]. There are two cases one must consider:

1. The line segment $\overline{\alpha^{(1)}\alpha^{(2)}}$ does not contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the line does not

[2]This choice is clearly a heuristic, but it is the most reasonable one among all the points of the line.

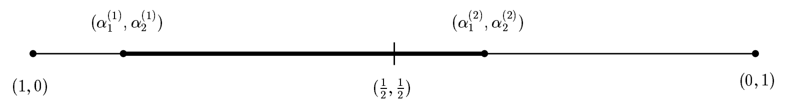

$(\alpha_1^{(1)}, \alpha_2^{(1)})$
$(\alpha_1^{(2)}, \alpha_2^{(2)})$
$(1, 0)$
$(\frac{1}{2}, \frac{1}{2})$
$(0, 1)$

FIGURE 6: Mapping the barycentric coordinate tuples in the two-material case. The mapping results in a line segment lying on the line having endpoints $(1, 0)$, and $(0, 1)$. We assume that the material boundary corresponds to the point $\left(\frac{1}{2}, \frac{1}{2}\right)$.

intersect the material boundary.

2. The line segment $\overline{\alpha^{(1)}\alpha^{(2)}}$ contains the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the line intersects the material boundary and use linear interpolation to calculate a value $r$ such that

$$\left(\frac{1}{2}, \frac{1}{2}\right) = (1-r)\alpha^{(1)} + r\alpha^{(2)}.$$

We can then calculate a point $\mathbf{p}$

$$\mathbf{p} = (1-r)\mathbf{p}_1 + r\mathbf{p}_2$$

in Euclidean space that lies on the line $\overline{\mathbf{p}_1\mathbf{p}_2}$ that crosses the material boundary.

If $T$ is a two-material triangle in a two-dimensional unstructured grid with vertices $(\mathbf{p}_1, \alpha^{(1)})$, $(\mathbf{p}_2, \alpha^{(2)})$, and $(\mathbf{p}_3, \alpha^{(3)})$, the barycentric coordinate tuples of each edge of $T$ de£ne a line segment in material space. We test these three line segments to determine if they contain the point $(\frac{1}{2}, \frac{1}{2})$, the material boundary. Two cases arise:

1. No material-space line segment contains $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the material boundary does not intersect the triangle.

2. Exactly two of the material-space line segments contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we calculate points on the line segments where the boundary interface exists (using the approach discussed above) and connect the two points with a line.

If $T$ is a two-material tetrahedron contained in a three-dimensional unstructured grid, the barycentric coordinate tuples of each of the six edges of $T$ de£ne line segments in material space. We test these line segments to determine if they contain the point $(\frac{1}{2}, \frac{1}{2})$, the material boundary. Three cases arise:

1. No material-space line segment contains $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the material boundary does not intersect the triangle.

2. Exactly three of the material-space line segments contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we calculate the intersections in material space and use the interpolated values to calculate Euclidean-space points on the edges of $T$. These points are connected to form a triangle. This case is illustrated in Figure 7a.

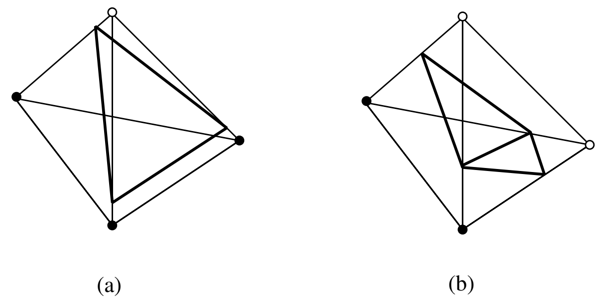


FIGURE 7: Calculating a material interface in the two-material case. A material interface can intersect a tetrahedron in two ways. In case (a), one triangle is produced, in case (b), two triangles are produced.

3. Exactly four of the material-space line segments contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we calculate intersections in material space, and use the interpolated values to calculate Euclidean-space points on the edges of $T$. Connecting these points forms a quadrilateral. This quadrilateral is split into two triangles. This is shown in Figure 7b.

This method uses the same technique underlying the marching-tetrahedra algorithm [5]. Therefore, £nding the boundary in the two-material case is equivalent to an isosurface calculation.

## 3.2. The Three-Material Case

Consider an unstructured simplicial grid $\mathcal{S}$ containing $m$ materials, *i.e.*, each vertex of $\mathcal{S}$ has the form $(\mathbf{p}, \alpha)$, where $\mathbf{p}$ is the Euclidean coordinate of the vertex and $\alpha$ is the associated barycentric coordinate tuple. A tetrahedron (triangle) $T$ of $\mathcal{S}$ contains *three materials* if there are three indices $i_1$, $i_2$ and $i_3$, such that the associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$ of each vertex of $T$ has the property that $\alpha_i = 0$ for $i \neq i_1, i_2, i_3$.

If $T$ contains three materials, it is suf£cient to assume that each vertex has an associated barycentric coordinate 3-tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 + \alpha_2 + \alpha_3 = 1$. For each vertex, the tuple $(\alpha_1, \alpha_2, \alpha_3)$ lies inside or on the boundary of the equilateral triangle with vertices $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ in material space, as shown in Figure 8. The triangle is partitioned into three regions, de£ned by the *Voronoi cells* $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$. The Voronoi cells $\mathcal{V}_j$ are bounded by the edges of the triangle and the three line segments $l_{12}$, $l_{13}$, and $l_{23}$, where (i) $\alpha_1 = \alpha_2$ and $\alpha_3 \leq \frac{1}{3}$, (ii) $\alpha_1 = \alpha_3$ and $\alpha_2 \leq \frac{1}{3}$, or (iii) $\alpha_2 = \alpha_3$ and $\alpha_1 \leq \frac{1}{3}$, respectively.

If $T$ is a triangle in an unstructured two-dimensional grid, the associated barycentric coordinate tuples of the vertices of $T$ form a triangle $T_\alpha$ in material space. The intersections of the edges of $T_\alpha$ with the edges of the Voronoi cells are used to de£ne material interfaces in $T_\alpha$. These intersections are used to form intersections on the Euclidean-space coordinates of $T$. There are three cases to consider:

- The triangle $T_\alpha$ does not intersect $l_{12}$, $l_{13}$, or $l_{23}$. In this case, it is assumed that no material boundary exists in $T$.

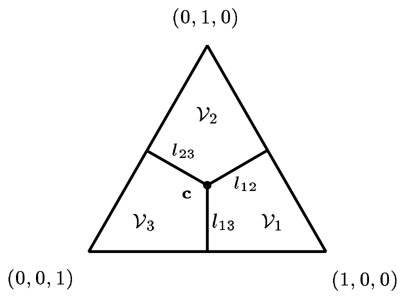


FIGURE 8: The partitioned triangle for the three-material case. The point $\mathbf{c}$ is the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the center of the triangle. The line segments $l_{12}$, $l_{13}$, and $l_{23}$ bound the Voronoi cells $\mathcal{V}_j$ in the interior of the triangle.

- The triangle $T_\alpha$ intersects at least one of the line segments $l_{12}$, $l_{13}$, or $l_{23}$ and the center $\mathbf{c}$ of the barycentric triangle does not lie inside $T_\alpha$. In this case, intersections of $T_\alpha$ with $l_{12}$, $l_{13}$, and $l_{23}$ are calculated. (The triangle $T_\alpha$ may intersect at most two of these lines.) These intersections are used to de£ne intersections in the coordinates of $T$, forming the material boundary in $T$. Figures 9a, 9d, and 9e illustrate these cases.

- The point $\mathbf{c}$ lies inside $T_\alpha$. In this case, three edge intersections are calculated for $T_\alpha$, one with each line segment $l_{ij}$. These intersections, and the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ in the interior of $T_\alpha$, are used to de£ne edge intersections for the edges of $T$ and a "face point" in the interior of $T$. Material boundary line segments are de£ned as the three lines connecting the edge intersections and the interior face point. Figures 9b and 9c illustrate the possible cases.

If $T$ is a tetrahedron in an unstructured three-dimensional grid, the barycentric coordinate tuples associated with the vertices of $T$ are used to map the tetrahedron to a tetrahedron $T_\alpha$ in material space. Intersections are calculated separately for each face of $T_\alpha$, which are then used to create the Euclidean-space coordinates of the material boundary in $T$. There are three cases to consider:

- No edge of the tetrahedron $T_\alpha$ intersects the line segments $l_{12}$, $l_{13}$, or $l_{23}$. In this case, no material boundaries exist in the tetrahedron $T$.

- The edges of the tetrahedron $T_\alpha$ intersect at least one of the line segments $l_{12}$, $l_{13}$, or $l_{23}$, but the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the center of the barycentric triangle, does not lie inside any of the faces of $T_\alpha$. In this case, the intersection line segments for each face of $T$ are calculated and a surface triangulation inside $T$ is determined from these segments by using the triangulation rules of an isosurface extraction algorithm [5]. Figures 10a-d illustrate possible cases.

- The center point lies inside two faces of $T_\alpha$. In this case, two faces of $T$ will have a line segment connecting two edge intersection points, and two faces have three line segments meeting in the interior of these faces.

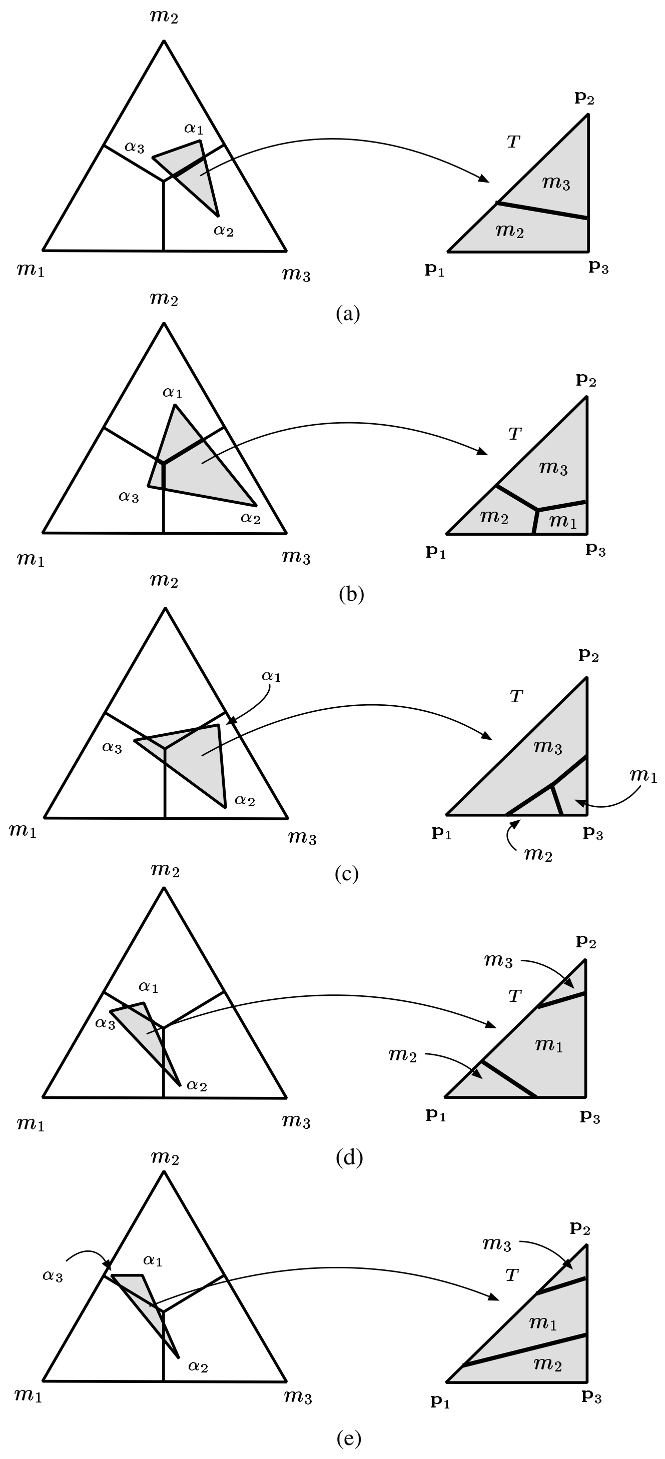


FIGURE 9: Mapping from material space to Euclidean Space for a triangle $T$. The images on the left show the triangle $T_\alpha$ in material space, and the images on the right show the material boundary line segments mapped from the intersections of $T_\alpha$ with the Voronoi cell boundaries to the Euclidean-space coordinates of the triangle $T$.

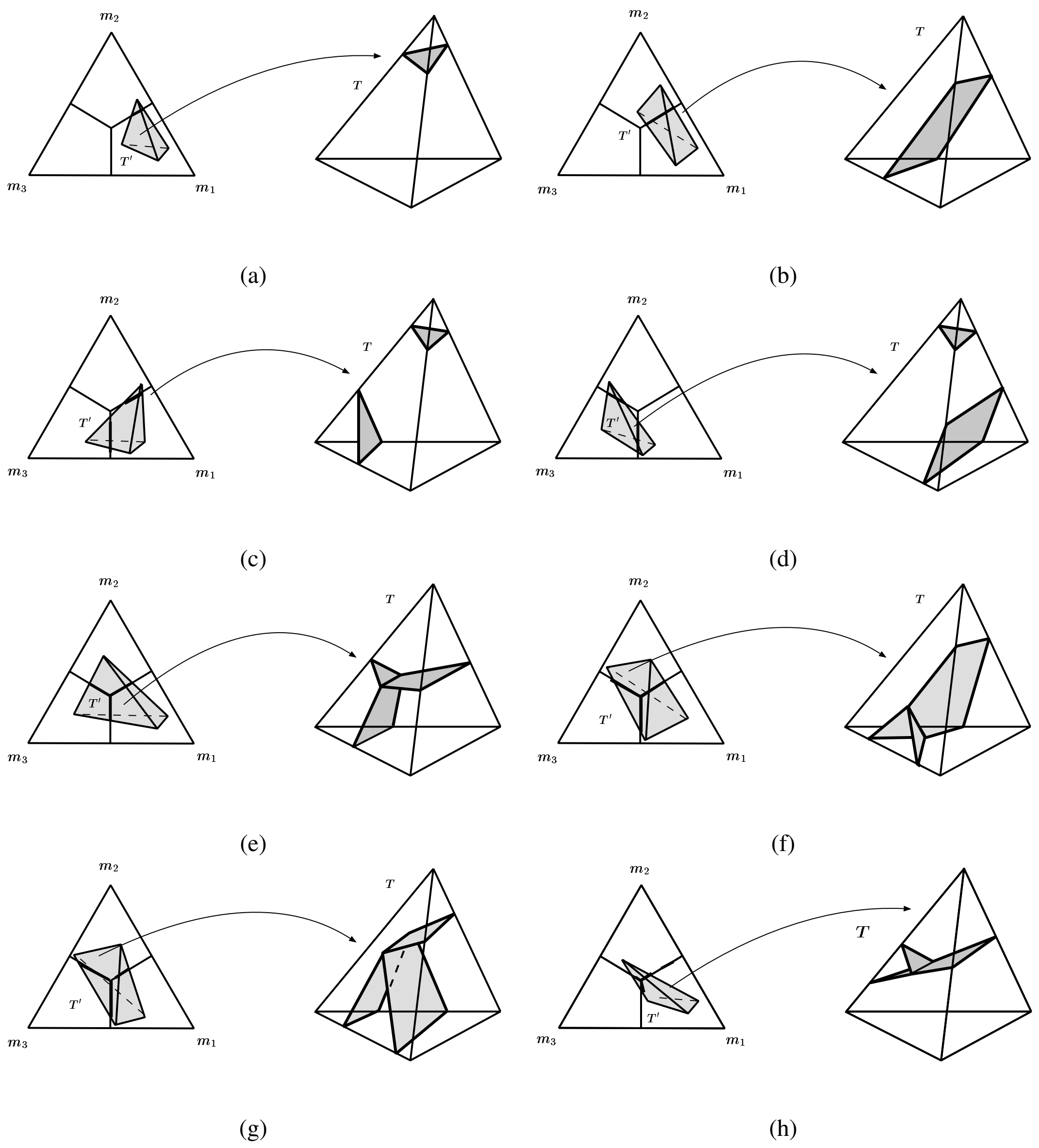


FIGURE 10: Material boundary construction for the three material case for tetrahedra.

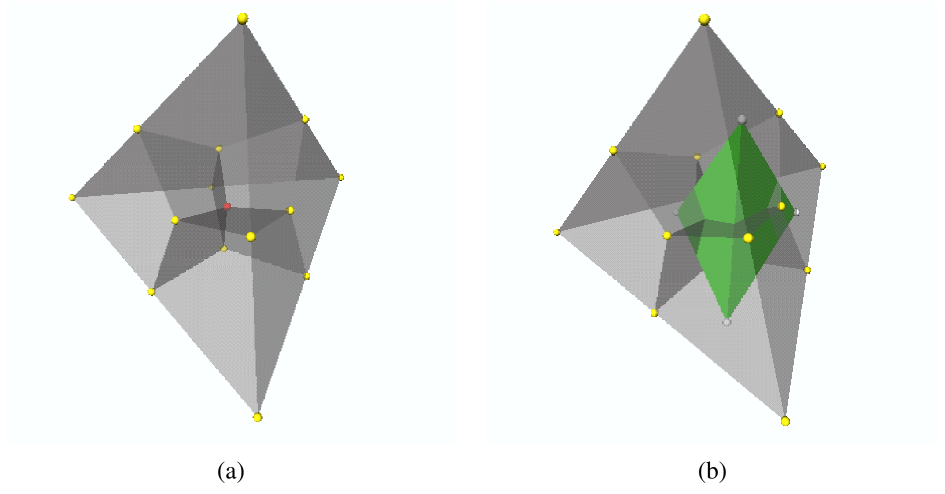


FIGURE 11: Voronoi cell decomposition for the four-material case. The £gure illustrates a three-dimensional projection of the 3-simplex. The 3-simplex is segmented into four Voronoi cells in (a). A 3-simplex $T_\alpha$, mapped from a tetrahedron $T$, is shown inside the 3-simplex in (b).

Using these line segments, a valid triangulation of the boundary surface can be determined. Figures 10e-h illustrate the possible cases.

## 3.3. The General Case

A tetrahedron (triangle) $T$ in an unstructured simplicial grid contains $k$-*materials* if there are $k$ indices $i_1, i_2, ..., i_k$, such that the associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2 ..., \alpha_m)$. of each vertex of $T$ has the property that $\alpha_i = 0$ for $i \neq i_1, ..., i_k$. It is helpful to examine the $k$-material case by £rst looking at the four-material case.

In the case of four materials, it is suf£cient to assume that each vertex of $T$ has an associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, where $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$, and $\alpha_i \geq 0$. By considering the 3-simplex having vertices $(1, 0, 0, 0)$, $(0, 1, 0, 0)$, $(0, 0, 1, 0)$, and $(0, 0, 0, 1)$ in material space, a partition of this simplex into Voronoi cells can be de£ned. The boundaries of these cells are bounded by the faces of the 3-simplex and six hyperplanes, de£ned by the set of $\alpha$ such that (i) $\alpha_1 = \alpha_2$, (ii) $\alpha_1 = \alpha_3$, (iii) $\alpha_1 = \alpha_4$, (iv) $\alpha_2 = \alpha_3$, (v) $\alpha_2 = \alpha_4$, and (vi) $\alpha_3 = \alpha_4$. The resulting Voronoi partition is shown in Figure 11a.

If $T$ is a triangle in a two-dimensional unstructured grid, the barycentric coordinate tuples associated with the vertices of $T$ are mapped into a triangle $T_\alpha$ in the material space 3-simplex. A clipping algorithm is applied to $T_\alpha$ to generate intersections with the boundaries of the Voronoi cells, by clipping against each of the six hyperplanes de£ning the Voronoi-cell boundaries.

Intersections can be found by a simple procedure. Suppose that an edge of $T_\alpha$ with endpoints $\alpha^{(1)}$ and $\alpha^{(2)}$

crosses the hyperplane de£ned by $\alpha_1 = \alpha_2$. If $\alpha$ is the intersection point, we can compute $r$ such that

$$\alpha = (1-r)\alpha^{(1)} + r\alpha^{(2)}.$$

If the £rst two coordinates of $\alpha$ are equal, then

$$(1-r)\alpha_1^{(1)} + r\alpha_1^{(2)} = (1-r)\alpha_2^{(1)} + r\alpha_2^{(2)},$$

which allows us to calculate $r$ directly. (See Hanson [14] for similar methods.) Once the intersections are determined by the clipping algorithm, the polygons in $T_\alpha$ are used to determine polygons in the Euclidean coordinates of $T$, which represent the material boundary.

In the $k$-material case, a tetrahedron (triangle) $T$ has an associated $k$-simplex $T_\alpha$ in material space. The $k$-simplex is partitioned into Voronoi cells whose boundaries consist of the faces of the $k$-simplex and the $\binom{k}{2}$ hyperplanes de£ned by the equations $\alpha_i = \alpha_j$, where $1 \le i < j \le k$. The intersections of $T_\alpha$ with the boundaries of the Voronoi cells are calculated performing clipping. The polygons of $T_\alpha$, determined by the clipping algorithm, are then used to determine polygons in the Euclidean coordinates of $T$ in physical space, which represent the material boundary in $T$.

## 4. DISCUSSION

The algorithm presented here is a generalization of the Nielson-Franke algorithm [10]. To duplicate this method, each vertex is associated with exactly one material. In this case, our algorithm produces the same results produced by the Nielson-Franke algorithm.

## 5. RESULTS

We have generated material interfaces for a variety of data sets. Figure 12 illustrates the material interfaces for a data set consisting of three materials. The boundary of the region containing material 1 has a spherical shape, and the other two material regions are formed as concentric layers around material 1 – forming two material interfaces. The original grid is rectilinear-hexahedral, consisting of $64 \times 64 \times 64$ cells. The dual grid was constructed, and each dual cell was split into six tetrahedra, see Nielson [1], creating 1,572,864 tetrahedra. Approximately 30% of the tetrahedra that contain material boundaries have two boundary surfaces and require the construction illustrated in Figure 10c and 10d.

The algorithm generalizes to data sets having several concentric boundary layers. If we have $n$ possible materials per cell, the algorithm can return up to $n-1$ boundaries per cell.

Figure 13 shows the material interfaces for a three-material data set of a simulation of a ball striking a plate consisting of two materials. The original data set is rectilinear-hexahedral and has a resolution of $53 \times 23 \times 23$

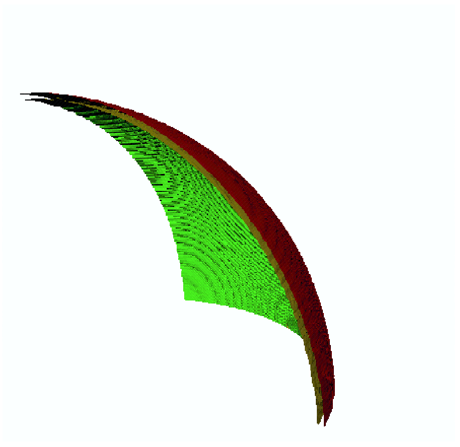


FIGURE 12: Boundary surfaces of three materials de£ned by two concentric spherical layers.

cells. Again, the dual grid was created, and each dual cell was split into six tetrahedra, creating 28,037 tetrahedra. Four time-steps are shown.

Figure 14 illustrates the material interfaces for a human brain data set containing three materials. The original grid is rectilinear-hexahedral containing $256 \times 256 \times 124$ cells. Each cell contains a probability tuple de£ning the probability that a material is present at the point: gray matter, white matter, background, or other material. The resulting dual data set contains over eight million tetrahedra.

## 6. ERROR ANALYSIS

Given a data set and an extracted material interface, we can use the generated boundary to approximate material fractions for each cell and compare them to the original fractions. Given an original cell $C$, and a point $\mathbf{p}_C$ at the center of the cell, then $\mathbf{p}_C$ is the coordinates of a vertex must be the coordinates of a number of tetrahedra that represent the dual grid. Let $\mathcal{T}_C$ be the set of tetrahedra that contain $\mathbf{p}_C$ as a vertex. Each tetrahedron of $\mathcal{T}_C$ is partitioned into a set of polyhedra, each polyhedron containing a single material. These polyhedra are clipped against the boundaries of $C$, and the volumes of the clipped polyhedra are added to the volume fractions for $C$. Normalizing by the volume of the cell, we obtain a set of volume fractions determined by the extracted material interface. This procedure enables us to calculate the difference between the original volume fractions and volume fractions implied by the extracted material interface. It is not accurate on the boundary of the data set since the dual cells do not cover the original cells there.

Figures 15 and Figures 17 illustrate the errors calculated from the "thin shells" and the "brain" data sets, respectively. Errors are reported as numbers of cells that fall into a certain error range (Dual cells that contained only a single material are not reported.) The £rst error range is the interval (0, 0.05]. Figure 15 shows the errors for the data set shown in Figure 12. There are $250,047$ cells total. The number of zero-error cells for materials

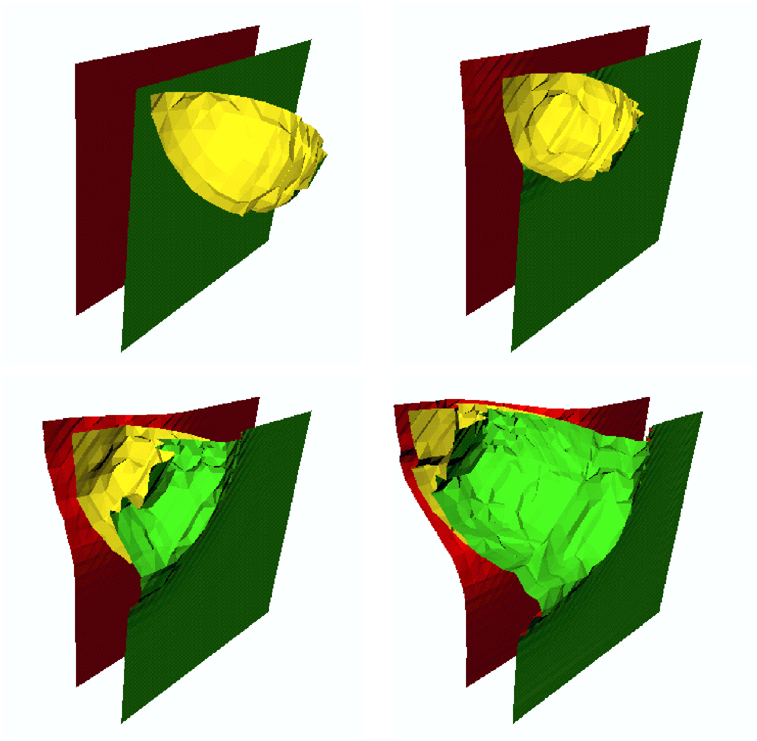


FIGURE 13: Time-dependent simulation of a ball striking a plate consisting of two materials. The sequence shows the boundary surfaces as the ball penetrates the plate.

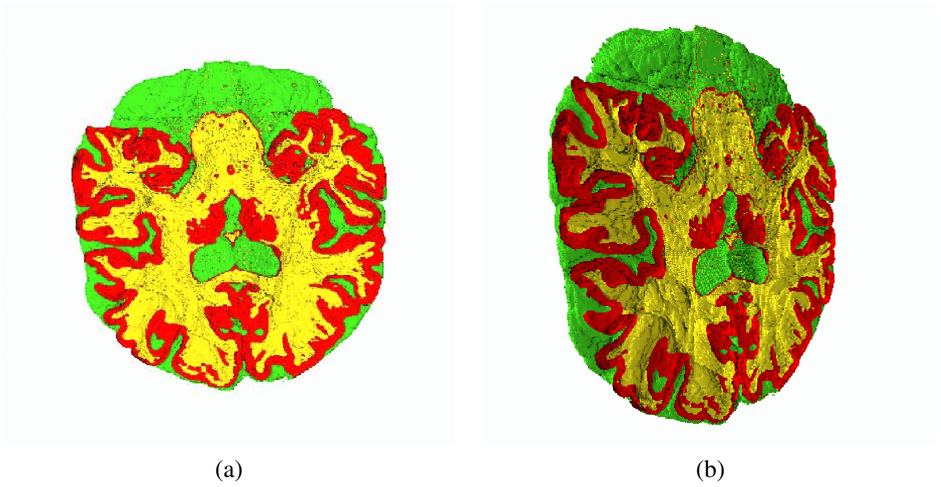
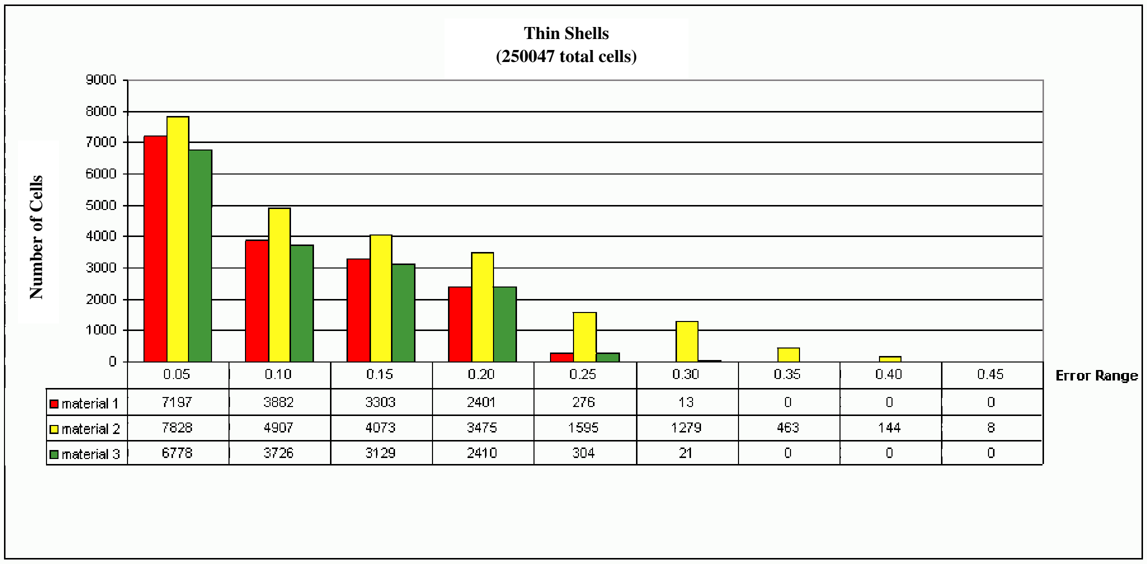


FIGURE 14: Brain data set. Material boundary surfaces are shown in red, green, and yellow. The polygons de£ning the material boundaries are clipped to show the interior of the data set. Two views of the material boundary surfaces, are shown in (a) and (b).



| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 | 0.45 |
|---|---|---|---|---|---|---|---|---|---|
| 🟥 material 1 | 7197 | 3882 | 3303 | 2401 | 276 | 13 | 0 | 0 | 0 |
| 🟨 material 2 | 7828 | 4907 | 4073 | 3475 | 1595 | 1279 | 463 | 144 | 8 |
| 🟩 material 3 | 6778 | 3726 | 3129 | 2410 | 304 | 21 | 0 | 0 | 0 |

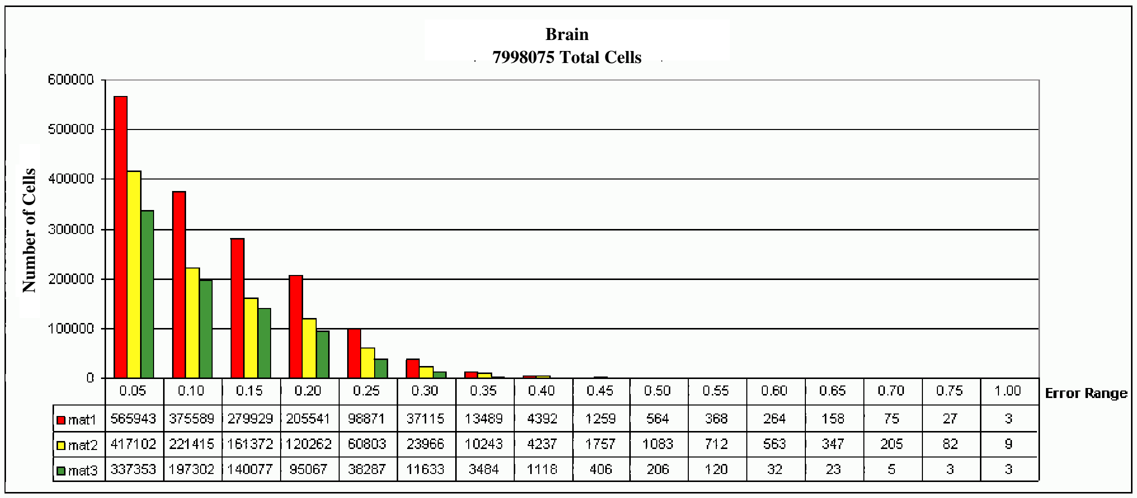
FIGURE 15: Error analysis for ”thin-shells data set” shown in Figure 12.

| **Thin Shells** | **material 1** | **material 2** | **material 3** |
|---|---|---|---|
| Original volume fractions (summed over original mesh) | 0.5205060 | 0.0276901 | 0.4518040 |
| New volume fractions (summed over dual mesh) | 0.5155890 | 0.0279787 | 0.4564320 |
| | | | |
| difference: | 0.0049170 | 0.0002886 | 0.0046280 |

FIGURE 16: Summing the total fractions.

1, 2 and 3 are 232, 975, 226, 275, and 233, 679, respectively. Figure 16 provides a comparison of the original and new volume fractions. If we sum the volume fractions for the complete data set and compare it with the calculated fractions, the error is actually quite low. This is evident from Figure 16.

Figure 17 shows the errors for the brain data set shown in Figure 14. The number of zero-error cells for materials 1, 2, and 3 are 6, 414, 488, 6, 973, 917, and 7, 172, 956, respectively. Figure 18 provides a comparison of original and new volume fractions. In general, the approximation faithfully represents the material interface, with little error.



| | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 | 0.45 | 0.50 | 0.55 | 0.60 | 0.65 | 0.70 | 0.75 | 1.00 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 🟥 mat1 | 565943 | 375589 | 279929 | 205541 | 98871 | 37115 | 13489 | 4392 | 1259 | 564 | 368 | 264 | 158 | 75 | 27 | 3 |
| 🟨 mat2 | 417102 | 221415 | 161372 | 120262 | 60803 | 23966 | 10243 | 4237 | 1757 | 1083 | 712 | 563 | 347 | 205 | 82 | 9 |
| 🟩 mat3 | 337353 | 197302 | 140077 | 95067 | 38287 | 11633 | 3484 | 1118 | 406 | 206 | 120 | 32 | 23 | 5 | 3 | 3 |

FIGURE 17: Error analysis for the brain data set in Figure 14.

| **Brain** | **material 1** | **material 2** | **material 3** |
|---|---|---|---|
| Original volume fractions (summed over original mesh) | 0.1177090 | 0.0840189 | 0.7982720 |
| New volume fractions (summed over dual mesh) | 0.1225630 | 0.0840548 | 0.7933830 |
| | | | |
| difference: | 0.0048540 | 0.0000359 | 0.0048890 |

FIGURE 18: Summing the total fractions.

## 7. CONCLUSIONS

In this paper, we have presented an algorithm for material interface construction from data sets containing volume-fraction information. A given grid is transformed to an unstructured representation of a dual grid, where each vertex has an associated barycentric coordinate tuple that represents the fractions of each material. The material interfaces are constructed by de£ning triangles (tetrahedra) in materials space and calculating the intersections with the boundaries of Voronoi cells in material space. These intersection points are used to de£ne intersections in the Euclidean coordinates of the original tetrahedron (triangle) space and triangulated to form the resulting boundary surface approximation. The algorithm can treat any number of materials per cell.

In the future, we would like to add a "measure-and-adjust" feature to this algorithm. Once an initial boundary surface approximation is calculated, (new) volume fractions can be calculated directly from this boundary surface approximation, as shown in Section 6. It is then possible to adjust material interfaces to minimize volume fraction deviations. It may also be possible to adjust the material interface within each simplex (or higher-level cell), to "optimize" the material interface. If so, it will be possible to better preserve volume fractions on a per-simplex (per-cell) basis.

## 8. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award), through the Large Scienti£c and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Of£ce of Naval Research under contract N00014-97-1-0222; the Army Research Of£ce under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; and the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159. We thank the members of the Visualization Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis for their support in this work.

We would like to thank the reviewers of an initial version of this paper. Their conscientious reviews helped make this a much better paper.

## References

[1] G. M. Nielson, “Tools for triangulations and tetrahedrizations and constructing functions de£ned over them,” in *Scienti£c Visualization: Overviews, Methodologies, and Techniques* (G. M. Nielson, H. Hagen, and H. Müller, eds.), pp. 429–525, IEEE Computer Society Press, Los Alamitos, CA, 1997.

[2] A. Okabe, B. Boots, and K. Sugihara, *Spatial Tesselations — Concepts and Applications of Voronoi Diagrams*. Wiley Publishing, Chichester, England, 1992.

[3] W. E. Lorensen and H. E. Cline, “Marching cubes: a high resolution 3D surface construction algorithm,” in *Computer Graphics (SIGGRAPH ’87 Proceedings)* (M. C. Stone, ed.), vol. 21, pp. 163–170, July 1987.

[4] G. M. Nielson and B. Hamann, “The asymptotic decider: Removing the ambiguity in marching cubes,” in *Proceedings of IEEE Visualization ’91*, pp. 83–91, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[5] Y. Zhou, B. Chen, and A. Kaufman, “Multiresolution tetrahedral framework for visualizing regular volume data,” in *IEEE Visualization ’97* (R. Yagel and H. Hagen, eds.), pp. 135–142, IEEE, IEEE Computer Society Press, Los Alamitos, CA, Nov. 1997.

[6] W. F. Noh and P. Woodward, “SLIC (Simple line interface calculation),” in *Lecture Notes in Physics* (A. I. van der Vooren and P. J. Zandbergen, eds.), pp. 330–340, Springer-Verlag, 1976.

[7] D. L. Youngs, “Time-dependent multi-matreial ¤ow with large ¤uid distortion,” in *Numerical Methods for Fluid Dynamics* (K. W. Morton and J. J. Baines, eds.), pp. 273–285, Academic Press, 1982.

[8] D. Gueyf£er, J. Li, A. Nadim, R. Scardovelli, and S. Zaleski, “Volume-of-¤uid interface tracking with smoothed surface stress methods for three-dimensional ¤ows,” *Journal of Computational Physics*, vol. 152, pp. 423–456, 1999.

[9] J. E. Pilliod and E. G. Puckett, “Second-order accurate volume-of-¤uid algorithms for tracking material interfaces,” technical report, Lawrence Berkeley National Laboratory, 2000.

[10] G. M. Nielson and R. Franke, “Computing the separating surface for segmented data,” in *Proceedings of IEEE Visualization ’97* (R. Yagel and H. Hagen, eds.), pp. 229–234, IEEE Computer Society Press, Los Alamitos, CA, Oct. 1997.

[11] K. Bonnell, D. Schikore, M. Duchaineau, B. Hamann, and K. I. Joy, “Constructing material interfaces from data sets with volume-fraction information,” in *Proceedings of IEEE Visualization 2000* (T. Ertl, B. Hamann, and A. Varshney, eds.), pp. 367–372, IEEE Computer Society Press, Los Alamitos, CA, Oct. 2000.

[12] H. Samet, *The Design and Analysis of Spatial Data Structures*. Series in Computer Science, Addison-Wesley, Reading, Massachusetts, 1990.

[13] K. Bonnell, “On material boundary surfaces,” M.S. thesis, Department of Computer Science, University of California, Davis, June 2000.

[14] A. J. Hanson”, *”Geometry for N-Dimensional Graphics*, pp. ”149–170”. ”Academic Press, Boston”, ”1994”.