Material Interface Reconstruction

By

Kathleen Sue Bonnell

B.S. (California State University, Dominguez Hills) 1998

THESIS

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved by the Committee in charge:

Chair Dr. Kenneth I. Joy

Dr. Bernd Hamman

Dr. Daniel R. Schikore

2000

Material Interface Reconstruction

Abstract

This thesis presents a new algorithm for material boundary interface reconstruction from data sets containing volume fractions. The reconstruction problem is transformed to a problem that analyzes the dual data set, where each vertex in the dual mesh has an associated barycentric coordinate tuple that represents the fraction of each material present. After constructing the dual tetrahedral mesh from the original mesh, material boundaries are constructed by mapping a tetrahedron into barycentric space and calculating the intersections with Voronoi cells in barycentric space. These intersections are mapped back to the original physical space and triangulated to form the boundary surface approximation. This algorithm can be applied to any grid structure and can treat any number of materials per element/vertex. It is a generalization of previous work done in other fields, allowing for any number of materials to be present in a given volume. This algorithm generates continuous surfaces. Experimental results shows the preservation of overall volume fractions within a difference range of 0.5

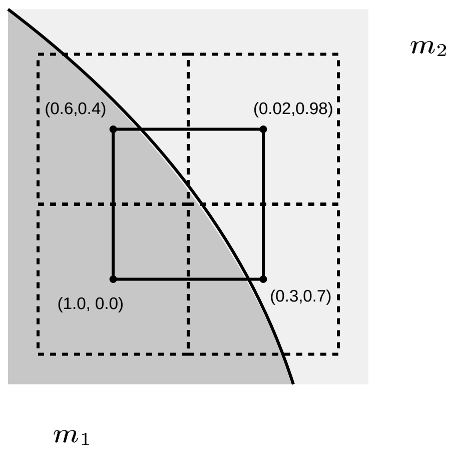


Figure 1: Grid and dual grid. The original grid (dashed lines) is replaced by a dual grid (solid lines), obtained by connecting the centers of the original elements. Barycentric coordinates are associated with each vertex of the dual grid. The barycentric coordinates represent the fractions of each material present in the original grid cells.

1. INTRODUCTION

There are numerous instances in which it is necessary to reconstruct or track the boundary surfaces (or "interfaces") between multiple materials that commonly result from finite element simulations. Multi-fluid Eulerian hydrodynamics calculations require geometric approximations of fluid interfaces to form the equations of motion to advance these interfaces correctly over time. In typical simulations, the grid cells (finite elements) contain fractional volumetric 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.*, α_i represents the fractional part of material i. It is assumed that $\alpha_1 + \alpha_2 + \cdots + \alpha_m = 1$ and $\alpha_i \geq 0$. Considerations in approaching this problem involve finding a (crack-free) piecewise two-manifold separating surface approximating the boundary surfaces between the various materials, local and global correctness, and spatial and temporal smoothness.

To solve this problem, consideration is given to the dual data set constructed from the given data set, 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 is associated with tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$, where m is the number of materials present in the data set, $\alpha_1 + \alpha_2 + \cdots + \alpha_m = 1$, and $\alpha_i \geq 0$. Thus, the boundary surface reconstruction problem reduces to constructing the material interfaces for a grid where each vertex has an associated barycentric coordinate representing the fractional parts of each material at the vertex. This “barycentric coordinate field” is used to approximate the material boundary surfaces.

Important applications of this problem occur for all grid types, *e.g.*, when the data points lie on a rectilinear grid, curvilinear grid, or an unstructured grid. Therefore a solution strategy is developed that is tailored to tetrahedral grids, as all other types of grid structures can be converted to this form, as demonstrated by Nielson [1]. In the case of rectilinear, curvilinear, or even hybrid polyhedral meshes, a given grid is preprocessed by subdividing each polyhedral cell into tetrahedra, and applying this algorithm to the resulting tetrahedral grid.

Given a data set containing m materials, one can process each tetrahedral cell of the grid and map our tetrahedral elements into an m simplex representing m-dimensional barycentric space. Next, one can calculate intersections with the edges of *Voronoi cells* [2] in the m-simplex. These Voronoi cells represent regions, where one material “dominates” the other materials locally. One can map these intersections back to the original space and triangulate the resulting points to obtain the boundary.

Section 2 describes previous work dealing with material boundary surfaces. Section 3 describes the two-material case, which can be viewed as a simple extension of a marching cubes/tetrahedra algorithm [3, 4, 5]. Section 4 describes the three-material case. Here, material boundaries are calculated in barycentric space (a triangle) and mapped back to the original data set. The general m-material case is described in Section 5. In this case, intersections are calculated in a barycentric m-simplex and mapped back to the tetrahedra in the data set. Implementation details are described in Section 6. Section 7 presents results for various data sets, and Section 8 provides error analysis for the same data sets. Section 9 contains conclusions and describes possible future work.

2. RELATED WORK

The bulk of research in material interface reconstruction has been conducted in computational fluid 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, describing a method for geometric approximation of fluid interfaces. Their algorithm is used in conjunction with hydrodynamics simulations to track the advection of fluids. Their algorithm produces an interface consisting of line segments, constructed parallel or perpendicular to a coordinate axis. Multi-fluid cells can be handled by grouping fluids together, calculating the interface between the groups, subdividing the groups, and iterating this process until interfaces are constructed. 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 classified according to the fluid index. The fluid index is simply the presence (1) or absence (0) of a material. Mixed fluid cells have multiple fluid indices, one per material. Fluids with the same fluid index are grouped together in order that only two types may be treated at one time, and the boundary constructed.

Consider a two-fluid 2D cell for and x-direction pass of the SLIC algorithm, consisting of materials A (30%) and B (70%). The left neighbor has only material A and the right neighbor has only material B. The interface in the mixed fluid cell would consist of a vertical line dividing the cell into two regions, 30% A on the left and 70% B on the right. Now conisder that same mixed fluid cell with different left and right neighbors. Suppose that both left and right neighbors consist 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.

Iinterface lines parallel to the pass direction axis are generally only drawn in mixed fluid cells consisting of three or more materials. Consider a three-fluid 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 two vertical lines, dividing the cell into three zones, with material A on the left, C in 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 first 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.

Figure 2 is an example data set that will be used to compare the various algorithms to be discusssed.

Figure 3 shows the approximation generated by the SLIC aglorithm for Figure 2 in an

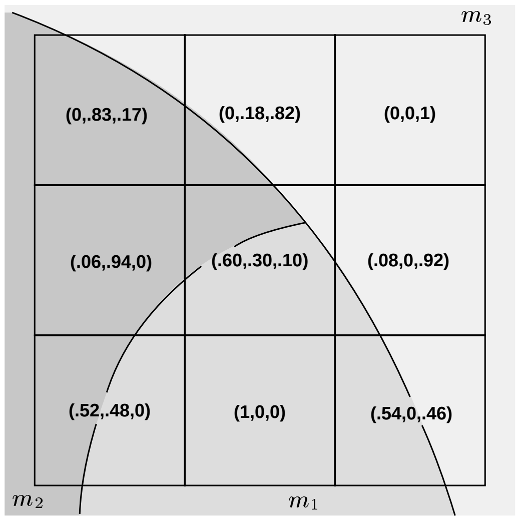


Figure 2: Three materials contributing to nine cells. The volume fractions for each material are shown in the individual cells.

x-coordinate and y-coordinate pass. It must be noted that although the interface is discontinuous, volume fractions are strictly perserved per 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 the volume fractions in a cell. Multiple materials are treated by grouping materials and determining interfaces on a two-material basis. Again, the interfaces produced by this method 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 pythagoream theorem. Figure 4 demonstrates the neigbors of a cell containing materials A and B. The cell is treated as a unit-square for this calculation. The slope is defined as: $\sqrt{(\delta_A - \gamma_A)^2 + (\beta_A - \alpha_A)^2}$, where α_A, β_A, γ_A, and δ_A are the fractions of material A present in the neighbor cells. Youngs mentions that neighboring ”corner” cells may also be used to determine the slope if this equation fails.
(One needs to be careful not to overlook the negative slope. If one (but not both) of the

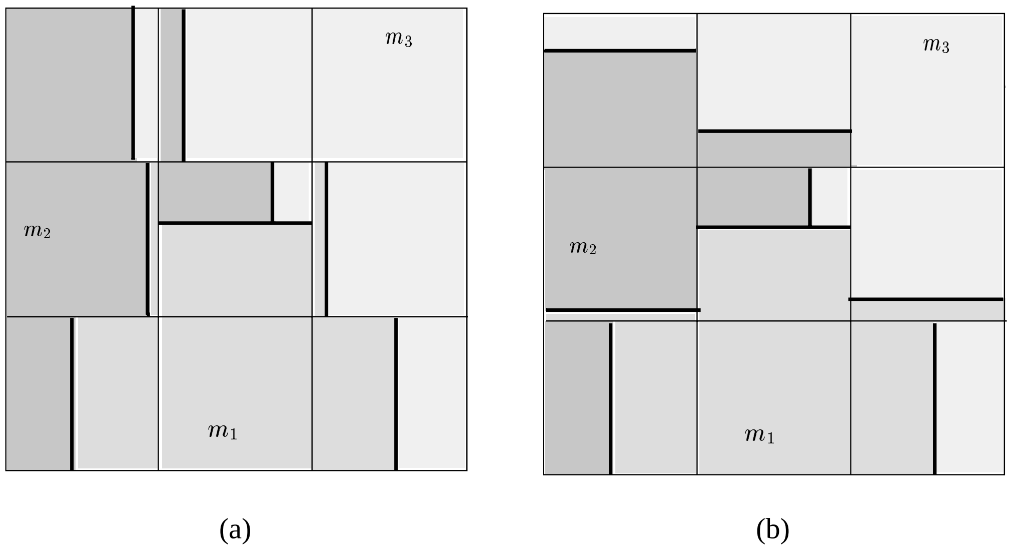


Figure 3: SLIC approximation in x-pass (a) and y-pass (b).

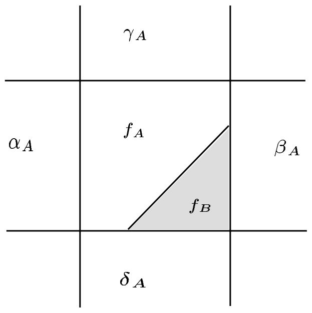


Figure 4: A two-material cell with material fractions f_A and f_B. Neighbor cells have material fractions α_A, β_A, γ_A, and δ_A for material A.

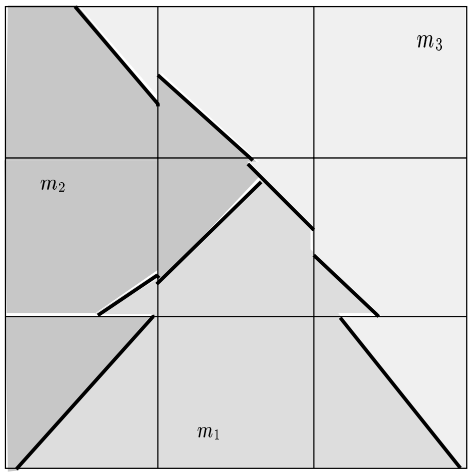


Figure 5: Result of Youngs' Algorithm. The material interface representation is discontinuous.

interior differences is negative, the negative root should be used for the slope.)

Once the slope is determined, the line segment with this slope is positioned in the cell such that the volume fractions are preserved. For more than two materials, the user must determine in which order the interfaces must be calculated. For example, when A, B, C, and D are present, are they grouped first by A and BCD or AB and CD? Different interfaces will result depending on the ordering/grouping chosen. Figure 5 demonstrates the results of applying Youngs' algorithm to the example in Figure 2.

The algorithm of Gueyffier [8] is similar to that of Youngs in that it requires an estimate of the normal vector to the interface in order to reconstruct the interface. He utilizes finite differencing or least-squares methods to determine this normal, depending upon the order of accuracy (first or second) desired. As with the SLIC algorithm, this one is used to track the advection of fluids.

In 2D, a line segment representing the boundary surface is constructed perpendicular to the interface normal. The line segment is positioned within the cell such that it divides the cell into appropriately proportioned areas. For 3D, a cutting-plane is computed whose normal is the computed interface normal. Again, the cutting plane is positioned within the cell so that volume fractions are preserved.

It is unclear how multiple materials can be handled, unless they are grouped as in other algorithms discussed here, and the cutting planes clipped appropriately.

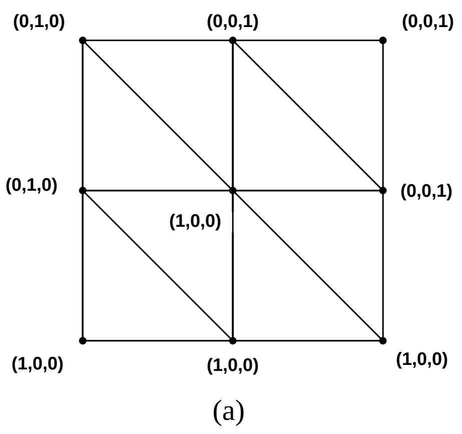
Pilliod and Puckett [9] compare various volume-of-fluid interface reconstruction algorithms, including SLIC, noting differences in the surfaces reconstructed and demonstrating first-order or second-order accuracy. Their goal is to reproduce a linear interface.

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/tetrahedra algorithm, but instead of using a strict binary classification of vertices, it allows any number of classes. Edges in tetrahedral grids whose endpoints have different classifications are intersected by the separating surface. Similarly, the faces of a tetrahedron whose three vertices are classified differently, are assumed to be intersected by the surface in the middle of the face. When all four vertices of a tetrahedron have different classifications, the boundary surface intersects in the interior of the tetrahedron. The resulting “mid-edge”, “mid-face” and “mid-tetrahedron” intersections are triangulated to form the surface.

Using our dual-cell representation, Figure 6 shows how Nielson’s and Franke’s algorithm might be applied to the example in Figure 2. As this algorithm requires classification of vertices, the greater α_i value in the volume fraction tuple for each cell was chosen as the classifier. The dual mesh has also been triangulated so that the algorithm might be applied. Mid-edge intersections are made half-way between two endpoints that have different classifications. Mid-face intersections are taken when all three vertices of the triangle have different classifications.

The algorithm presented here generalizes the above schemes. It utilizes a grid that has a barycentric coordinate associated with each vertex. This allows the generation of material boundaries directly from the intersections calculated in “barycentric space.” This algorithm handles multiple materials and can reconstruct layers and “Y-type” (non-manifold) interfaces with equal ease. This algorithm does not rely on application-specific knowledge of hydrodynamics or other simulation codes, but solves the problem from a purely mathematical viewpoint.

Figure 7 demonstrates the results of applying this algorithm to the example in Figure 2. It should be noted that the intersection point for this algorithm is not in the correct cell, for the reasons provided in Section 6.



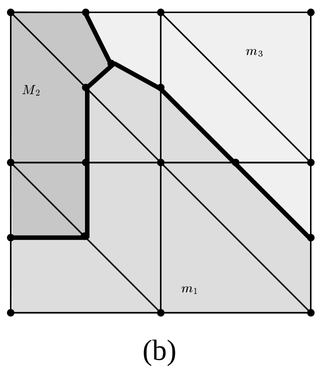


Figure 6: Dual representation (a) and result (b) for the Nielson-Franke Algorithm.

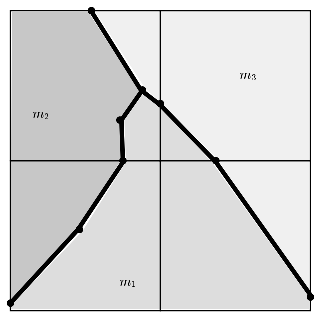


Figure 7: Result of the algorithm developed in this thesis. The material interface representation is continuous.

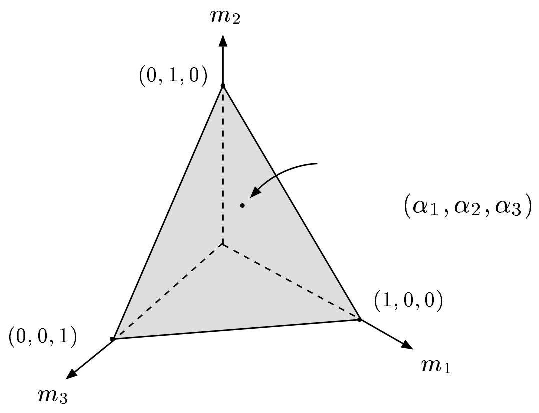


Figure 8: Barycentric triangle.

3. THE TWO-MATERIAL CASE

Consider a grid $\mathcal{S}$, where each vertex of $\mathcal{S}$ has an associated barycentric coordinate α. In the case of two materials, α is the two-tuple $\alpha = (\alpha_1, \alpha_2)$, where $\alpha_1 + \alpha_2 = 1$. In this case, I define the material boundary to be the set of points where $\alpha_1 = \alpha_2 = \frac{1}{2}$.[1] As linear interpolation of barycentric coordinates is used, only the point on each edge where $\alpha_1 = \alpha_2 = \frac{1}{2}$ needs to be found. By computing these points on all edges of a cell, I can use a marching tetrahedra implementation [5] to compute the boundary surface in a cell when only two materials are present. (If only one material is present in a cell, no boundary surface exists). The two-material case reduces to an isosurface calculation, determining the isosurface $\alpha_1 = \frac{1}{2}$, which can easily be implemented by a marching tetrahedra method.

4. THE THREE-MATERIAL CASE

In the three-material case, each vertex has an associated 3-tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 + \alpha_2 + \alpha_3 = 1$. Here, α_1 is the fraction of material m_1, α_2 is the fraction of m_2, and α_3 is the fraction of m_3, respectively. The coordinate $(\alpha_1, \alpha_2, \alpha_3)$ lies on the equilateral triangle with vertices $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$, as shown in Figure 8. This triangle

[1]The rationale behind this decision is that an infinitesimally small cell whose center is on the boundary will contain approximately half of each material.

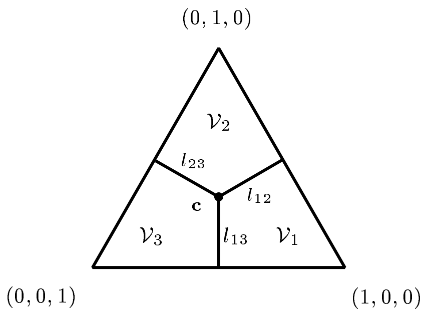


Figure 9: Partitioning the barycentric triangle into regions. The point $\mathbf{c}$ is the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the center of the triangle. The line segments l_{12}, l_{13}, and l_{23} bound the Voronoi cells $\mathcal{V}_j$ in the interior of the triangle.

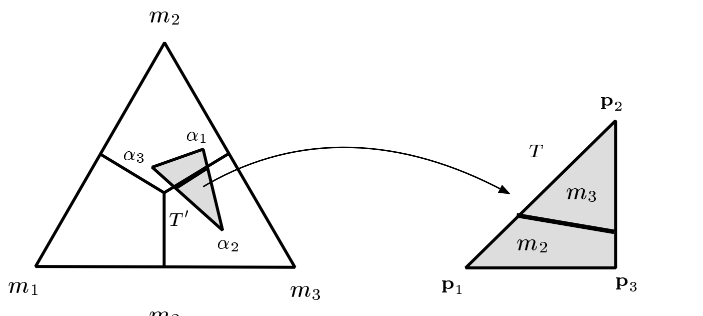
is partitioned into three regions, defined by the *Voronoi cells* $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$, see Figure 9. The Voronoi cells $\mathcal{V}_j$ are bounded by the edges of the triangle, and the three line segments l_{12}, l_{13}, and l_{23}, where $\alpha_1 = \alpha_2$ and $\alpha_3 \leq \frac{1}{3}$, $\alpha_1 = \alpha_3$ and $\alpha_2 \leq \frac{1}{3}$, or $\alpha_2 = \alpha_3$ and $\alpha_1 \leq \frac{1}{3}$, respectively.

For two-dimensional triangular grids, the associated barycentric coordinates of a triangle T are mapped onto a triangle T' in barycentric space. The intersections of the edges of T' with the edges of the Voronoi cells in the barycentric triangle are used to define material interfaces in T'. These intersections are then mapped back to points in T, using the same linear parameters to determine the intersections on the edges of T. There are three cases:

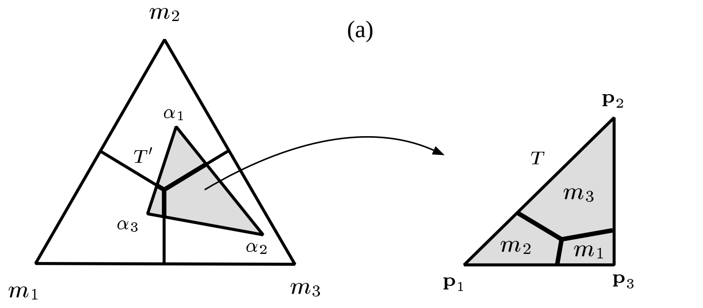
- The triangle T' does not intersect l_{12}, l_{13}, or l_{23}. In this case, it is assumed that no material boundary exists in T.

- The triangle T' 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'. In this case, intersections are calculated on the edges of T, corresponding to the intersections of T' with l_{12}, l_{13}, and l_{23}, respectively. (The triangle T' may intersect at most two of these lines.) The material boundary line segments inside T are then defined as the line segments that connect the corresponding edge intersections in T. Figures 10a, 10d, and 10e illustrate these cases.

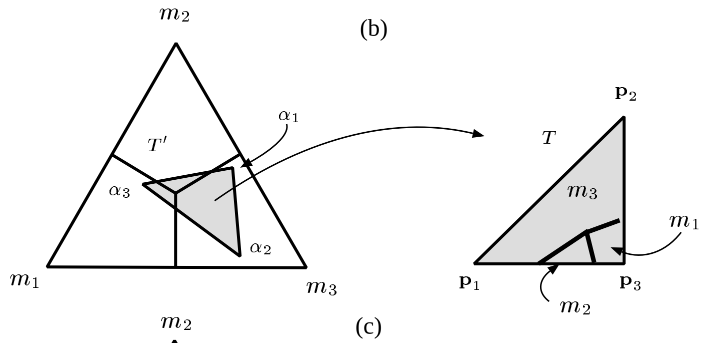
- The point $\mathbf{c}$ lies inside T'. In this case, three edge intersections are calculated for T,



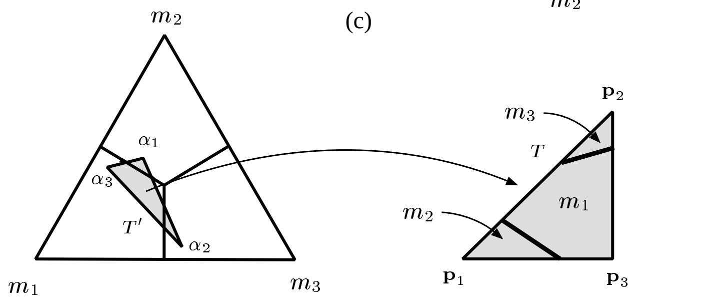
(a)



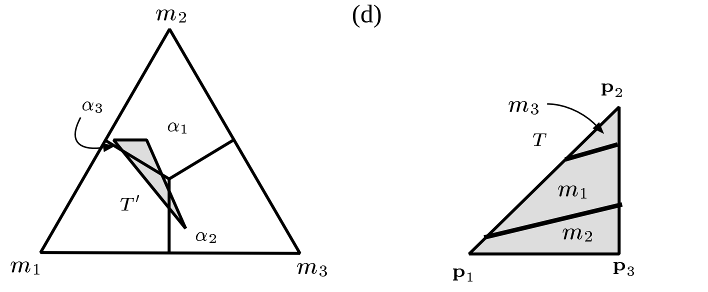
(b)



(c)



(d)



(e)

Figure 10: Mapping intersections from barycentric space to the triangle T. The images on the left show the triangle T' in barycentric space, and the images on the right show the material boundary line segments mapped from barycentric space to the original triangle T in physical space.

corresponding to the intersections of T' with l_{12}, l_{13}, and l_{23}, respectively, and a point in the interior of T, corresponding to the point $\mathbf{c}$ in T'. The material boundary line segments are defined as the three lines connecting the edge intersections and the face point. Figures 10b and 10c illustrate this case.

If one of the α_i values is zero for each of the three vertices of a triangle, then all points map to an edge of the barycentric triangle. Thus, the situation reduces to the two-material case. If only one material is present at all three vertices, then no intersections are calculated.

For three-dimensional tetrahedral grids, the associated barycentric values of the vertices of each face of a tetrahedron T are used to map the tetrahedron to an image T' of T in barycentric space. Intersections are calculated separately for each face of T' and mapped back to T.

There are three cases:

- No edge of the tetrahedron T' 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' 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'. In this case, the intersection line segments for each triangular face of T are calculated and a triangulation is determined from these segments by following the marching tetrahedra algorithm [5]. Figures 11a, 11c, 12d, 12f, and 13g illustrate the possible cases.

- The center point of the barycentric triangle lies inside two faces of T'. In this case, two faces have a single material boundary line segment connecting two edge intersection points, and two faces have three material boundary line segments meeting in the interior of two faces. The intersections are mapped back to the tetrahedron T, using linear interpolation. Using the material boundary line segments for each face, and the line segment connecting the two points in the interior of two faces of T, a valid triangulation of the boundary surface can be determined. Figures 11b, 12e, reffig:tet-split3h, 13i illustrate the possible cases.

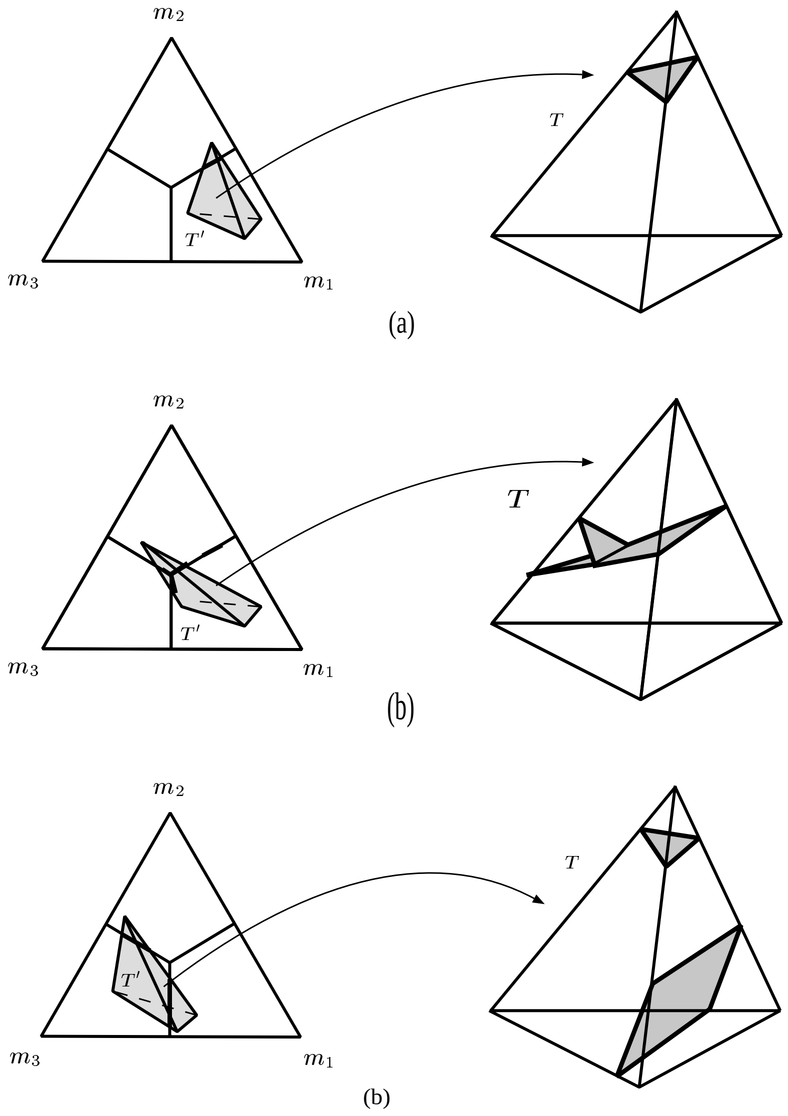
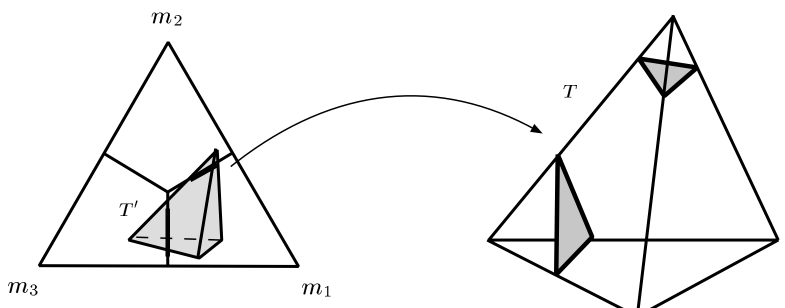
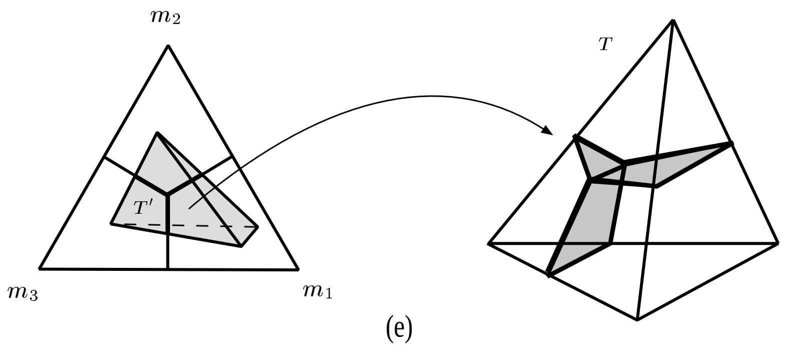


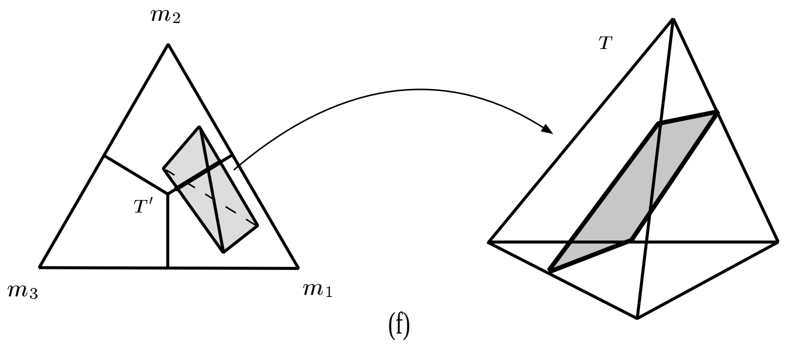
Figure 11: Examples of material boundary surface determination for tetrahedral grids.



(d)

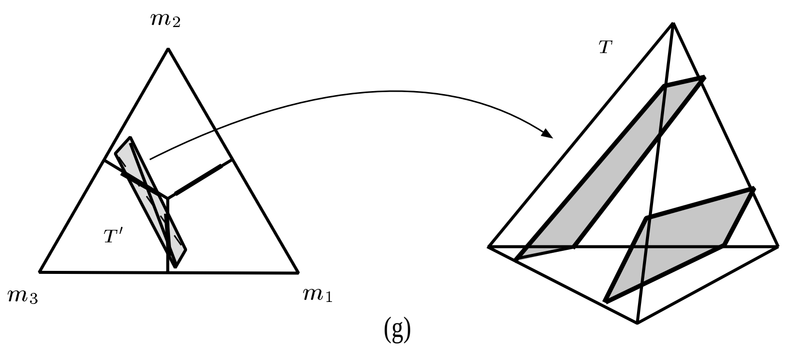


(e)

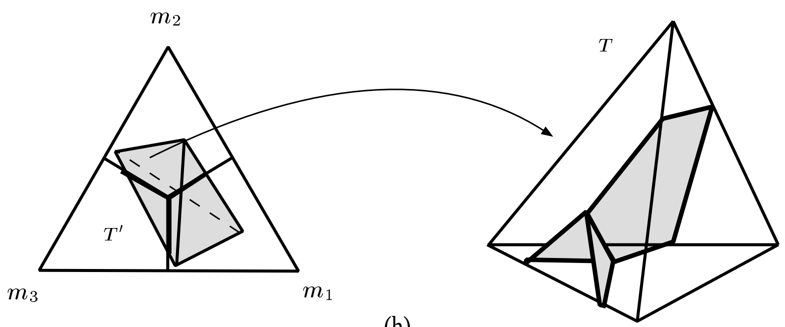


(f)

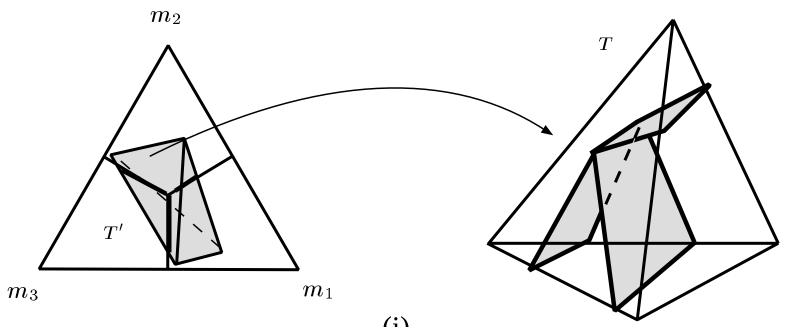
Figure 12: Examples of material boundary surface determination for tetrahedral grids.



(g)



(h)



(i)

Figure 13: Examples of material boundary surface determination for tetrahedral grids.

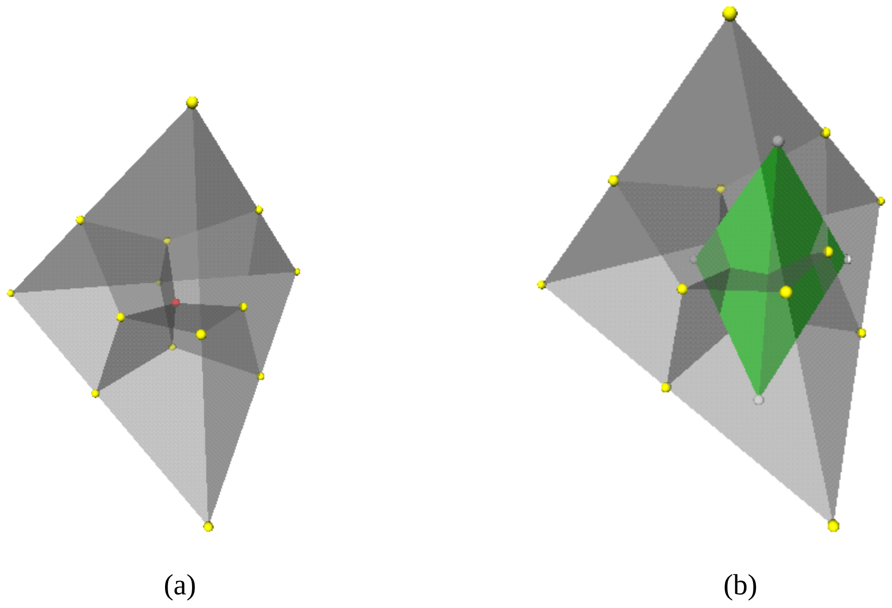


Figure 14: Voronoi cell decomposition in four-material case. The figure illustrates a three-dimensional projection of the barycentric tetrahedron from four-dimensional space. The tetrahedron is segmented into four Voronoi cells in (a). In (b), a tetrahedron, mapped from physical space, is shown inside the barycentric tetrahedron.

5. THE GENERAL CASE

In the case of four materials, each vertex has an associated barycentric coordinate given by the four-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 tetrahedron having vertices $(1, 0, 0, 0)$, $(0, 1, 0, 0)$, $(0, 0, 1, 0)$, and $(0, 0, 0, 1)$ in four-dimensional space, a partition of this tetrahedron can be constructed similarly to the three-material case. Again, the Voronoi cells are used for the decomposition of the barycentric tetrahedron. The boundaries of these cells include parts of the faces of the tetrahedron and six planar pieces, which are defined by $\alpha_1 = \alpha_2$, $\alpha_1 = \alpha_3$, $\alpha_1 = \alpha_4$, $\alpha_2 = \alpha_3$, $\alpha_2 = \alpha_4$, and $\alpha_3 = \alpha_4$. This Voronoi partition is shown in Figure 14a.

For two-dimensional grids, the four-dimensional barycentric coordinates associated with the vertices of a triangle T are mapped into a triangle T' in barycentric space. A clipping algorithm is used to generate the intersections in the triangle T', clipping against the six planes defining the boundaries of the Voronoi cells of the barycentric tetrahedron.

The tetrahedron is stored in a binary space partitioning (BSP) tree, and the clipping algorithm described by Samet [11] is applied. Once the intersections are determined by the clipping algorithm, the material boundary line segments can be determined for the triangle T.

For three-dimensional tetrahedral grids, a similar clipping algorithm is used for the image T' of a tetrahedron T. This enables the calculation of the boundary surfaces inside the tetrahedron T', which are then mapped back to the tetrahedron T in physical space.

In the general case of m materials, a tetrahedron T is mapped to a tetrahedron T' in an m-simplex in barycentric space. The m-simplex is partitioned into Voronoi cells whose boundaries consist of the faces of the m-simplex and the $\binom{m}{2}$ hyperplanes defined by $\alpha_i = \alpha_j$, where $1 \leq i < j \leq m$. The material boundaries for T' are calculated by using a clipping algorithm and then are mapped back to physical space to form the material boundaries inside T. A BSP algorithm is utilized to perform the clipping.

6. DISCUSSION

The algorithm presented in this thesis runs in effectively the same time as does the marching cubes/tetrahedra algorithm. I traverse the cells of a grid and calculate, for each cell, a polygonal representation of the material boundaries. Most grid cells in common examples contain only one material, and boundaries do not exist in these cells.

One must note that the algorithm can miss material boundaries in tetrahedra. In any isosurface-type algorithm, it is possible for the isosurface to enter a tetrahedron, but only intersect one edge. In this case, the algorithm cannot detect the material boundary from only the information at the vertices. It should also be ntoed that, as illustrated in Figure 11a, the tetrahedorn has non-zero fractions of all three materials, but only two materials are extracted.

In the three-material case, the point $\mathbf{c} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ has been chosen as the "center" of the barycentric triangle. This assumes that there are three distinct sectors in the barycentric triangle, subdividing the triangle in a "Y" fashion, and that a cell of infinitesimally small size contains about one-third of each material in the cell. This is not always the case. For example, consider a "T intersection," where any small cell would contain one-half of one material and one-quarter of the other two materials. The segmentation of the barycentric triangle can be adjusted so that the point $\mathbf{c}$ is at an arbitrary location in the triangle, and the edges that determine the intersections can be adjusted appropriately. This can be done by sampling in a larger neighborhood of a specific cell to understand how to weigh the materials about the “Y point.” This is a global process: neighboring cells must agree witht the change in order to maintain continuity.

In the four-material case, the center of the tetrahedron can also be adjusted. However, this implies that the “center” vertices on the faces must also be adjusted so that the separating surfaces remain planar. In the m-material case, similar considerations also hold when adjusting the center of the m-simplex.

The algorithm presented here can be considered as a direct generalization of the Nielson-Franke algorithm [10]. Each vertex of a grid $\mathcal{S}$ has an associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$, and by restricting material fractions such that exactly one $\alpha_i = 1$, the case is obtained where each vertex is only associated with one material. In this case, this algorithm produces the results produced by the Nielson-Franke algorithm.

7. RESULTS

This algorithm has been implemented and used to generate material interfaces for a variety of data sets. Figure 15 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 then each dual cell was split into six tetrahedra, see Nielson [1], creating 1,572,864 tetrahedra. Approximately 30% of the tetrahedra containing the material boundaries contain two boundary surfaces.

Figure 16 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 rectilinear-hexahedral and has a resolution of $53 \times 23 \times 23$ cells. Again, the dual grid was created, and each dual cell was split into six tetrahedra, creating 28,037 tetrahedra. Four instances of the data set are shown.

Figure 17 illustrates the material interfaces for a human brain data set. The original grid is rectilinear-hexahedral containing $256 \times 256 \times 124$ cells. Each cell contains a probability

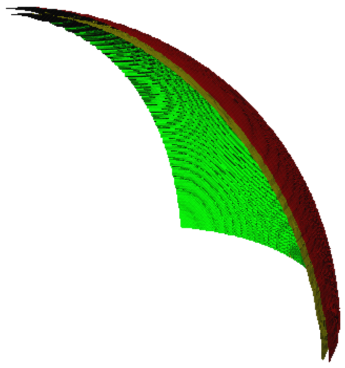


Figure 15: Boundary surfaces of two materials formed as concentric spherical layers.

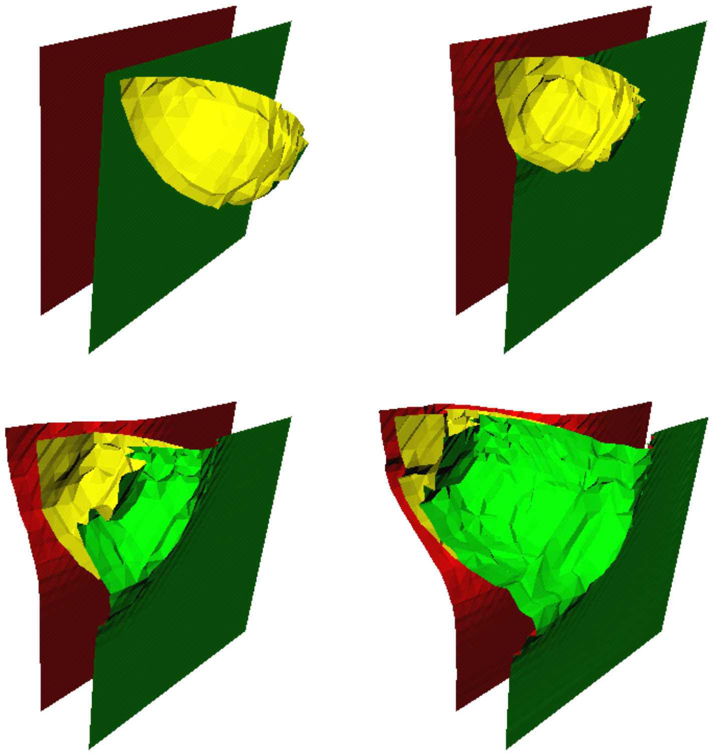
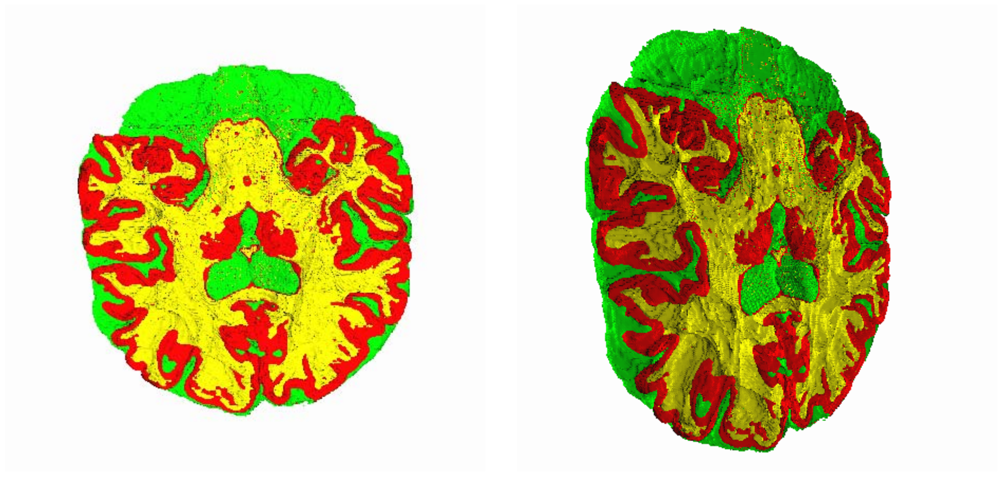


Figure 16: Simulation of a ball striking a plate consisting of two materials. The picture in the upper-left corner shows the initial configuration, and the following sequence of pictures shows the boundary surfaces as the ball penetrates the two-material plate.



(a) (b)

Figure 17: Brain data set. The material boundary surfaces are shown in red, green and yellow. The polygons forming the material boundaries are clipped to show the interior of the data set. Two views of the material boundary surfaces, rotated differently, are shown in (a) and (b).

tuple giving the probability that each material is present at the point. The resulting dual data set contains over eight million tetrahedra.

8. ERROR ANALYSIS

It would be interesting to know how the volume fractions resulting from the described method compare to the original volume fractions, both over the entire mesh and on a per-cell basis. To determine resulting volumes, a closed polyhedra is first computed, representing the individual material volumes, using the interface as a starting point. The volume of the polyhedra is then calculated. This provides volumes for each material in the tetrahedron. The volumes for each of the tetrahedra in a cell are summed, then averaged to volume fractions.

Comparing overall volumes is then a straightforward procedure. First, the original volume fractions are summed and averaged to volume fractions, then the resulting volume fractions are summed and averaged.

In order to perform a comparison on a per-cell basis, data representing similar things

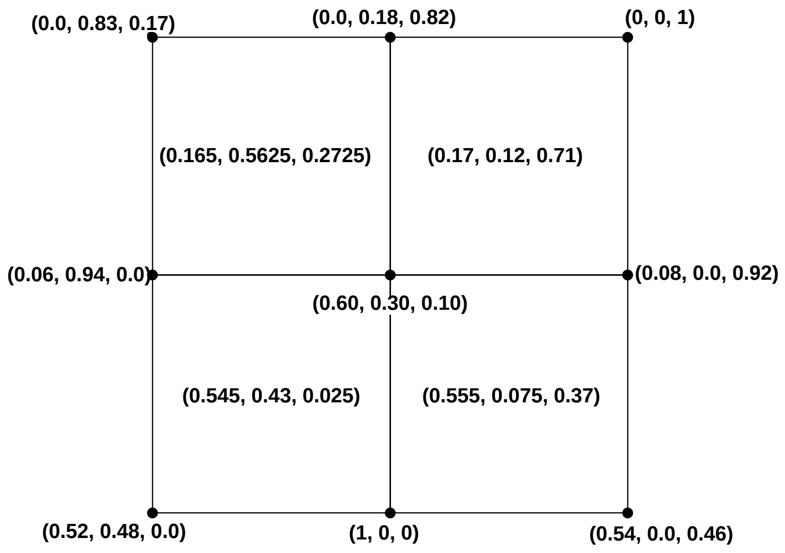


Figure 18: Average volume fractions at vertices of dual mesh. The averages are used to obtain ”original” volume fractions per cell.

are needed. For this reason, the original volume fractions at the vertices of a dual cell are averaged to obtain an original volume-fraction average per dual cell. These averages can then be compared to the resulting fractions. Figure 18 illustrates this point. Figure 19 shows the computed volumes fractions resulting from the application of this algorithm, and Figure 20 compares original volume fractions with these results.

The following charts illustrate the errors calculated from examples presented previously. Errors are reported as number of dual cells that fall into a certain error range. Dual cells that contained zero error for a particular material are not reported in the charts. Thus, the first error range is the interval (0, 0.05].

For Figure 21, which shows the errors for the data set in Figure 15, there were $250,047$ total dual cells. The number of zero-error dual cells for materials 1, 2 and 3 are 232975, 226275, and 233679, respectively. Figure 22 gives a comparison of the original and new volume fractions.

Figures 23, 25, 27, and 29, show the errors for the data sets in Figure 16. There were $25,168$ total dual cells in these data sets. Figures 24, 26, 28, and 30 give comparisons of the original and new volume fractions.

Figure 31 charts the errors for the brain data set in Figure 17. The number of zero-

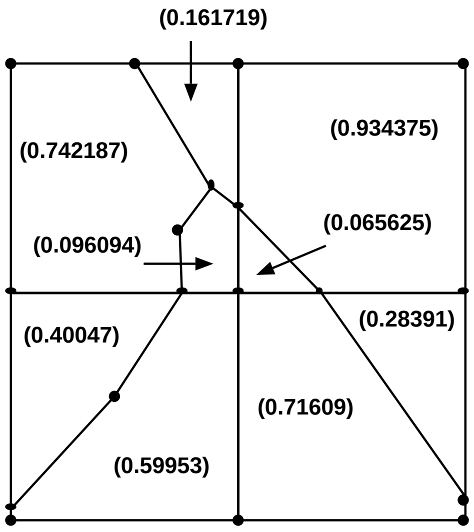
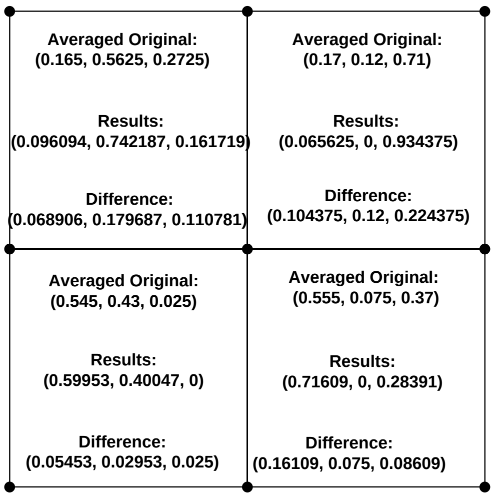
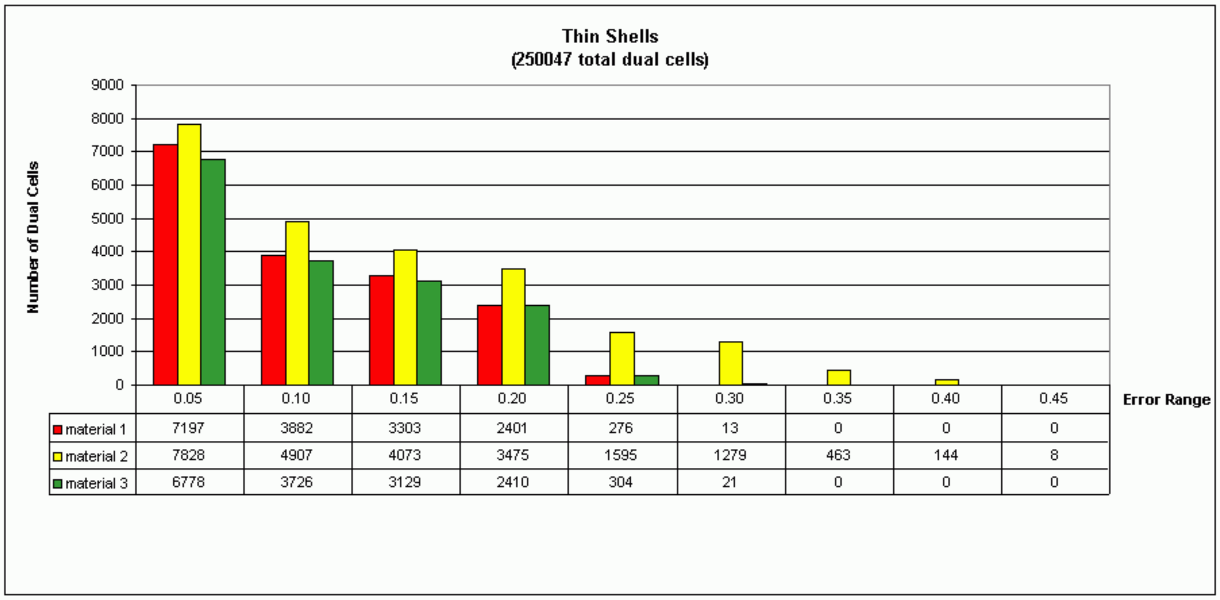


Figure 19: Volume fractions resulting from described algorithm.



Averaged Original: (0.165, 0.5625, 0.2725) Results: (0.096094, 0.742187, 0.161719) Difference: (0.068906, 0.179687, 0.110781)	Averaged Original: (0.17, 0.12, 0.71) Results: (0.065625, 0, 0.934375) Difference: (0.104375, 0.12, 0.224375)
Averaged Original: (0.545, 0.43, 0.025) Results: (0.59953, 0.40047, 0) Difference: (0.05453, 0.02953, 0.025)	Averaged Original: (0.555, 0.075, 0.37) Results: (0.71609, 0, 0.28391) Difference: (0.16109, 0.075, 0.08609)

Figure 20: Overall volume fractions. Given: (0.3111, 0.3033, 0.3855); Computed: (0.3693, 0.2857, 0.3450); Difference: (0.0582, 0.0176, 0.0405). The higher error differences on the per-cell basis were expected because mid-face intersections cause problems due to the dual representation.

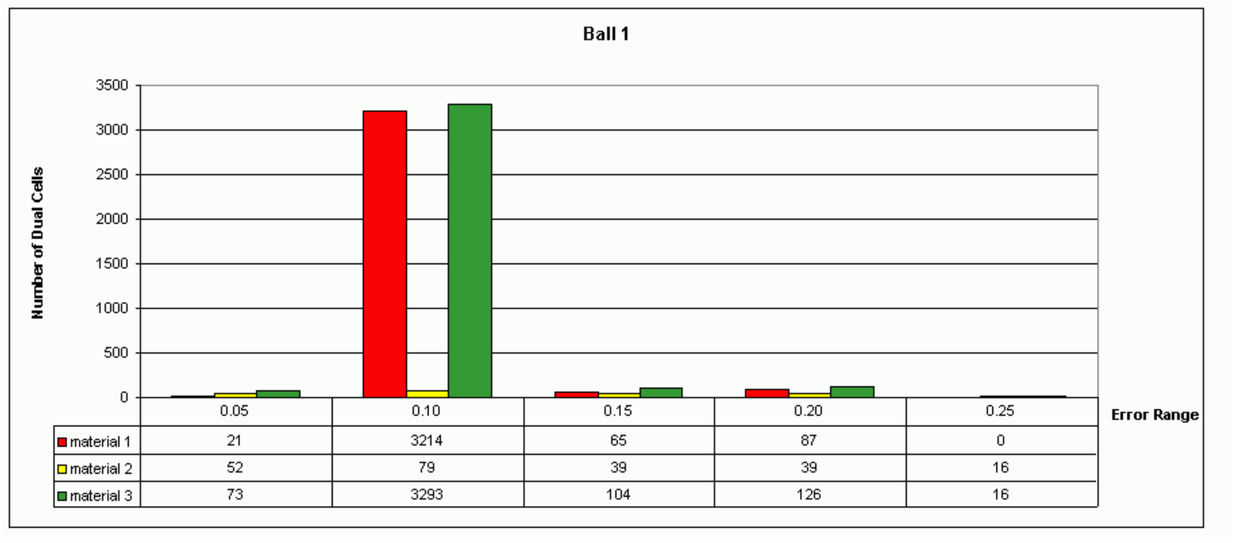


	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45
🟥 material 1	7197	3882	3303	2401	276	13	0	0	0
🟨 material 2	7828	4907	4073	3475	1595	1279	463	144	8
🟩 material 3	6778	3726	3129	2410	304	21	0	0	0

Figure 21: Error analysis for ”thin shells data set” shown in Figure 15.

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 22:

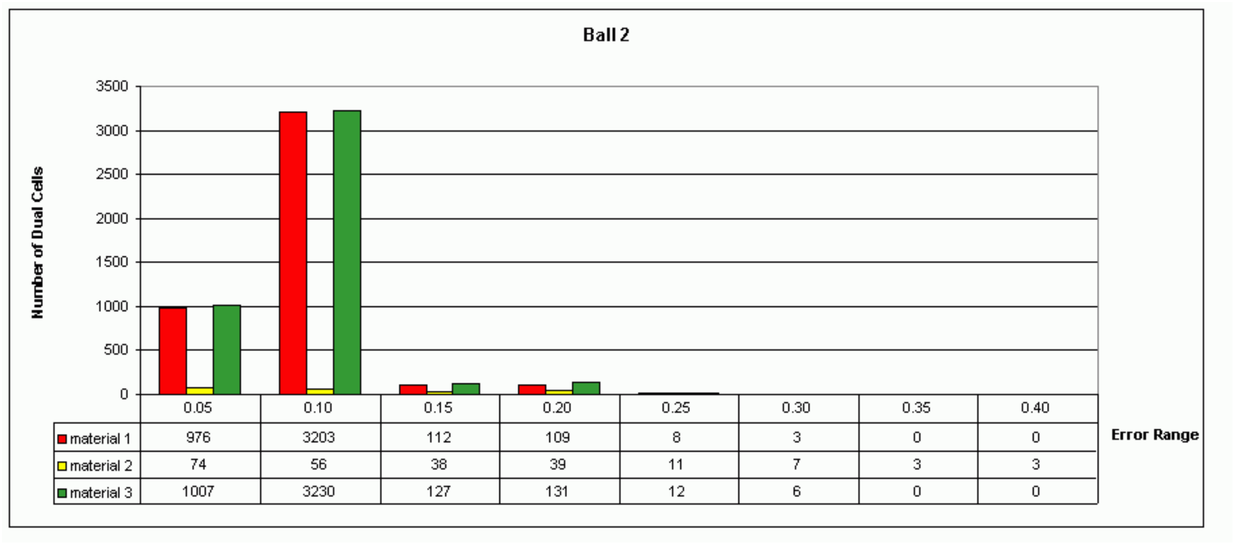


	0.05	0.10	0.15	0.20	0.25
🟥 material 1	21	3214	65	87	0
🟨 material 2	52	79	39	39	16
🟩 material 3	73	3293	104	126	16

Figure 23: Error analysis for first data set shown in Figure 16. The number of zero-error dual cells for materials 1, 2 and 3 are 21781, 24943, and 21556, respectively.

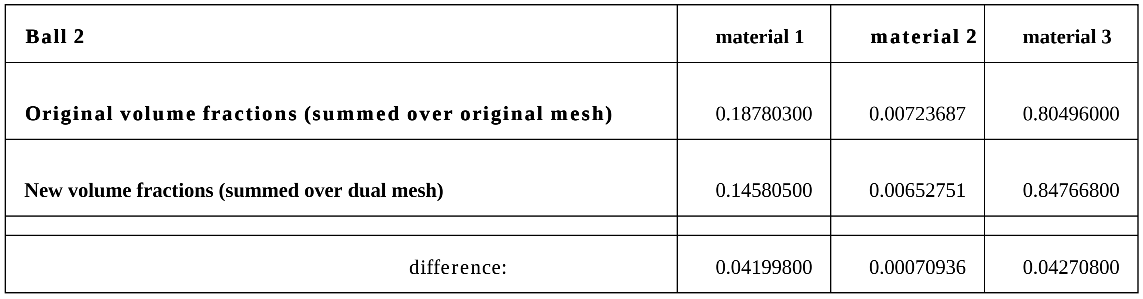
Ball 1	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.18864400	0.00778844	0.80356800
New volume fractions (summed over dual mesh)	0.14660600	0.00666681	0.84672800
difference:	0.04203800	0.00112163	0.04316000

Figure 24:



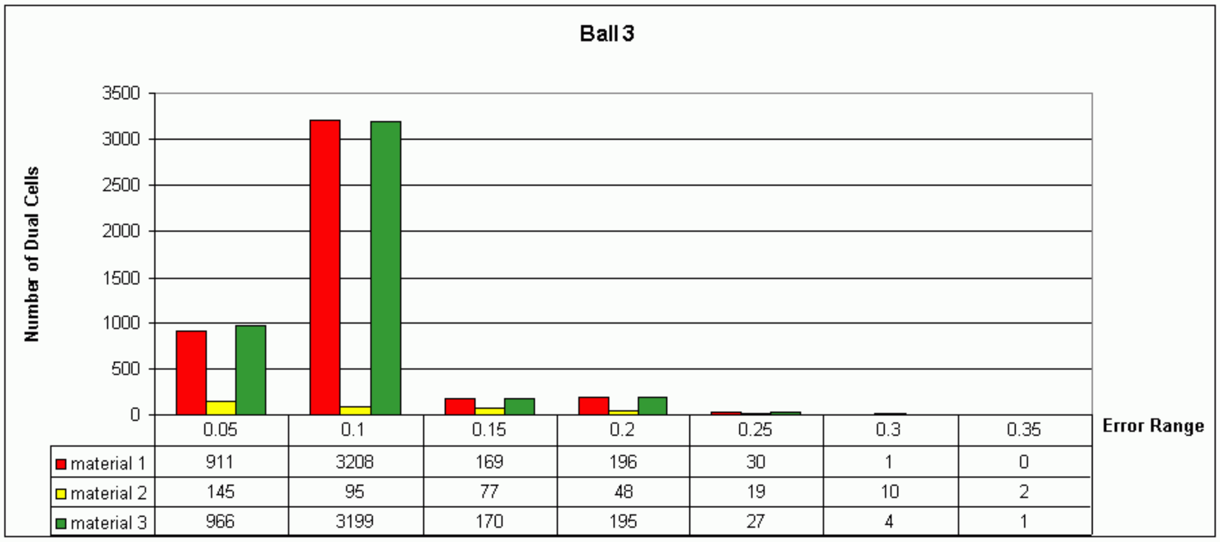
	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40
🟥 material 1	976	3203	112	109	8	3	0	0
🟨 material 2	74	56	38	39	11	7	3	3
🟩 material 3	1007	3230	127	131	12	6	0	0

Figure 25: Error analysis for second data set shown in Figure 16. The number of zero-error dual cells for materials 1, 2 and 3 are 20757, 24937, and 20655, respectively.



Ball 2	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.18780300	0.00723687	0.80496000
New volume fractions (summed over dual mesh)	0.14580500	0.00652751	0.84766800
difference:	0.04199800	0.00070936	0.04270800

Figure 26:

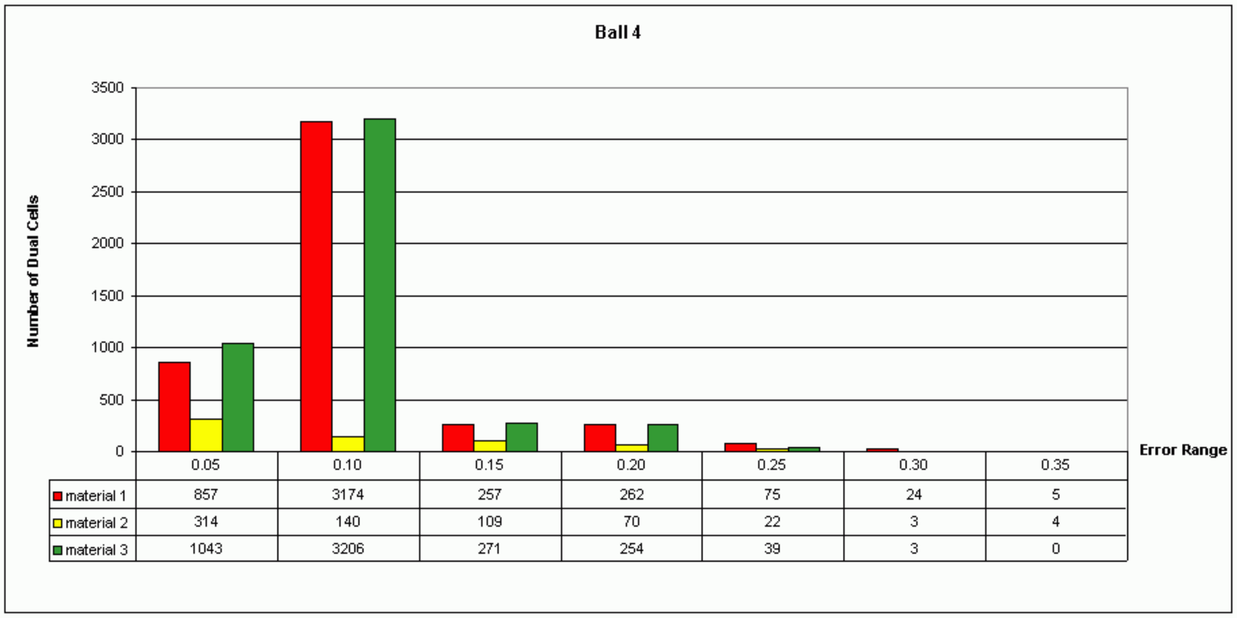


	0.05	0.1	0.15	0.2	0.25	0.3	0.35
🟥 material 1	911	3208	169	196	30	1	0
🟨 material 2	145	95	77	48	19	10	2
🟩 material 3	966	3199	170	195	27	4	1

Figure 27: Error analysis for third data set shown in Figure 16. The number of zero-error dual cells for materials 1, 2 and 3 are 20653, 24772, and 20655, respectively.

Ball 3	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.18778000	0.00695660	0.80526300
New volume fractions (summed over dual mesh)	0.14580900	0.00625933	0.84793100
difference:	0.04197100	0.00069727	0.04266800

Figure 28:

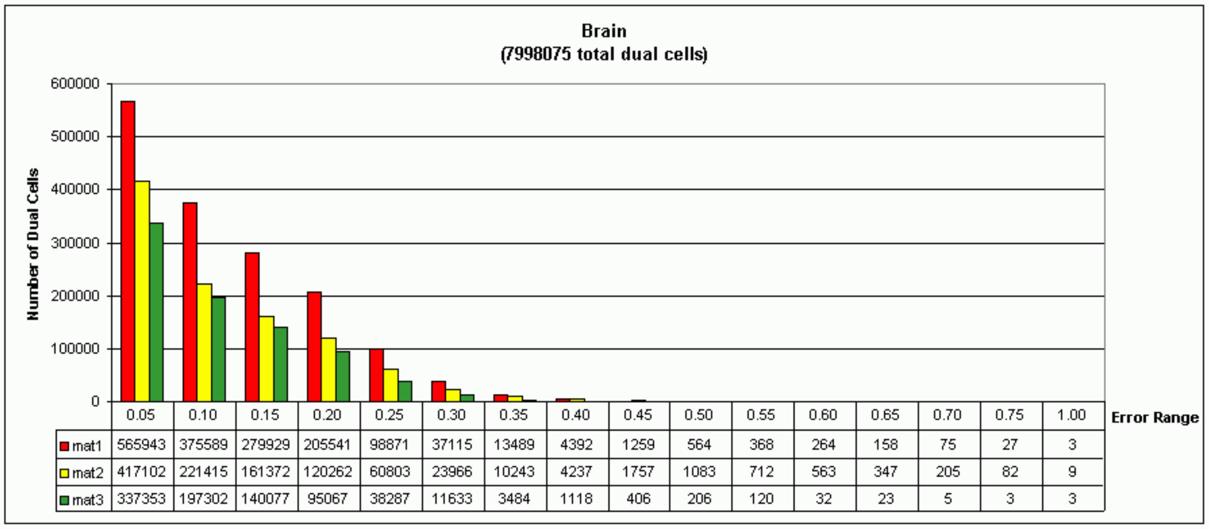


	0.05	0.10	0.15	0.20	0.25	0.30	0.35
🟥 material 1	857	3174	257	262	75	24	5
🟨 material 2	314	140	109	70	22	3	4
🟩 material 3	1043	3206	271	254	39	3	0

Figure 29: Error analysis for fourth data set shown in Figure 16. The number of zero-error dual cells for materials 1, 2 and 3 are 20514, 24772, and 20665, respectively.

Ball 4	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.1876770	0.0667801	0.8056450
New volume fractions (summed over dual mesh)	0.1461330	0.0054576	0.8484090
difference:	0.0415440	0.0613225	0.0427640

Figure 30:



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

error dual cells for materials 1, 2 and 3 are 6414488, 6973917, and 7172956, respectively. Figure 32 gives a comparison of the original and new volume fractions.

9. CONCLUSIONS

In this thesis, a new algorithm for material boundary surface reconstruction from data sets containing material volume-fraction information has been presented. A given grid is trans-

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 32:

formed to a dual grid, where each vertex has an associated barycentric coordinate that represents the fractions of each material present. After tetrahedrizing the dual grid, the material interfaces are constructed by mapping each tetrahedron to barycentric space, calculating the intersections with Voronoi cells in barycentric space. These intersection points are mapped back to physical space and triangulated to form the resulting boundary surface. The algorithm can treat any number of materials per cell, and since it is based on tetrahedral grids, it can be used with any grid structure.

Concerning future work, a "measure-and-adjust" feature could be added to the algorithm. Once an initial boundary surface approximation is calculated, the caluclation of (new) volume fractions for cells can be done directly from this boundary surface. This will enable the calculation of the difference between the original volume fractions and the volume fractions as implied by the initial boundary surface approximation. It is then possible to adjust material interfaces to minimize the volume fraction deviations. It may also be possible to adjust the material interface within each tetrahedron (or cell), in a manner similar to the works cited in Section2. If so, it will be possible to presrve volume fractions on a per-tetrahedron (per-cell) basis. It is planned to extend this algorithm to multidimensional grids.

10. ACKNOWLEDGMENTS

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract no. W-7405-Eng-48. It was also supported by the National Science Foundation under contracts ACI 9624034 and ACI 9983641 (CAREER Awards), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Office of Naval Research under contract N00014-97-1-0222; the Army Research Office under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; and the North Atlantic Treaty Organization (NATO) under contract CRG.971628 awarded to the University of California, Davis. I also acknowledge the support of ALSTOM Schilling Robotics, Chevron, General Atomics, Silicon Graphics, and ST Microelectronics, Inc. I would like to thank Randall Frank of the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory for supplying several of the data sets presented in this paper. I also 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.

References

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

[2] A. Okabe, B. Boots, and K. Sugihara, *Spatial Tesselations — Concepts and Applications of Voronoi Diagrams*. Chichester: Wiley, 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, 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, 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 flow with large fluid distortion,” in *Numerical Methods for Fluid Dynamics* (K. W. Morton and J. J. Baines, eds.), pp. 273–285, Academic Press, 1982.

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

[9] J. E. Pilliod and E. G. Puckett, “Second-order accurate volume-of-fluid 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.), (Los Alamitos), pp. 229–234, IEEE Computer Society Press, Oct. 1997.

[11] H. Samet, *The Design and Analysis of Spatial Data Structures*. Series in Computer Science, Reading, Massachusetts, U.S.A.: Addison-Wesley, reprinted with corrections ed., Apr. 1990.