Title
Multiresolution Representation of Datasets with Material Interfaces

Authors
Gregorski, Benjamin F.
Sigeti, David E.
Ambrosiano, J. J.
et al.

Publication Date
2003

Peer reviewed

MULTIRESOLUTION REPRESENTATION OF DATASETS WITH MATERIAL INTERFACES

Benjamin Gregorski and Kenneth I. Joy
Center for Image Processing and Integrated Computing (CIPIC)
Department of Computer Science
University of California, Davis

David E. Sigeti and John Ambrosiano and
Gerald Graham and Murray Wolinski
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Mark Duchaineau
Center for Applied Scientific Computing (CASC)
Lawrence Livermore National Laboratory

Abstract We present a new method for constructing multiresolution representations of data sets that contain material interfaces. Material interfaces embedded in the meshes of computational data sets are often a source of error for simplification algorithms because they represent discontinuities in the scalar or vector field over mesh elements. By representing material interfaces explicitly, we are able to provide separate field representations for each material over a single cell. Multiresolution representations utilizing separate field representations can accurately approximate datasets that contain discontinuities without placing a large percentage of cells around the discontinuous regions. Our algorithm uses a multiresolution tetrahedral mesh supporting fast coarsening and refinement capabilities; error bounds for feature preservation; explicit representation of discontinuities within cells; and separate field representations for each material within a cell.

Keywords: Material Interfaces, Data Simplification, Multiresolution Tetrahedral Meshes, Multiresolution Frameworks, Function Approximation

1. Introduction

Computational physics simulations are generating larger and larger amounts of data. They operate on a wide variety of input meshes, for example rectilinear meshes, adaptively refined meshes for Eulerian hydrodynamics, unstructured meshes for Lagrangian hydrodynamics and arbitrary Lagrange-Eulerian meshes. Often, these data sets contain special physical features such as material interfaces, physical boundaries, or thin slices of material that must be preserved when the field is simplified. In order to ensure that these features are preserved, the simplified version of the data set needs to be constructed using strict L^∞ error bounds that prevent small yet important features from being eliminated.

Data sets of this type require a simplification algorithm to approximate data sets with respect to several simplification criteria. The cells in the approximation must satisfy error bounds with respect to the dependent field variables over each mesh cell, and to the representation of the discontinuities within each cell. In addition, the simplification algorithm must be able to deal with a wide range of possible input meshes.

We present an algorithm for generating an approximation of a computational data set that can be used in place of the original high-resolution data set generated by the simulation. Our approximation is a resampling of the original data set that preserves user-specified as well as characteristic features in the data set and approximates the dependent field values to within a specified tolerance.

2. Related Work

Hierarchical approximation techniques for triangle meshes, scattered data, and tetrahedral meshes have matured substantially over recent years. A large amount of research has been done in the field of surface simplification. In [1] an iterative triangle mesh decimation method is introduced. Triangles in nearly linear regions are identified and collapsed to a point that is computed in a locally optimal fashion. In [2], Hoppe describes a progressive mesh simplification method for triangle meshes. An arbitrary mesh is simplified through a series of edge collapse operations to yield a very simple base mesh. Heckbert and Garland present a comprehensive survey of these techniques in [7]. In more recent work, Heckbert and Garland [4] use a quadric error metric for surface simplification. They use vertex pair collapse operations to simplify triangle meshes and they use *quadric matrices* that define a quadric object at each vertex to control the error of the simplified surfaces. In [8], they use the same technique to simplify meshes that have associated colors and textures. Hoppe [9] uses a modified quadric metric for simplify-

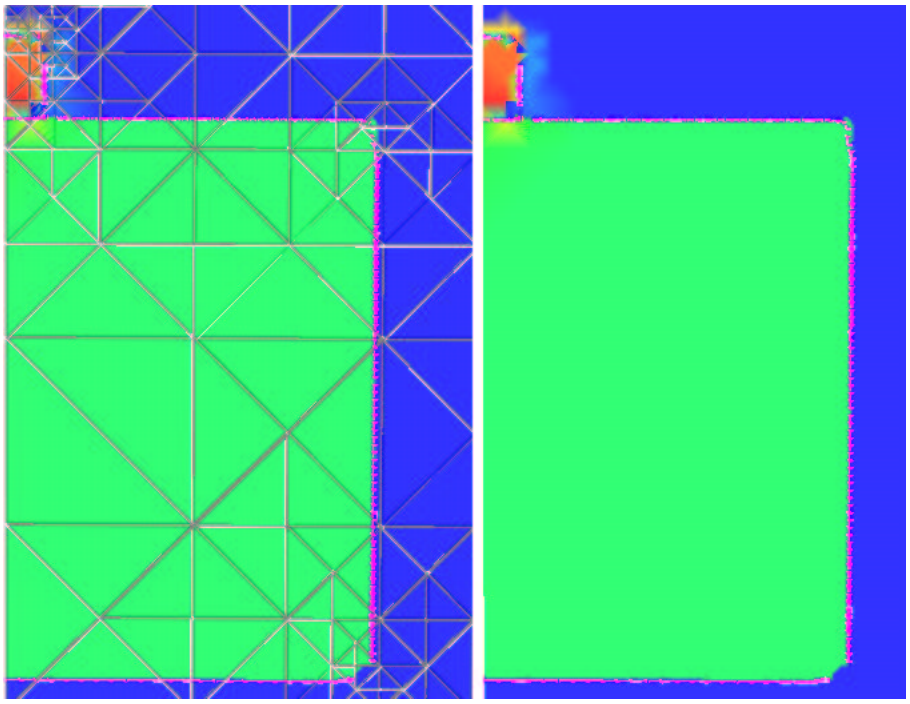

Figure 1. Density field using explicit interface representations and separate field representations. The left picture shows the field along with the approximating tetrahedral mesh. (interface error = 0.15).

ing polygonal meshes while preserving surface attributes such as normal vectors, creases, and colors. The ROAM system, described in [6], uses priority queue-driven split and merge operations to provide optimal real-time display of triangle meshes for terrain rendering applications. The tetrahedral mesh structure used in our framework is an extension of the original ROAM data structure for triangle meshes. Multiresolution methods for reconstruction and simplification have also been explored using subdivision techniques and wavelets.

The approximation of material interfaces is similar to the approximation or simplification of large polygonal meshes. The approximating mesh represents the large mesh to within a certain error tolerance using a substantially smaller number of triangles. Mesh approximation and simplification algorithms can be divided into decimation techniques and remeshing techniques.

Our approximation of material interfaces falls into the category of remeshing techniques. As described in Section 5, the material interfaces are given as triangle meshes. Within each of our cells we construct a piecewise linear approximation of the material interfaces to within a specified error tolerance based on the distance between the original mesh and our approximation.

3. Multiresolution Tetrahedral Mesh

The first basis for our simplification algorithm is the subdivision of a tetrahedral mesh as presented by Zhou et al. [3]. This subdivision scheme has an important advantage over other multiresolution spatial data structures such as an octree as it makes it easy to avoid introducing spurious discontinuities into representations of fields. The way we perform the binary subdivision ensures that the tetrahedral mesh will always be a conformant mesh, i.e, a mesh where all edges in the mesh end at the endpoints of other edges and not in the interior of edges. The simplest representation for a field within a tetrahedral cell is given by the unique linear function that interpolates field values specified at the cell's vertices. In the case of a conformant mesh, this natural field representation will be continuous across cell boundaries, resulting in a globally C^0 representation.

We have generalized the implementation presented by Zhou et al. by removing the restriction that the input data needs to be given on a regular rectilinear mesh consisting of $(2^N + 1)$x$(2^N + 1)$x$(2^N + 1)$ cells. A variety of input meshes can be supported by interpolating field values to the vertices of the multiresolution tetrahedral mesh. In general, any interpolation procedure may be used. In some cases, the procedure may be deduced from the physics models underlying the simulation that produced the data set. In other cases, a general-purpose interpolation algorithm will be appropriate.

We construct our data structure as a binary tree in a top-down fashion. Data from the input data set, including grid points and interface polygons, are assigned to child cells when their parent is split.

The second basis for our algorithm is the ROAM system, described in [6]. ROAM uses priority queue-driven split and merge operations to provide optimal real-time display of triangle meshes for terrain rendering applications. The tetrahedral mesh structure used in our framework can be regarded as an extension to tetrahedral meshes of the original ROAM data structure for triangle meshes.

Since our data structure is defined recursively as a binary tree, a representation of the original data can be pre-computed. We can utilize

the methods developed in ROAM to efficiently select a representation that satisfies an error bound or a desired cell count. This makes the framework ideal for interactive display. Given a data set and polygonal representations for the material interfaces, our algorithm constructs an approximation as follows:

1 Our algorithm starts with a base mesh of six tetrahedra and associates with each one the interface polygons that intersect it.

2 The initial tetrahedral mesh is first subdivided so that the polygonal surface meshes describing the material interfaces are approximated within a certain tolerance. At each subdivision, the material interface polygons lying partially or entirely in a cell are associated with the cell's children; approximations for the polygons in each child cell are constructed, and interface approximation errors are computed for the two new child cells.

3 The mesh is further refined to approximate the field of interest, for example density or pressure, within a specified tolerance.

For the cells containing material interfaces, our algorithm computes a field representation for each material. This is done by extrapolating *ghost* values for each material at the vertices of the tetrahedron where the material does not exist. A ghost value is an educated guess of a field value at a point where the field does not exist. When material interfaces are present, field values for a given material do not exist at all of the tetrahedron's vertices. Since tri-linear approximation over a tetrahedron requires four field values at the vertices, extra field values are needed to perform the interpolation. Thus for a given field and material, the ghost values and the existing values are used to form the tri-linear approximation within the tetrahedron.

This is illustrated in Figure 2 for a field sampled over a triangular domain containing two materials. For the field sampled at V_0 and V_1, a ghost value at vertex V_2 is needed to compute a linear approximation of the field over the triangle. The approximation is used only for those sample points that belong to this material. Given a function sampled over a particular domain, the ghost value computation extrapolates a function value at a point outside of this domain. When the field approximation error for the cell is computed, the separate field representations, built using these ghost values, are used to calculate an error for each distinct material in the cell. The decomposition process of a cell that contains multiple materials consists of these steps:

1 The signed distance values and ghost values for the new vertex are computed when the vertex is created during a split or subdivision

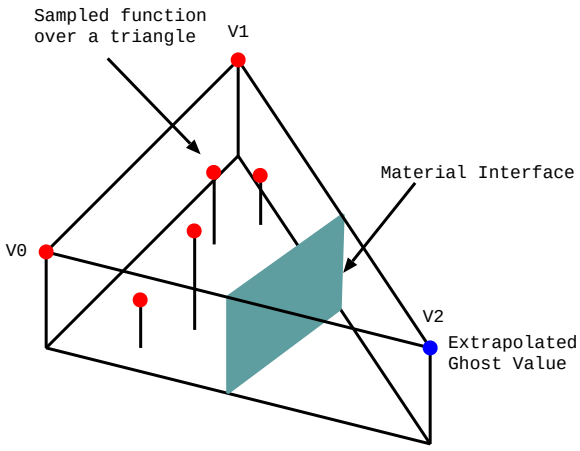

Figure 2. 2D Ghost Value Computation

operation. This is done by examining those cells that share the edge being split.

2 The interface representations, i.e., triangle meshes, are associated with the child cells, and the approximating representations of interfaces and their associated errors are computed.

3 The field error for each of the materials is computed, and the maximum value of these errors is defined as the overall error associated with a cell containing multiple materials.

4. General Multiresolution Framework

A multiresolution framework for approximating numerically simulated data needs to be a robust and extensible. The framework must be capable of supporting the wide range of possible input structures used in the simulations and the wide range of output data generated by these simulations. The following properties and operations are desirable for such a framework:

1 **Interactive transition between levels of detail.** The ability to quickly move between different levels of detail allows a user to select a desired approximation at which to perform a calculation (for a visualization application).

2 **Strict L^∞ error bounds.** Strict error bounds prevent small yet important features from being averaged or smoothed out by the approximation process.

3 **Local and adaptive mesh refinement and local error computations**. Local mesh refinement allows the representation to be refined only in the areas of interest. The error calculations for datasets consisting of millions or billions of cells should not involve a large amount of original data.

4 **Accommodating different mesh types**. Computational simulations are done on a large variety of mesh structures. In order for a framework to be useful it should be easily adaptable to a broad class of input meshes.

5 **Explicit representation of field and/or material discontinuities**. Discontinuities are very important in scientific datasets and very often need to be preserved when the datasets are approximated. A multiresolution framework should support the explicit representation and approximation of these discontinuities.

Our multiresolution recursive tetrahedral framework satisfies these design criteria. Tetrahedral cells allow us to use linear basis functions to approximate the material interfaces and the dependent field variables in a cell. A representation of the original data can be computed in a pre-processing step, and we can utilize methods developed for the ROAM system [6] to efficiently select a representation that satisfies an error bound or a desired cell count. This makes the framework ideal for interactive display. Strict L^∞ error bounds are incorporated into the refinement process.

The framework supports various input meshes by resampling them at the vertices of the tetrahedral mesh. The resampling error of the tetrahedral mesh is a user specified variable. The resampled mesh can be refined to approximate the underlying field to within a specified tolerance or until the mesh contains a specific number of tetrahedra. The resampled field is a linear approximation of the input field. The boundaries of the input mesh are represented in the same manner as the surfaces of discontinuity. The volume of space outside of the mesh boundary is considered as a separate material with constant field values. This region of *empty* space is easy to evaluate and approximate; no ghost values and no field approximations need to be computed. This allows a non-rectilinear input mesh to be embeded into the larger rectilinear mesh generated by the refinement of the tetrahedral grid. Discontinuities are supported at the cell level allowing local refinement of the representations of surfaces of discontinuity in geometrically complex areas. The convergence of the approximation depends upon the complexity of the input field and the complexity of the input mesh.

The framework has several advantages over other multiresolution spatial data structures such as an octree. The refinement method ensures that the tetrahedral mesh will always be free of cracks and *T-intersections*. This makes it easy to guarantee that representations of fields and surfaces of discontinuity are continuous across cell boundaries.

Our resampling and error bounding algorithms require that an original data set allow the extraction of the values of the field variables at any point and, for a given field, the maximum difference between the representation over one of our cells and the representation in the original dataset over the same volume.

5. Material Interfaces

A material interface defines the boundary between two distinct materials. Figure 3 shows an example of two triangles crossed by an single interface (smooth curve). This interface specifies where the different materials exist within a cell. Field representations across a material interface are often discontinuous. Thus, an interface can introduce a large amount of error to cells that cross it. Instead of refining an approximation substantially in the neighborhood of an interface, the discontinuity in the field is better represented by explicitly representing the surface of discontinuity in each cell. Once the discontinuity is represented, two separate functions are used to describe the dependent field variables on either side of the discontinuity. By representing the surface of discontinuity exactly, our simplification algorithm does not need to refine regions in the spatial domain with a large number of tetrahedra.

5.1 Extraction and Approximation

In the class of input datasets with which we are working, material interfaces are represented as triangle meshes. In the case that these triangle meshes are not known, they are extracted from volume fraction data by a material interface reconstruction technique, see [5]. Such an interface reconstruction technique produces a set of crack-free triangle meshes and normal vector information that can be used to determine on which side and in which material a point lies.

Within one of our tetrahedra, an approximate material interface is represented as the zero set of a signed distance function. Each vertex of a tetrahedron is assigned a signed distance value for each of the material interfaces in the tetrahedron. The signed distance from a vertex $\mathbf{V}$ to an interface mesh $\mathbf{I}$ is determined by first finding a triangle mesh vertex $\mathbf{V}_i$ in the triangle mesh describing $\mathbf{I}$ that has minimal distance to $\mathbf{V}$. The sign of the distance is determined by considering the normal

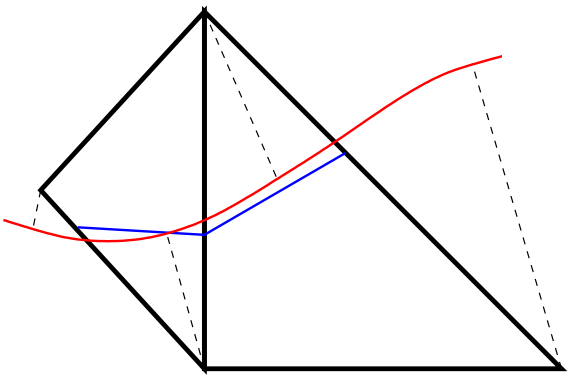

Figure 3. True and approximated interfaces.

vector $\mathbf{N}_i$ at $\mathbf{V}_i$. If $\mathbf{N}_i$ points towards $\mathbf{V}$, then $\mathbf{V}$ is considered to be on the *positive side* of the interface; otherwise it is considered to be on the *negative side* of the interface. The complexity of this computation is proportional to the complexity of the material interfaces within a particular tetrahedra. In general a coarse cell in the mesh will contain a large number of interface polygons, and a fine cell in the mesh will contain a small number of interface polygons. The signed distance values are computed as the mesh is subdivided. When a new vertex is introduced via the mesh refinement, the computation of the signed distance for that vertex only needs to look at the interfaces that exist in the tetrahedra around the split edge. If those tetrahedra do not contain any interfaces, no signed distance value needs to be computed.

In Figure 3, the true material interface is given by the smooth curve and its approximation is given by the piecewise linear curve. The minimum distances from the vertices of the triangles to the interface are shown as dotted lines. The distances for vertices on one side of the interface (say, above the interface) are assigned positive values and those on the other side are assigned negative values. These signed distance values at the vertices determine linear functions in each of the triangles, and the approximated interface will be the zero set of these linear functions. Because the mesh is conformant, the linear functions in the two triangles will agree on their common side, and the zero set is continuous across the boundary. The situation in three dimensions is analogous.

The signed distance function is assumed to vary linearly in the cell, i.e., a tetrahedron. The coefficients for the linear function defining a boundary representation are found by solving a 4x4 system of equations, considering the requirement that the signed distance function over the

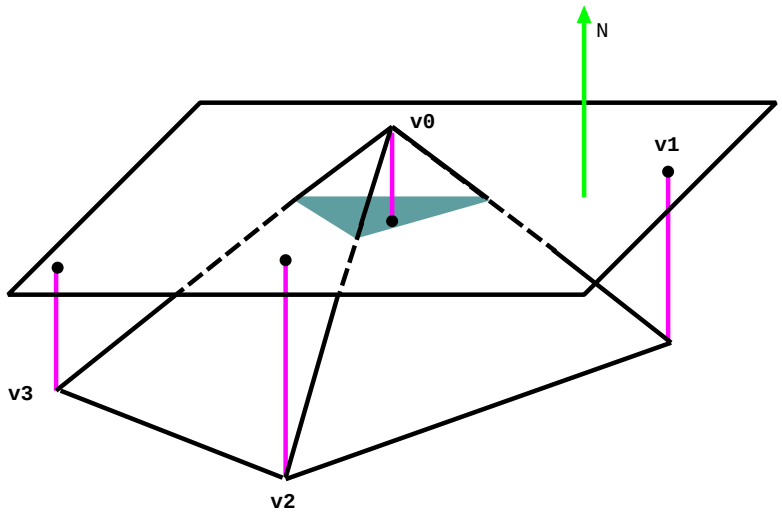

Figure 4. Tetrahedron showing signed distance values and the corresponding boundary approximation.

tetrahedron must interpolate the signed distance values at the four vertices.

The three-dimensional example in Figure 4 shows a tetrahedron, a material interface approximation, and the signed distance values d_i for each vertex V_i. The approximation is shown as a plane cutting through the tetrahedron. The normal vector N indicates the positive side of the material boundary approximation. Thus, the distance to V_0 is positive and the distances for V_1, V_2, and V_3 are negative.

We note that a vertex has at most one signed distance value for each interface. This ensures that the interface representation is continuous across cell boundaries. If a cell does not contain a particular interface, the signed distance value for that interface is meaningless for that cell. Given a point **P** in an interface polygon and its associated approximation B_r, the error associated with **P** is the absolute value of the distance between **P** and B_r. The material interface approximation error associated with a cell is the maximum of these distances, considering all the interfaces within the cell.

6. Discontinuous Field Representations

Cells that contain material interfaces typically also have discontinuities in the fields defined over them. For example, the density field over a cell that contains both steel and nickel is discontinuous exactly where

the two materials meet. In these situations, it is better to represent the density field over the cell as two separate fields, one field for the region containing only the first material and one for the second material. Our algorithm represents the discontinuity by constructing a field representation for each material in the cell. Each of the vertices in a cell must have distinct field values for each material in the cell. These extrapolated values are the ghost values.

For a vertex **V** that does not reside in material **M**, we compute a ghost value for the field associated with material **M** at vertex **V**. This ghost value is an extrapolation of the field value for **M** at **V**. As described in Section 3, the ghost value computation is performed when the vertex is created during the tetrahedral refinement process.

6.1 Computation of Ghost Values

The ghost values for a vertex **V** are computed as follows:

1 For each material interface present in the cells that share the vertex, find a vertex V_{min} in a triangle mesh representing an interface with minimal distance to **V**.

2 Evaluate the data set on the far side of the interface at V_{min} and use this as the ghost value at **V** for the material on the opposite side of the interface.

Only one ghost value exists for a given vertex, field and material. This ensures that the field representations are C^0 continuous across cell boundaries.

7. Error Metrics

The error metrics employed in our framework are similar to the nested error bounds used in the ROAM system. Each cell has two associated error values: a field error and a material interface error. In order to calculate the field errors for a leaf cell in our tetrahedral mesh hierarchy, we assume that the original data set can be divided into *native data elements* that are grid points of zero volume. For a given field, we assume that it is possible to bound the difference between the representation over one of our leaf cells and the representation over each of the native data elements with which the given cell intersects. The error for the given field in the given cell is the maximum of the errors associated with each of the intersecting native data elements.

The field error e_T for a non-leaf cell is computed from the errors associated with its two children according to:

$$e_T = \max\{e_{T_0}, e_{T_1}\} + |z(v_c) - z_T(v_c)|, \tag{1}$$

where e_{T_0} and e_{T_1} are the errors of the children; v_c is the vertex that splits the parent into its children; $z(v_c)$ is the field value assigned to v_c; and $z_T(v_c)$ is the field value that the parent assigns to the spatial location of v_c. The approximated value at v_c, $z_T(v_c)$, is calculated as:

$$z_T(v_c) = \frac{1}{2}(z(v_0) + z(v_1)), \tag{2}$$

where v_0 and v_1 are the vertices of the parent's split edge. This error is still a genuine bound on the difference between our representation and the representation of the original data set. However, it is looser than the bound computed directly from the data. The error computed from the children has the advantage that the error associated with a cell bounds not only the deviation from the original representation but also the deviation from the representation at any intermediate resolution level. Consequently, this error is *nested* or monotonic in the sense that the error of a child is guaranteed not to be greater than the error of the parent. Once the errors of the leaf cells are computed, the nested bound for all cells higher in the tree can be computed in time proportional to K, where K is the number of leaf cells in the tree. This can be accomplished by traversing the tree in a bottom-up fashion.

The material interface error associated with a leaf node is the maximum value of the errors associated with each of the material interfaces in the node. For each material interface, the error is the maximum value of the errors associated with the vertices constituting the triangle mesh defining the interface and being inside the cell. The error of a vertex is the absolute value of the distance between the vertex and the interface approximation. The material interface error **E** for a cell guarantees that no point in the original interface polygon mesh is further from its associated approximation than a distance of **E**. This error metric is an upper bound on the deviation of the original interfaces from our approximated interfaces. A cell that does not contain a material interface is considered to have an interface error of zero.

8. Results

We have tested our algorithm on a data set resulting from a simulation of a hypersonic impact between a projectile and a metal block. The simulation was based on a logically rectilinear mesh of dimensions 32x32x52.

For each cell, the average density and pressure values are available, as well as the per-material densities and volume fractions. The physical dimensions in x, y, and z directions are [0,12] [0,12] and [-16,4.8].

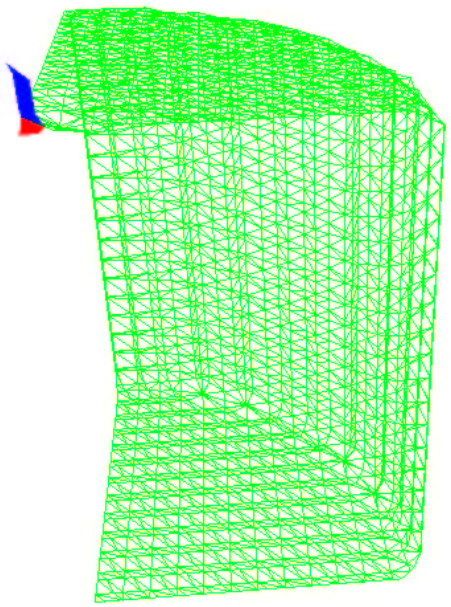

Figure 5. Original triangular meshes representing material interfaces.

There are three materials in the simulation: the projectile, the block, and *empty* space. The interface between the projectile and the block consists of 38 polygons, the interface between the projectile and empty space consists of 118 polygons and the interface between empty space and the block consists of 17574 polygons. Figure 5 shows the original interface meshes determined from the volume fraction information. The largest mesh is the interface between the metal block and empty space; the next largest mesh in the top, left, front corner is the interface between the projectile and empty space; the smallest mesh is the interface between the projectile and the block.

Figure 6 shows a cross-section view of the mesh created by a cutting plane through the tetrahedral mesh. The darker lines are the original interface polygons, and the lighter lines are the approximation to the interface. The interface approximation error is 0.15. (An error of 0.15 means that all of the vertices in the original material interface meshes are no more that a physical distance of 0.15 from their associated inter-

face approximation. This is equivalent to an error of (0.5 - 1.5)% when considered against the physical dimensions.) A total of 3174 tetrahedra were required to approximate the interface within an error of 0.15. The overall mesh contained a total of 5390 tetrahedra. A total of 11990 tetrahedra were required to approximate the interface to an error of 0.15 and the density field within an error of 3. The maximum field approximation error in the cells containing material interfaces is 2.84, and the average field error for these cells is 0.007. These error measurements indicate that separate field representations for the materials on either side of a discontinuity can accurately reconstruct the field.

Figures 6 and 1 compare the density fields generated using linear interpolation of the density values and explicit field representations on either side of the discontinuity. These images are generated by intersecting the cutting plane with the tetrahedra and evaluating the density field at the intersection points. A polygon is draw through the intersection points to visualize the density field. In the cells where material interfaces are present, the cutting plane is also intersected with the interface representation to generate data points on the cutting plane that are also on the interface. These data points are used to draw a polygon for each material that the cutting plane intersects.

Figure 1 shows that using explicit field representations in the presence of discontinuities can improve the quality of the field approximation. This can be seen in the flat horizontal and vertical sections of the block where the cells approximate a region that contains the block and empty space. In these cells, the use of explicit representations of the discontinuities leads to an exact representation of the density field. The corresponding field representations using linear interpolation, shown in Figure 6, capture the discontinuities poorly. Furthermore, Figure 1 captures more of the dynamics in the area where the projectile is entering the block (upper-left corner). The linear interpolation of the density values in the region where the projectile is impacting the block smooths out the density field, and does not capture the distinct interface between the block and the projectile.

9. Conclusions and Future Work

We have presented a method for constructing multiresolution representations of scientific datasets that explicitly represents material interfaces. Our algorithm constructs an approximation that can be used in place of the original data for visualization purposes. Explicitly representing material and implicit field discontinuities allows us to use multiple field representations to better approximate the field within each cell. The

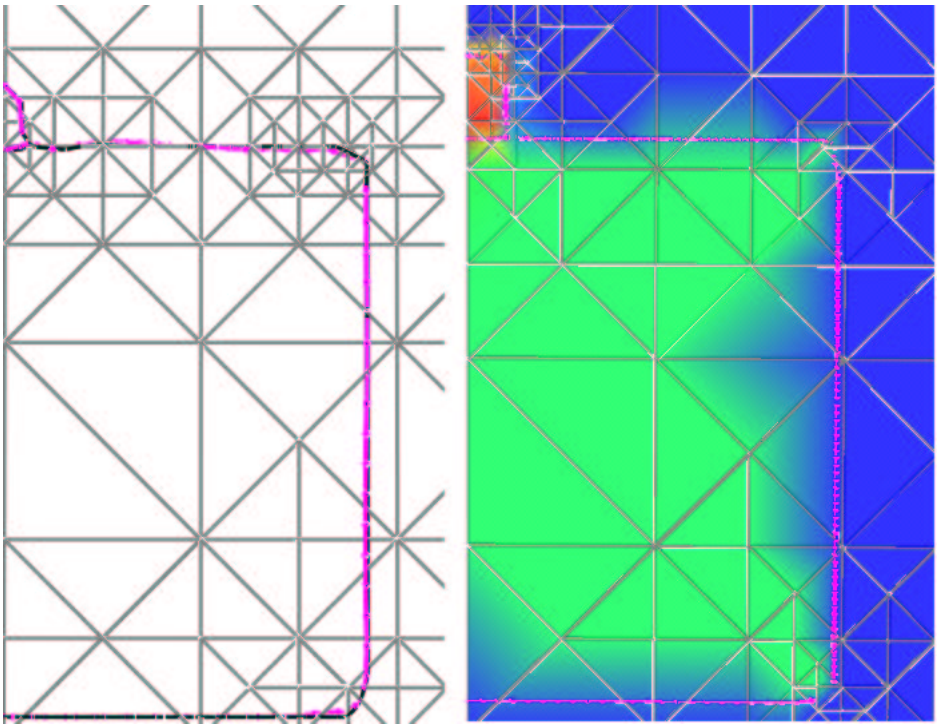

Figure 6. Cross section of the tetrahedral mesh. The left picture shows the original interfaces and their approximations. The picture on the right shows density field using linear interpolation.

use of the tetrahedral subdivision allows us to generalize our algorithm to a wide variety of data sets and to support interactive level-of-detail exploration and view-dependent simplification.

Acknowledgments

This work was supported by, Los Alamos National Laboratory and by the National Science Foundation under contract ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 awarded to the University of California, Davis. We thank the members of the Visualization Group at the Center for Image Processing and Integrated Computing

(CIPIC) at the University of California, Davis and the members of the Advanced Computing Laboratory at Los Alamos National Laboratory.

References

[1] B. Hamann. A data reduction scheme for triangulated surface, *Computer Aided Geometric Design*, 11:197-214.

[2] Hugues Hoppe. Progressive Meshes, *SIGGRAPH 96 Conference Proceedings*, 1996:99–108

[3] Yong Zhou and Baoquan Chen and Arie E. Kaufman. Multiresolution Tetrahedral Framework for Visualizing Regular Volume Data, IEEE Visualization '97, 135–142.

[4] Michael Garland and Paul S. Heckbert. Surface Simplification Using Quadric Error Metrics, SIGGRAPH 97 Conference Proceedings, 209-216.

[5] Kathleen S. Bonnell and Daniel R. Schikore and Kenneth I. Joy and Mark Duchaineau and Bernd Hamann. Constructing Material Interfaces From Data Sets With Volume-Fraction Information, Proceedings Visualization 2000, 367–372.

[6] M. A. Duchaineau and M. Wolinsky and D. E. Sigeti and M. C. Miller and C. Aldrich and M. B. Mineev-Weinstein. ROAMing Terrain: Real-time Optimally Adapting Meshes, IEEE Visualization 1997, 81–88.

[7] Paul S. Heckbert and Michael Garland. Survey of Polygonal Surface Simplification Algorithms

[8] M. Garland and P. S. Heckbert, Simplifying surfaces with color and texture using quadric error metrics, IEEE Visualization 1998, 263–270

[9] Hugues H. Hoppe. New Quadric Metric for Simplifying Meshes with Appearance Attributes, IEEE Visualization 1999, 59–66.