Particle-based Sampling and Meshing of Surfaces in Multimaterial Volumes

Miriah Meyer, Ross Whitaker, *Member, IEEE*, Robert M. Kirby, *Member, IEEE*,
Christian Ledergerber, and Hanspeter Pfister, *Senior Member, IEEE*

Abstract— Methods that faithfully and robustly capture the geometry of complex material interfaces in labeled volume data are important for generating realistic and accurate visualizations and simulations of real-world objects. The generation of such multimaterial models from measured data poses two unique challenges: first, the surfaces must be well-sampled with regular, efficient tessellations that are consistent across material boundaries; and second, the resulting meshes must respect the nonmanifold geometry of the multimaterial interfaces. This paper proposes a strategy for sampling and meshing multimaterial volumes using dynamic particle systems, including a novel, differentiable representation of the material junctions that allows the particle system to explicitly sample corners, edges, and surfaces of material intersections. The distributions of particles are controlled by fundamental sampling constraints, allowing Delaunay-based meshing algorithms to reliably extract watertight meshes of consistently high-quality.

Index Terms—Sampling, meshing, visualizations.

1 Introduction

Three dimensional images (volumes) provide an important source of information for generating realistic computer models of real-world objects. For example, biological and geophysical data is often captured using volumetric scanning methods such as magnetic resonance imaging (MRI) or ultrasound. The data from these devices is usually stored as a regular grid of values that provide information about the surface of the scanned object and its detailed internal structure. Most objects, natural or man-made, contain multiple materials with vastly different physical properties that are typically organized in complicated geometric configurations. Extracting precise geometric models of the interfaces between these materials is important both for visualization and for realistic physically-based simulations in a variety of fields, from biomedical computing and computer animation to oil-and-gas exploration and engineering.

Multimaterial volumes impose particular challenges for sampling and meshing algorithms because the boundaries between materials are typically not smooth manifolds. As a result, intersections of materials can produce sharp features such as edges and corners (see Section 3). Furthermore, the development of increasingly realistic simulations dictates additional constraints, such as a sufficient number of samples for accurately representing the geometry, compact sets of nearly-regular triangles, and consistent tessellations across material boundaries. The construction of geometric models that meet these requirements for surfaces of distinct objects is well-studied. However, generating high-quality models of objects that contain multiple materials has thus far received little attention.

This paper proposes the use of a dynamic particle system to produce well-spaced distributions of points on material interfaces in multimaterial volumes. The particles move to minimize an objective function that is designed to produce configurations of samples that are locally adaptive, geometrically accurate, and well-suited for subsequent meshing. The set of material boundaries are described as a CW-complex [26], and we present new, analytic representations for the different kinds of cells that form this structure. We also define projection operators that allow the particles to sample these material boundaries in a hierarchical fashion — 0-cells, 1-cells, and then 2-cells. The result is a set of surface points that adapt to the underlying geometry and meet fundamental surface sampling requirements. From these surface points, Delaunay-based meshing schemes create high-quality surface meshes that are well-suited to the generation of well-shaped volumetric elements [3].

The main contribution of the paper is a novel scheme for representing the nonmanifold sets formed at the material interfaces in multimaterial volume data, and a corresponding set of projection operators that allow these interfaces to be sampled with dynamic particle systems. Also presented is an algorithm for distributing sets of particle systems such that each type of interface is sampled explicitly. The resulting sets of points generate surface meshes that are well-defined subsets of a Delaunay tetrahedralization, the extraction of which is reliably produced through a simple labeling algorithm. This scheme generates high-quality meshes *directly* from the multimaterial representation, and thus part of the pipeline is a volume processing methodology that allows for controlled smoothing of the data for a specific downstream application prior to sampling and meshing (see Section 5.1) — this is a significant difference from existing grid-based methods (see Section 2.1). We provide implementation details of the proposed algorithm, and demonstrate its effectiveness for several multimaterial volumes generated from MRI scans of real-world objects, such as the torso in Figure 1.

2 Previous Work

2.1 Meshing Multimaterial Volumes

Most of the previous work on meshing multiple material data focuses on grid-based tessellation algorithms. These algorithms work on the original, labeled volumes (*i.e.*, a regular grid that stores an integer at each node that indicates a specific material, most often the result of a segmentation of an MRI or CT scan), and focus on extensions of the marching cubes case tables [27] to handle the nonmanifold surface intersections. A postprocessing step is often applied to the mesh to reduce the voxelization artifacts, as well as to reduce the overall number of triangles.

Some of the earliest work presents methods that generate nonmanifold meshes from tetrahedral elements that are created from the original rectilinear volume [8, 33, 11], including the recent method of Dillard *et al.* [18] for creating boundary meshes of polycrystal data. Surface extractions from tetrahedral elements, however, result in an

- *M. Meyer, C. Ledergerber, and H. Pfister are with the Initiative in Innovative Computing at Harvard University, E-mail: miriah,ledergerber,pfister@seas.harvard.edu.*
- *R. Whitaker and R.M. Kirby are with the Scientific Computing and Imaging Institute at the University of Utah, E-mail: whitaker,kirby@sci.utah.edu.*
- *M. Meyer accomplished this work while at the Scientific Computing and Imaging Institute at the University of Utah.*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mailto:tvcg@computer.org.

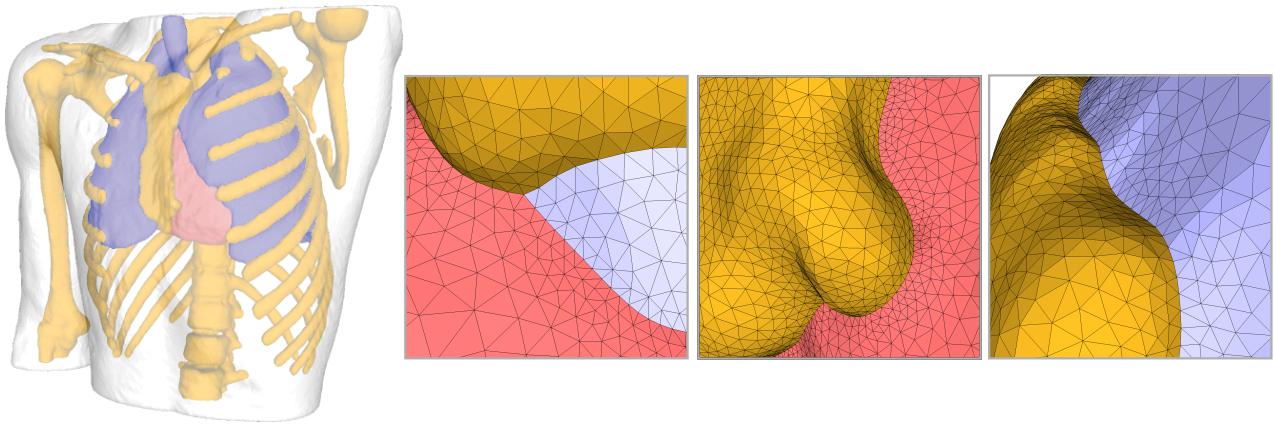


Fig. 1. Multimaterial surfaces of a torso extracted from an MRI scan, with closeups of meshes generated using dynamic particle systems.

excessive number of triangles. Methods based on extracting interfaces in hexahedral cells can reduce the overall number of triangles, but they must handle an increased number of nonmanifold ambiguities [22, 47, 39, 6, 7].

While these grid-based approaches are generally robust and efficient, the first-order approximation of the material boundaries can construct incorrect topology [34]. More importantly, the resolution of the underlying grid, and not the geometry of the surface, dictates the resulting mesh resolution, while the triangles are usually poorly shaped (*i.e.*, not-regular) because the methods do not focus on the placement of the vertices. Even with smoothing and decimation, the quality is limited (see Section 6). Recently, Zhang *et al.* [48] proposed an octree-based approach that relies on the dual contouring method [23] to produce adaptive tetrahedral elements as well as to preserve sharp features. Like the other grid-based methods, however, this approach does not optimize the mesh resolution to reflect the underlying surface geometry.

Another avenue of surface meshing research focuses exclusively on the boundary of objects and the generation of well-shaped triangles. Methods such as advancing front [24, 40] or particle-based schemes [16, 31] place samples along an implicit surface such that nearly-regular triangles are generated that adapt to the underlying geometry. A different approach is that taken by Delaunay-based methods [5, 17], which provide algorithms for constructing provably-correct triangulations of a given set of surface samples. These algorithms rely on the seminal work of Amenta *et al.* [4] that proves a sampling constraint based on the *local feature size* (LFS) such that the correct topology of a surface can be inferred from an unorganized set of sample points. These methods do not, however, address the challenges of meshing multimaterial surfaces or handling surfaces with sharp features.

Recently, Pons *et al.* [38] apply a Delaunay-refinement algorithm [9] to multimaterial volumes. The algorithm described in their paper builds on the work of Oudot *et al.* [36] that requires C^2 surfaces for termination guarantees, which is not the case for multimaterial datasets as discussed in the next section. Some recent work on the description of the LFS [10, 12] and the Delaunay-refinement algorithm [14] may, however, provide some theoretical foundations that extend to multimaterial datasets.

2.2 Dynamic Particle Systems

Dynamic particle systems, first introduced to the computer graphics community by Witkin and Heckbert [46], are a mechanism for producing even distributions of samples across implicitly-defined surfaces. Surface samples (called *particles*) are constrained to the zero-set of an implicit function, but are allowed to move along the surface (*i.e.*, in the local tangent plane) to obtain minimal energy configurations. In this work, we use a modified version of the particle system proposed by Meyer *et al.* [30] that locally adapts the distance between particles to create higher densities of points around surface features.

2.3 Multimaterial Representations

Work on level sets and CSG has addressed the problem of representing multimaterial data analytically. Level set methods [41, 35], although often used for capturing and deforming a single interface, have been extended for tracking the motion of multimaterial datasets [28, 49], while implicit representations of multimaterials have also been developed for handling CSG operations [37]. These algorithms describe the multimaterial datasets using a functional model of the materials. In this paper we extend the model so that a dynamic particle system can explicitly capture different types of multimaterial junctions to produce high-quality triangle meshes.

3 Topology of Multimaterial Interfaces

We represent interfaces in a multimaterial dataset using a model that describes each material with a smooth, volumetric *indicator function*, f_i [28, 37]. A set of N indicator functions $F = \{f_i | f_i : V \mapsto \Re\}$ represents n materials. A material label i is assigned to a point $x \in V$ if (and only if) $f_i(x) > f_j(x) \;\; \forall \;\; j \neq i$.

Looking at the simple case when only two materials exist in the dataset, for all points x where $f_1(x) - f_2(x) > 0$ the model will assign a label of 1, while assigning a label of 2 otherwise — we note that this description corresponds to the conventional formulation of an implicit surface. In the multimaterial model, the set of points x, where $f_1(x) - f_2(x) = 0$, forms the interface, or *junction*, between these two materials.

In the case of an arbitrary number of materials in the dataset, the configurations of interfaces become somewhat more complex. The boundaries that separate materials are no longer manifold, and can form sharp corners and edges. The topology of these junctions, however, can be characterized by certain *generic* configurations (see Figure 2). The term *generic*, from the field of singularity theory, refers

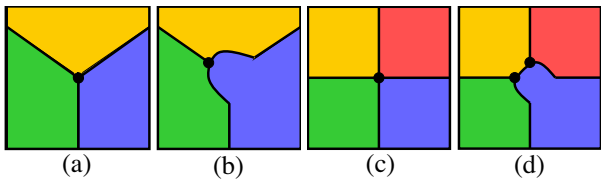


Fig. 2. In 2D, a 3-material junction is *generic* and forms a 0-cell (a); it maintains its topology under small perturbations (b). A 4-material junction (c), however, is a nongeneric case, and is annihilated under small perturbations (d) to form generic 2- and 3- material junctions.

to the cases where the set of functions F are in *general position*. This situation is analogous to the finite-dimensional spaces considered in discrete geometry — *i.e.*, three points in general position cannot lie on a line, and if they do, a general position can be restored through very

small perturbations. In this work we typically consider only generic configurations, which is justified by our reliance on measured data that inherently contains some level of noise, as well as our use of a data processing pipeline that ensures a degree of smoothness in the indicator functions (see Section 5.1).

We characterize each material interface in terms of the number of material indicator functions that are maximal (and equal) at that junction. For $V \subset \Re^2$, 2-junctions and 3-junctions occur generically, as shown in Figure 2 (a-b) as lines and points, respectively, while a 4-junction is a nongeneric case, as shown in Figure 2 (c-d). For $V \subset \Re^d$ each K-junction forms a subset of V that is topologically equivalent (homeomorphic) to a P-disk, where $P = d - K + 1$. Thus each type of material junction can be considered a P-cell, as described in the literature on discrete topology [21, 2]. Generically, for $d = 3$ we have 4-junctions, which are 0-cells or points; 3-junctions, which are 1-cells or curves; and 2-junctions, which are 2-cells or surfaces.

The collection of cells that describe the different types of material junctions, taken together, form a CW-complex. That is, we can organize them hierarchically, such that each 2-cell is attached to a collection of 1-cells (at its border), and each one cell is attached to one or more 0-cells. The strategy in this paper is to sample nonmanifold multimaterial boundaries using this hierarchy of manifolds, and to form the appropriate relationships between samples at each level in the hierarchy.

4 Representing and Sampling Junctions

Given a set of material indicator functions F, a set of analytical *cell indicator functions*, J, approximate the cells formed by each type of material junction (Section 4.1). In the proposed particle system sampling scheme, each particle will be constrained to a particular material junction. This formulation for each type of cell includes a set of projection operators to enforce this constraint (Section 4.2). A hierarchy of particle systems then samples each type of generically occurring material junction such that each junction is represented in the final mesh. That is, in 3D the 0-cells (points) are sampled first, followed by the 1-cells (curves), and concluding with the 2-cells (surfaces). Finally, a simple labeling algorithm extracts the multimaterial surface meshes as a subset of a Delaunay tetrahedralization of the samples (Section 4.3).

4.1 Differentiable Multimaterial Junctions

The proposed method analytically represents the individual material junctions present in the volumetric model of multimaterial datasets to allow sets of particles to specifically sample each junction. To do this, a definition of an *inside/outside* (IO) function for each material is defined using the volumetric model described in Section 3. These functions are:

$$\tilde{f}_i = f_i - \max_{j=1, j\neq i}^{n} f_j, \tag{1}$$

where positive values indicate the presence of material i and negative values indicate some other material. These functions have the property that the zero-set of any one IO function coincides with the material transitions between i and some other material. This means, for instance, that for two adjacent materials, i and j, we have $\tilde{f}_i = \tilde{f}_j = 0$ along the 2-junction where these two materials meet.

This coincidence of zero-sets for adjacent materials in Equation 1 allows for a novel representation that approximates the different kinds of material junctions (cells) within a multimaterial volume. These junctions are detected by a *cell indicator function* that identifies points in V where a set of IO functions evaluate to zero, such as the material interface between materials 1 and 2 shown as the red dashed line in Figure 3. Along this curve, $\tilde{f}_1 = \tilde{f}_2 = 0$ and $\tilde{f}_3 < 0$, while in the vicinity of this curve $\tilde{f}_1$ and $\tilde{f}_2$ will be nonzero (one negative and the other positive). Thus, in 3D, the set of 2-cells that form the interface between two materials i and j, where $i \neq j$, can be represented as the zero-set of the continuous cell indicator function:

$$J_{ij} = \tilde{f}_i^2 + \tilde{f}_j^2. \tag{2}$$

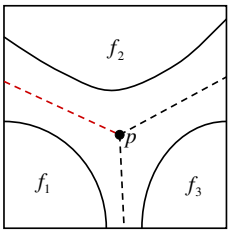


Fig. 3. Material interfaces in multimaterial datasets exist where a volumetric model of the data transitions from one maximal material to another, shown by the dotted lines for a set of three indicator functions.

In this scheme, the 1-cells for the set of materials i, j, k (assumed distinct) are given by the set of points $J_{ijk} = 0$ where:

$$J_{ijk} = \tilde{f}_i^2 + \tilde{f}_j^2 + \tilde{f}_k^2, \tag{3}$$

and likewise, the indicator for a 0-cell is:

$$J_{ijkl} = \tilde{f}_i^2 + \tilde{f}_j^2 + \tilde{f}_k^2 + \tilde{f}_l^2. \tag{4}$$

4.2 Sampling Multimaterial Junctions with Particles

To distribute a set of dynamic particles across a manifold we need to define two things: first, how particles will be projected onto the manifold; and second, how particles will be constrained to move along the manifold. The first case is usually done using a gradient descent method such as Newton-Raphson, while the latter case is most often accomplished by projecting motion vectors onto the local tangent space of the manifold. For distributing particles across multimaterial intersections, both of these tasks require first derivative information of the cell indicator functions.

The gradient of Equation 2 (with analogous definitions for Equations 3 and 4) is:

$$\nabla J_{ij} = 2\tilde{f}_i \nabla \tilde{f}_i + 2\tilde{f}_j \nabla \tilde{f}_j. \tag{5}$$

The max function is only C^0, however, and the derivative is not defined at the transition between materials. Thus, we approximate max with a smooth function that is differentiable, interpolates the $\max(i, j)$ at $i = j$, and can be tuned (via a parameter) to be arbitrarily close to max. The approximation used for the results in this paper is discussed further in the Appendix. Using this approximation, it can be shown through a Taylor series expansion around points on material junctions that zero-set surfaces exist for the cell indicator functions to within machine precision.

Notice that the cell indicator functions have the property that they are zero on the set of interest (*i.e.*, a material junction) and positive everywhere else. Because the set of interest is locally minimal, the gradient is zero on the material junctions, and thus these cell indicators, unlike the IO functions, do not directly provide the tangent spaces that are needed to constrain the motion of interacting particles. The cell indicators are constructed, however, from combinations of implicit functions for the individual materials (*i.e.*, the IO functions), and the gradients of these IO functions give the local orientation of the cells. Thus we can use a series of projection operators that rely on gradients of the IO functions to reconstruct the tangent spaces (planes or lines in 3D) of the corresponding cells.

For 2-cells, the gradients of the IO functions that characterize the junction will be approximately equal and opposite near the zero set, thus we project a motion vector of a particle, $\mathbf{v}$, onto a tangent plane that is defined by the average (for numerical robustness) of these IO function gradients:

$$\mathbf{n}_t = \frac{\nabla \tilde{f}_i - \nabla \tilde{f}_j}{|\nabla \tilde{f}_i - \nabla \tilde{f}_j|} \tag{6}$$

and update the motion vector as:

$$\mathbf{v} \leftarrow \mathbf{v} - < \mathbf{v}, \mathbf{n}_t > \mathbf{n}_t = (I - \mathbf{n}_t \otimes \mathbf{n}_t)\mathbf{v}. \tag{7}$$

On the 1-cells, particles must move along the tangent line of the zero-set of J_{ijk}. A tangent line is computed as the summation of the cross products of each pair of the three characterizing IO function normals, which all have zero-crossings along that set:

$$\mathbf{t}_{ijk} = \frac{\nabla \tilde{f}_i}{|\nabla \tilde{f}_i|} \times \frac{\nabla \tilde{f}_j}{|\nabla \tilde{f}_j|} + \frac{\nabla \tilde{f}_j}{|\nabla \tilde{f}_j|} \times \frac{\nabla \tilde{f}_k}{|\nabla \tilde{f}_k|} + \frac{\nabla \tilde{f}_k}{|\nabla \tilde{f}_k|} \times \frac{\nabla \tilde{f}_i}{|\nabla \tilde{f}_i|}. \quad (8)$$

The particle motion vectors are then projected onto this normalized tangent line, constraining the motions to the 1-cell:

$$\mathbf{v} \leftarrow \left\langle \frac{\mathbf{t}_{ijk}}{|\mathbf{t}_{ijk}|}, \mathbf{v} \right\rangle \frac{\mathbf{t}_{ijk}}{|\mathbf{t}_{ijk}|}. \quad (9)$$

The 0-cells, which are the first to be sampled, are isolated points, each sampled by a single particle (see Section 5.2), thus, we do not need to define a projection operator.

4.3 Meshing Multimaterial Samples

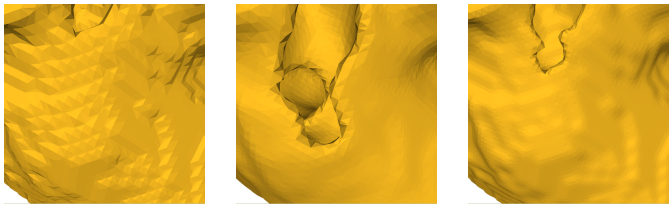
Fundamental work in inferring correct topology from a set of unorganized surface points relies on a sampling criterion that links the density of points to the LFS of the surface [4]. In order to guarantee topologically and geometrically correct surface reconstructions, surface sampling algorithms based on the LFS require an infinite sampling density (in the limit) near sharp features, such as those formed by at the 0-cells in a multimaterial dataset. Recent theoretical work [10, 12, 14], however, indicates that carefully designed meshing algorithms can handle sharp features and discontinuous surfaces. Furthermore, it can be shown in 2D that the LFS sampling constraint can be lifted around sharp features if sample points are placed explicitly on cells, allowing for the reconstruction of geometrically and topologically correct tessellations of surfaces with sharp features using Delaunay-based meshing schemes. The proof [29] guarantees this claim in 2D for a lower bound of 45° on the material angles formed by the tangent lines of the 1-cells at the 0-cell where they meet. Results from the level set literature indicate that angles between 1-cells, defined by smooth indicators functions, are 120° at the 0-cells [49], making the 45° lower bound a reasonable constraint for this work. We anticipate a similar result, with a larger angle constraint, in 3D.

Based on this proof, we know that there exists a Delaunay-based method that can reconstruct topologically correct manifold surfaces of individual materials in multimaterial datasets from the set of particles. These datasets contain additional information, though — namely, a material label for almost every point in V — that allows for a simple labeling algorithm to reliably extract the manifold material surfaces, as well as the nonmanifold intersection surface [38]. The labeling algorithm first computes a Delaunay tetrahedralization of the the sets of points sampling the 0-, 1-, and 2-cells. Next, each tetrahedron is assigned a material label by determining the material type at the location of its circumsphere center. Finally, the algorithm generates a surface mesh by extracting all faces bounded by tetrahedra with different material labels.

5 Implementation

5.1 Preprocessing Data

Representing multimaterial interfaces begins with the generation of the indicator functions f_i that characterize each material in the volume. We do this by first isolating each material label into a separate volume, smoothing each volume to control the feature size, and then constructing smooth implicit representations of the smoothed volumes. As the labeled data is smoothed as a preprocessing step, the proposed method allows a domain expert to control the amount of data processing and smoothing such that the resulting meshes capture the appropriate amount of geometry for a specific application. This allowance is a significant difference from the data smoothing that occurs in grid-based approaches, which smooth voxelization artifacts present in the multimaterial mesh as a postprocessing step with a limited amount of user control.



(a) (b) (c)

Fig. 4. When materials have thin regions, *tightening* can remove a large amount of material to obtain the required minimum radius of curvature. In (a), an isosurface of a labeled, single material volume is shown for a thin region of the material. After tightening with $r = 1$ (b), a large portion of a thin wall is eroded. In these situations it can be useful to tighten the material at a higher resolution (c) with a smaller tightening radius ($r = 0.6$ in this image). These meshes were extracted using *marching cubes* [27] and Catmull-Rom spline reconstruction kernels.

The preprocessing begins by generating a binary volume for each material, which can be down-sampled to meet the resolution constraints of the final simulation, as well as coarsely smoothed using binary morphology operations [20]. We note that the volume representation used through this work is a regular grid. Next, the volumes are antialiased [44] to produce gray scale images. Each antialiased material volume is then smoothed using a grayscale morphology algorithm proposed by Williams and Rossignac [45] called *tightening*, which limits the radius of curvature of the resulting boundary using constrained, level set curvature flow. The algorithm takes as input an antialiased volume representing a single, labeled material, along with a user-defined tightening radius r that specifies the desired minimum radius of curvature for the final, tightened surface. The output of the algorithm is a grayscale volume that stores the signed distance to a tightened material surface, where positive values indicate material.

In the process of tightening a surface to ensure a specific minimum radius of curvature, thin regions of the surface can be altered drastically by the algorithm. An example of this is shown in Figure 4, where a thin wall of material is noticeably eroded during tightening. To remedy this undesirable effect, a smaller tightening radius can be used, at the cost of less smoothing of the overall surface. However, due to the numerics of the level set framework in which the tightening algorithm is implemented, subvoxel tightening is not possible. Instead, the material volume can be upsampled to a higher resolution grid, over which a smaller tightening radius can then be applied. For the torso results shown in Figure 1, the heart material resides over a higher resolution grid then the other materials, and was tightened with $r = 0.6$ (where the units of r are given in the units of the courser grid) — all other materials presented in this paper use a tightening radius of $r = 1.0$.

Finally, we reconstruct continuous, differentiable f_i from the tightened volumes of each material using *separable convolution*, which convolves a 1D continuous kernel with grid points along each separate axis of a volume. For the results presented in this paper, we use an interpolating 4^3 Catmull-Rom spline as the continuous kernel. The reconstructed implicit functions are then input to the system as the set of indicator functions F.

5.2 Distributing Particles on Junctions

We use the particle system framework of Meyer *et al.* [31] for placing points along each material junction. This framework uses a *sizing field* that informs particles of how far they should be from their neighbors to meet LFS sampling requirements. We generate a sizing field volume for a multimaterial dataset by first computing the LFS of each IO function. We then store at each grid point in the sizing field volume the minimum LFS for the set evaluated at the grid point location.

Along sharp features, however, the LFS will go to zero, causing an infinite sampling requirement. Because sharp features in the data are explicitly sampled, the strict LFS requirements near 0-cells and 1-cells can be violated, and thus, we place a lower bound on the sizing field. This lower bound is determined by the tightening radius r (see the

previous section) that drives the tightening algorithm when smoothing the material volumes. This is (from analytical results in 2D, and empirically in 3D) a good estimate of the size of the ball within which the angle constraint (from Section 4.3) holds. For surfaces in 3D, the preprocessing (tightening) does not guarantee a lower bound on the principal curvatures in hyperbolic regions, which could lead to problems in obtaining sampling densities that ensure the angle-constraint near 0- and 1-cells, which could result in triangulations that contain faces that cross the material boundaries. In practice, this appears to be very rare, and we have not observed this problem in the results shown in this paper, despite the complexity of the datasets.

Using an ordered sampling scheme for distributing sets of particle systems, we first sample each 0-cell with a single particle, which remains fixed in place. Next, we sample 1-cells with particle systems that interact with these 0-cell particles. Similarly, the 1-cell particles are allowed to converge to a steady state and are then fixed in place. Finally, we distribute particles on the various 2-cells that interact with 0- and 1-cell particles.

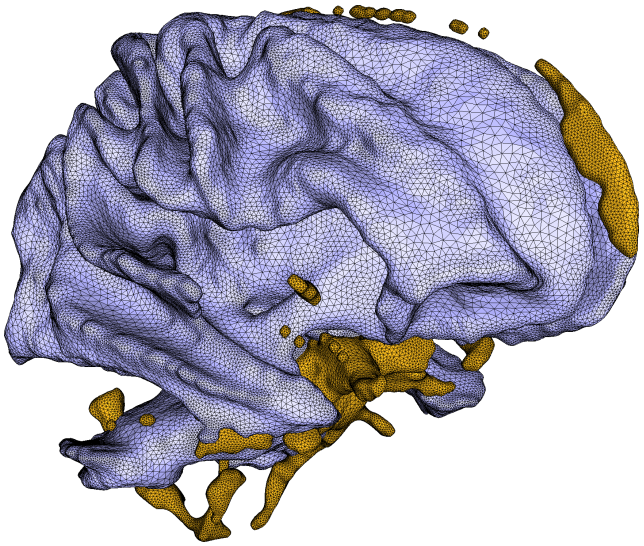


Fig. 5. Meshes of the white matter and cerebral spinal fluid (CSF) of a brain dataset generated from an MRI scan.

5.3 Meshing the Surface

We use Tetgen[1] to generate a tetrahedralization of the convex hull of the set of particles. Each tetrahedral element is labeled according to the material in which its circumscribing sphere center lies. The watertight, nonmanifold mesh of the material interfaces is the subset of faces that are bounded by tetrahedra of different material types. Faces that lie on the convex hull are labeled as having a second bounding tetrahedra of the outside material type. This nonmanifold mesh can then be used to generate volume filling elements that conform to the LFS of the boundary (see Section 6). Manifold meshes of each individual material can also be extracted in a similar fashion, with the shared boundaries of surface meshes having consistent triangulations by construction.

We have experimented with other Delaunay-based surface reconstruction algorithms, such as TightCocone [17], that are designed to infer topology from an organized set of points without knowledge of the underlying surface/solid. Our experiments showed these methods can sometimes fail, and thus we advocate the use of a simpler labeling algorithm, such as the one described in this paper, that includes information about the underlying multimaterial volume which guarantees conformal, watertight surfaces.

6 Results

We present results from several real-world datasets generated from MRI scans. The dimensions of each dataset are given in Table 1, along with the number of materials and sample points. The torso and brain datasets were sampled on a P4 3.2GHz CPU with 2GB of memory in approximately 12 hours and 3 hours, respectively. The frog and spheres datasets were sampled on a laptop with a Celeron 1.4GHz CPU and 1GB of memory in approximately 5 hours and 4 minutes, respectively. Running on four Dual-Core AMD Opteron 3.0GHz processors, the preprocessing steps for each dataset (*i.e.*, tightening each material volume, computing the medial axes, and generating the sizing field) required roughly the same amount of time as the respective particle sampling steps. We note that very little work was done to optimize the preprocessing pipeline, and we believe that these steps could be further streamlined, or, implemented with different algorithms (*e.g.*, other methods for computing the medial axes).

Table 1. The dimensions and number of materials of each dataset, and the number of particles used to sample the material junctions.

Dataset (# Materials)	Source	Volume Dimensions	# Particles
torso (5)	MRI	$260 \times 121 \times 169$	394k
frog (5)	MRI	$260 \times 245 \times 150$	186k
brain (3)	MRI	$149 \times 188 \times 148$	161k
two spheres (3)	synthetic	$128 \times 128 \times 128$	1214

A driving application for this work is the simulation of cardiac defibrillation in children. The goal is to generate a pipeline that will acquire a MRI scan of a child, generate patient-specific geometry from the scanned data, and to then determine an ideal placement for a cardiac defibrillator through FEM simulation. The torso dataset shown in Figure 1 was generated from a segmented MRI volume in this study, and consists of five materials: the torso tissue, bone, lung, heart, and air. Although other materials exist in the original MRI scan, decisions on which to include in the final simulation must be made to keep the number of elements manageable. For example, including the thin layers of fluid that exist between different organs would induce excessive numbers of elements as the feature size of this material layer is very small. Both the brain and frog datasets shown in Figures 5 and 6 were also generated from MRI scans that had been segmented into multilabel volumes. The synthetic two-sphere example in Figure 7 was generated over a grid from the difference of two analytically represented spheres.

For many FEM simulations used in biomedical computing, the *condition number*, i.e., the value that describes how numerically well-behaved a simulation will be, is directly related to the most poorly-shaped element in a tessellation. A metric that is commonly used to quantify the quality of surface meshes for FEM is the ratio of the inscribed circle to the circumscribing circle of a triangle, $2r_{in}/r_{circ}$ [43]. A ratio of one indicates an equilateral triangle, and a ratio of zero indicates a triangle that has collapsed down to an edge. In Table 2 we present statics for the tessellations generated with the proposed method, including the ratio of the most poorly shaped triangle for each mesh. These statistics indicate that not only are the bulk of the triangles nearly regular, but also that the worst shaped triangle is of consistently high-quality. This latter result is important for eliminating the time-consuming, and common, chore of hand tweaking mesh elements to make them suitable for simulations.

We compare our results against a grid-based multimaterial meshing scheme using the VTK software[2]. Common to these approaches is a pipeline that first extracts a nonmanifold mesh from a discrete, multilabel volume, followed by a smoothing step to eliminate voxelization artifacts, and finally a decimation of the mesh to decrease the number of triangles [6, 18]. Our implementation uses the *vtkDiscreteMarchingCubes* class to extract a mesh of the interfaces, the *vtkWindowedSincPolyDataFilter* to smooth the voxelization artifacts, and the *vtkQuadricDecimation* to reduce the number of triangles. We generated meshes using this pipeline of the two-sphere and frog examples

[1] tetgen.berlios.de

[2] vtk.org

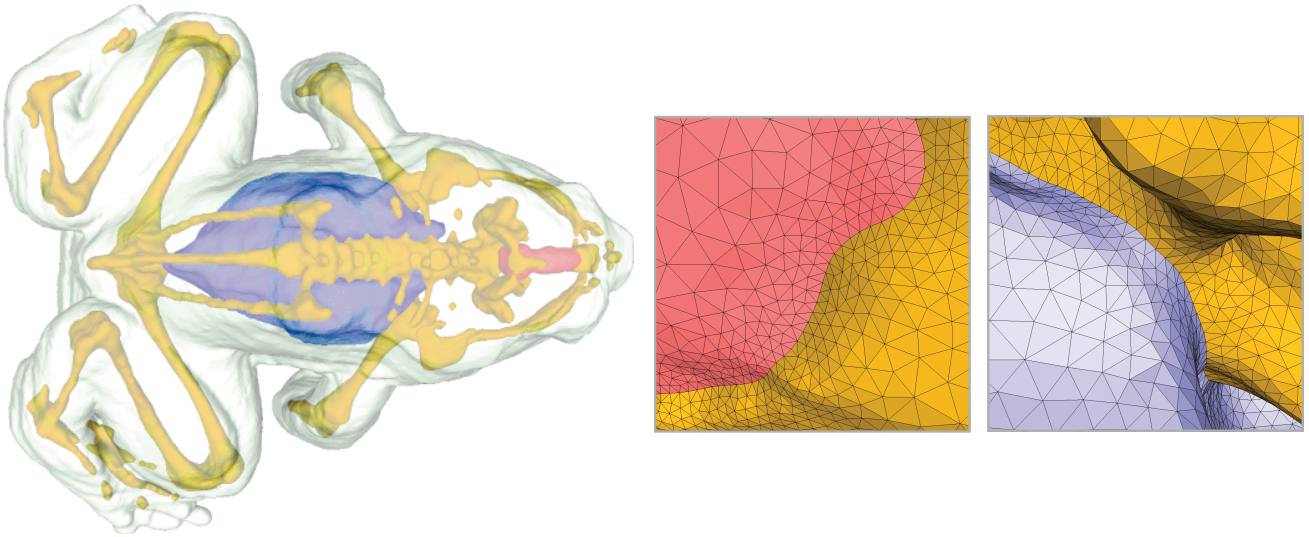


Fig. 6. Meshes of the frog dataset generated from an MRI scan, which was segmented into five different materials.

Table 2. Statistics about each mesh and their quality.

Material	Number of Triangles	Min / Avg Radius Ratio
torso tissue	673k	0.39 / 0.94
torso bone	460k	0.31 / 0.94
torso lung	215k	0.32 / 0.93
torso heart	140k	0.38 / 0.93
frog tissue	367k	0.30 / 0.94
frog bone	197k	0.37 / 0.94
frog guts	46k	0.30 / 0.94
frog brain	8k	0.52 / 0.94
brain white matter	255k	0.39 / 0.94
brain csf	92k	0.42 / 0.94
spheres top	1544	0.51 / 0.92
spheres bottom	1506	0.52 / 0.92

with approximately the same number of triangles as the analogous particle system-based examples. The minimum and average radius ratios for the spheres are 0.014 and 0.79, respectively, and for the frog 0.0 and 0.79, respectively. Not only is the quality of these meshes significantly lower than for the particle system-based meshes, but the sizes of the triangles do not adapt to the underlying geometry. Adaptive triangulations are important for efficiently capturing the geometry of an object with as few elements as possible. We present a visual comparison of these results in Figure 8.

The proposed method is also well-suited for generating volumetric samples of multimaterial datasets. We have extended the particle system framework for packing spheres inside of sampled multimaterial interfaces. The spheres are distributed using the same sizing field that guides the surface samples, which is smoothed away from the surface such that larger values are on the inside of materials. The spheres are distributed as an additional step at the end of the ordered distribution process. These volume samples can be used to generate tetrahedral meshes that conform to the material interfaces and respect the LFS of the boundaries (see Figure 9).

We have experimented with TetGen for creating tetrahedral meshes with, and without, the packed spheres, to test a range of options for generating volumetric meshes. In the latter case, we allow TetGen to place additional points inside of the surface meshes to meet quality metrics based on the *radius-edge ratio*. In both cases, however, poorly-shaped elements known as *slivers* are present in the mesh — these slivers are characterized as a flat tetrahedron with vertices that are nearly coplanar, and are a common challenge for tetrahedralization algorithms as well-spaced points do not guarantee well-shaped tetrahedra [19]. Several algorithms with shape guarantees have been developed for eliminating slivers [15, 13], although it has been noted that, in practice, inserting points at the circumsphere centers of slivers works well [42]. In Table 3 we report the minimum and average radius ratios for the tetrahedral meshes in Figure 9 (where again, a value of one indicates a regular tetrahedron), along with the percentage of elements that are characterized as slivers (*i.e.*, with a radius ratio less than 0.1). We observed that the slivers in the tetrahedral meshes generated from packed spheres exist almost exclusively near the surface mesh, which is a similar phenomena encountered in other tetrahedral mesh generation algorithms [3].

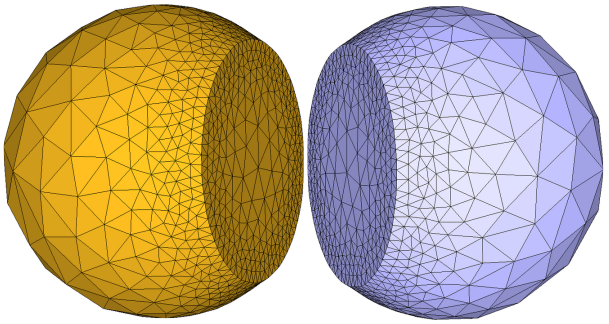


Fig. 7. A synthetic example of two intersecting spheres, illustrating the consistency of the meshes along the shared boundary.

To provide a proof-of-concept for taking real-world data containing multiple materials through a complete simulation pipeline, we have implemented a point-based physics algorithm using the packed spheres and surface meshes. In these simulations, research has shown that both physical and geometric complexity are highly correlated [1], and that the stability and accuracy of the simulation is directly related to the ability of the surface and volume samples to capture the LFS of the material boundaries. The algorithm simulates elastic materials [32], and is extended to handle multimaterial objects by assigning different physical properties to volume samples of different material types. We utilize surface meshes generated from the particles sampling the multimaterial interfaces to include collision detection between materials of different types [25], as well as for rendering the simulation results. In the supplemental video we present simulation results of the frog dataset dropping onto a flat surface. The video contrasts the results from assigning all volume samples the same material parameters with that from assigning different parameters to each material. By assigning different properties to each material in the frog, more

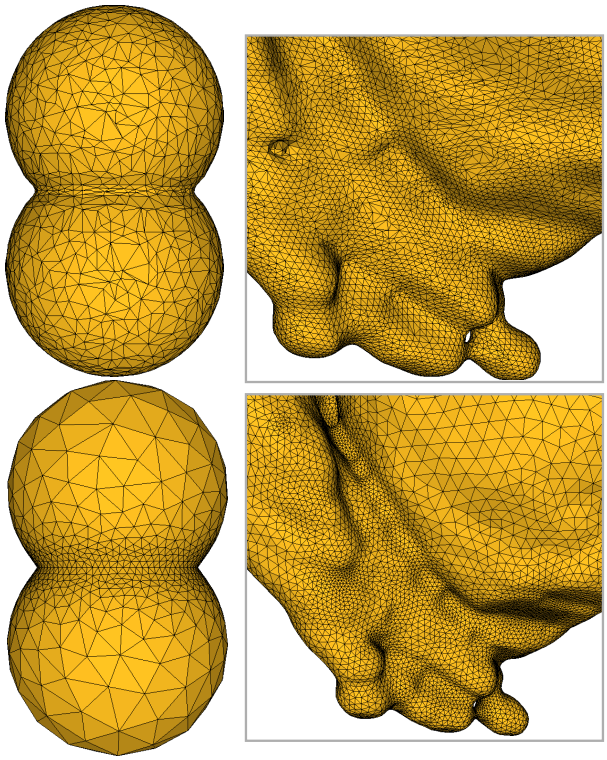


Fig. 8. Comparisons of multimaterial meshes generated using a grid-based approach (top) and our particle system-based approach (bottom). The left column is the two-sphere example, and the right column is a closeup from the frog example.

complex and realistic simulations can be achieved, including stiffness within the head and body due to the rigid bone structure. This example illustrates not only the importance of modeling multiple materials of objects for increasingly realistic simulations, but also the potential for automatically generating complex digital models from scanned, real-world objects.

Table 3. Quality of tetrahedral meshes generated for the datasets in Figure 9. The different generation methods are: (SP) sphere packing; (SP-1R) sphere packing with one iteration of refinement; (SP-2R) sphere packing with two iterations of refinement; (TG) volumetric points added by TetGen. An element is considered a sliver when its radius ratio is less than 0.1.

Material	Generation Method	Number of Tetrahedra	Min / Avg Radius Ratio	Sliver %
two spheres	SP	12.7k	0.02 / 0.78	0.24(%)
two spheres	SP-1R	12.8k	0.05 / 0.78	0.02
two spheres	SP-2R	12.8k	0.10 / 0.78	0.00
two spheres	TG	30.2k	0.06 / 0.59	0.1
frog	SP	443k	0.0001 / 0.77	0.30
frog	SP-1R	452k	0.0001 / 0.77	0.06
frog	SP-2R	453k	0.0001 / 0.77	0.06
frog	TG	1.146M	0.0259 / 0.60	0.08

7 Conclusions and Future Work

The high-quality results of our method come at the cost of long computation times. Most of that time is spent in preprocessing the multilabel data and distributing the sets of particles. The number of materials in the dataset also adds to the overall computation time as the max function must evaluate every f_i that exists in the set. However, the quality of the meshes from our particle-based scheme is so high that we can avoid the usually time-consuming step of meticulous hand-editing of mesh vertices to ensure well-shaped elements. We are currently working towards an optimized implementation of this pipeline for public release, and we are interested in adapting more complex tetrahedralization algorithms to the multimaterial surfaces to produce higher quality volume meshes.

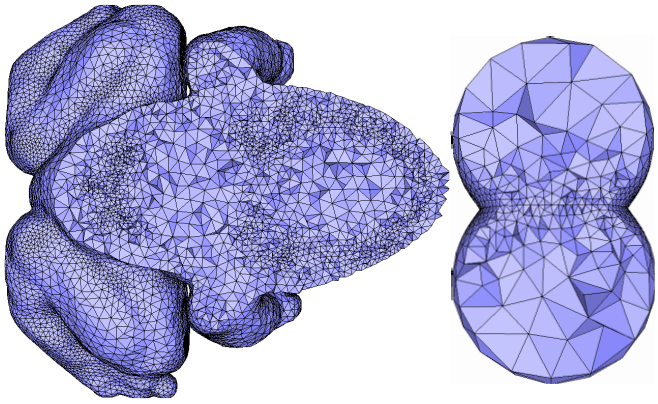


Fig. 9. Tetrahedralizations of the interface and volume particle samples, shown with a cutting plane.

8 Appendix

Differentiable Approximation to **max** We present an analytic, differentiable approximation to max for a set of m functions $v_i \in V$ that interpolates the $\max(i, j)$ at $i = j$ by first defining a function g:

$$g(v) = 1 + \frac{v}{(v^2 + \varepsilon_{\max}^2)^{1/2}}. \tag{10}$$

The max function is then:

$$\max(V) = \frac{1}{2^{m-1}} \sum_{i=1}^{m} v_i \prod_{j=1, j\neq i}^{m} g(v_i - v_j) \tag{11}$$

with the gradient given by:

$$\nabla \max(V) = \frac{1}{2^{m-1}} \sum_{i=1}^{m} \left[\nabla v_i \prod_{j=1, j\neq i}^{m} g(v_i - v_j) + v_i \left(\sum_{j=1, j\neq i}^{m} \nabla g(v_i - v_j) \prod_{j=1, j\neq i}^{m} g(v_i - v_j) \right) \right] \tag{12}$$

where:

$$\nabla g(v) = \nabla v \left[\frac{1}{(v^2 + \varepsilon_{\max}^2)^{1/2}} - \frac{v^2}{(v^2 + \varepsilon_{\max}^2)^{3/2}} \right]. \tag{13}$$

For the results in this paper we use $\varepsilon_{\max} = 10^{-5}$.

Acknowledgements

The authors wish to thank Claudio Silva for the insightful conversations that inspired this work. The torso dataset is courtesy of Matthew Jolley at Children's Hospital Boston, and the brain dataset is courtesy of Tolga Tasdizen. This work was supported in part by ARO grant W911NF-05-1-0395, NIH/NCRR Center for Integrative Biomedical Computing, P41-RR12553-07, and NSF-CNS grant CNS 0551724.

References

[1] B. Adams, M. Pauly, R. Keiser, and L. J. Guibas. Adaptively sampled particle fluids. *ACM Transactions on Graphics*, 26(3):48, July 2007.

[2] A. Adamson and M. Alexa. Point-sampled cell complexes. *ACM Trans. Graph.*, 25(3):671–680, 2006.

[3] P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun. Variational tetrahedral meshing. *ACM Transactions On Graphics*, 24(3):617–625, 2005.

[4] N. Amenta, M. Bern, and D. Eppstein. The crust and the beta-skeleton: Combinatorial curve reconstruction. *Graphic Models and Image Processing*, 60(2):125–135, Mar. 1998.

[5] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH*, pages 415–421, July 1998.

[6] M. Bertram, G. Reis, R. H. van Lengen, S. Köhn, and H. Hagen. Non-manifold mesh extraction from time-varying segmented volumes used for modeling a human heart. In *Proceedings of EuroVis*, pages 199–206, June 2005.

[7] S. Bischoff and L. Kobbelt. *Extracting Consistent and Manifold Interfaces from Multi-valued Volume Data Sets*. 2006.

[8] J. Bloomenthal and K. Ferguson. Polygonization of non-manifold implicit surfaces. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 309–316, New York, NY, USA, 1995. ACM.

[9] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405–451, Sept. 2005.

[10] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of lipschitz surfaces. In *SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry*, pages 337–346, New York, NY, USA, 2006. ACM.

[11] K. S. Bonnell, M. A. Duchaineau, D. R. Schikore, B. Hamann, and K. I. Joy. Material interface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):500–511, Oct./Dec. 2003.

[12] F. Chazal, D. Cohen-Steiner, and A. Lieutier. A sampling theory for compact sets in euclidean space. In *SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry*, pages 319–326, New York, NY, USA, 2006. ACM.

[13] S.-W. Cheng, T. K. Dey, H. Edelsbrunner, M. A. Facello, and S.-H. Teng. Sliver exudation. *J. ACM*, 47(5):883–904, 2000.

[14] S.-W. Cheng, T. K. Dey, and E. A. Ramos. Delaunay refinement for piecewise smooth complexes. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1096–1105, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[15] L. P. Chew. Guaranteed-quality delaunay meshing in 3d (short version). In *SCG '97: Proceedings of the thirteenth annual symposium on Computational geometry*, pages 391–393, New York, NY, USA, 1997. ACM Press.

[16] P. Crossno and E. Angel. Isosurface extraction using particle systems. In *Proceedings of the 8th conference on Visualization '97*, pages 495–498. IEEE Computer Society Press, 1997.

[17] T. K. Dey and S. Goswami. Tight Cocone: A water-tight surface reconstructor. *Journal of Computing and Information Science in Engineering*, 3(4):302–307, December 2003.

[18] S. Dillard, J. Bingert, and D. Thoma. Construction of simplified boundary surfaces from serial-sectioned metal micrographs. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1528–1535, Nov./Dec. 2007.

[19] D. Eppstein. Global optimization of mesh quality. In *Tutorial at the 10th International Meshing Roundtable*. Newport Beach.

[20] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[21] J. C. Hart. Using the CW-complex to represent the topological structure of implicit surfaces and solids. In *Proceedings of Implicit Surfaces*, pages 107–112, 1999.

[22] H.-C. Hege, M. Seebaß, D. Stalling, and M. Zöckler. A generalized marching cubes algorithm based on non-binary classifications. ZIB Preprint SC-97-05, 1997.

[23] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. *ACM Transactions on Graphics*, 21(3):339–346, July 2002.

[24] T. Karkanis and A. J. Stewart. Curvature-dependent triangulation of implicit surfaces. *IEEE Computer Graphics and Applications*, 22(2):60–69, March 2001.

[25] R. Keiser, M. Müller, B. Heidelberger, M. Teschner, and M. Gross. Contact handling for deformable point-based objects. In *Proceedings of the 9th International Workshop "Vision, Modeling, and Visualization"*, pages 339–346, Nov 2004.

[26] J. Kniss, W. Hunt, K. Potter, and P. Sen. Istar: A raster representation for scalable image and volume data. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1424–1431, 2007.

[27] W. E. Lorensen and H. E. Cline. Marching Cubes: A high resolution 3d surface construction algorithm. In *Proceedings of ACM SIGGRAPH*, pages 163–169, July 1987.

[28] B. Merriman, J. K. Bence, and S. J. Osher. Motion of multiple junctions: A level set approach. *J. Comput. Phys.*, 112(2):334–363, 1994.

[29] M. Meyer. *Dynamic Particles for Adaptive Sampling of Implicit Surfaces*. PhD thesis, School of Computing, University of Utah, 2008.

[30] M. Meyer, P. Georgel, and R. Whitaker. Robust particle systems for curvature dependent sampling of implicit surfaces. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI)*, pages 124–133, June 2005.

[31] M. Meyer, R. M. Kirby, and R. Whitaker. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1704–1711, September/October 2007.

[32] M. Miller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 141–151, New York, NY, USA, 2004. ACM Press.

[33] G. Nielson and R. Franke. Computing the separating surface for segmented data. In *IEEE Visualization*, pages 229–236, Nov. 1997.

[34] G. M. Nielson, A. Huang, and S. Sylvester. Approximating normals for marching cubes applied to locally supported isosurfaces. In *IEEE Visualization*, pages 459–466, 2002.

[35] S. Osher and R. Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2003.

[36] S. Oudot, L. Rineau, and M. Yvinec. Meshing volumes bounded by smooth surfaces. In *Proc. 14th International Meshing Roundtable*, pages 203–219, 2005.

[37] A. A. Pasko, V. Adzhiev, A. Sourin, and V. V. Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer*, 11(8):429–446, 1995.

[38] J.-P. Pons, F. Ségonne, J.-D. Boissonnat, L. Rineau, M. Yvinec, and R. Keriven. High-quality consistent meshing of multi-label datasets. In *Information Processing in Medical Imaging*, pages 198–210, 2007.

[39] B. Reitinger, A. Bornik, and R. Beichel. Constructing smooth non-manifold meshes of multi-labeled volumetric datasets. In *WSCG (Full Papers)*, pages 227–234, 2005.

[40] J. Schreiner, C. Scheidegger, and C. Silva. High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1205–1212, Sept./Oct. 2006.

[41] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge Monograph on Applied and Computational Mathematics. Cambridge University Press, 1999.

[42] J. R. Shewchuk. Tetrahedral mesh generation by delaunay refinement. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry*, pages 86–95, New York, NY, USA, 1998. ACM.

[43] J. R. Shewchuk. What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures. Technical Report Manuscript, 2002.

[44] R. T. Whitaker. Reducing aliasing artifacts in iso-surfaces of binary volumes. In *VVS '00: Proceedings of the 2000 IEEE symposium on Volume visualization*, pages 23–32, New York, NY, USA, 2000. ACM Press.

[45] J. Williams and J. Rossignac. Tightening: Curvature-limiting morphological simplification. In *Proceedings of the Symposium on Solid and Physical Modeling*, pages 107–112, 2005.

[46] A. P. Witkin and P. S. Heckbert. Using particles to sample and control implicit surfaces. In *Proceedings of SIGGRAPH*, pages 269–277, July 1994.

[47] S. Yamazaki, K. Kase, and K. Ikeuchi. Non-manifold implicit surfaces based on discontinuous implicitization and polygonization. *Geometric Modeling and Processing, 2002. Proceedings*, pages 138–146, 2002.

[48] Y. Zhang, T. J. R. Hughes, and C. Bajaj. Automatic 3d mesh generation for a domain with multiple materials. In *IMR*, pages 367–386, 2007.

[49] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *Journal of Computational Physics*, 127(1):179–195, 1996.