Tetrahedral Mesh Generation for Deformable Bodies

Neil Molino* Stanford University
Robert Bridson† Stanford University
Ronald Fedkiw‡ Stanford University

Abstract

Motivated by the simulation of deformable bodies, we propose a new tetrahedral mesh generation algorithm that produces both high quality elements *and* a mesh that is well conditioned for subsequent large deformations. We use a signed distance function defined on a grid in order to represent the object geometry. After tiling space with a uniform lattice based on crystallography, we identify a subset of these tetrahedra that adequately fill the space occupied by the object. Then we use the signed distance function or other user defined criteria to guide a red green mesh subdivision algorithm that results in a candidate mesh with the appropriate level of detail. After this, both the signed distance function and topological considerations are used to prune the mesh as close to the desired shape as possible while keeping the mesh flexible for large deformations. Finally, we compress the mesh to tightly fit the object boundary using either masses and springs, the finite element method or an optimization approach to relax the positions of *both* the interior and boundary nodes. The resulting mesh is well suited for simulation since it is highly structured, has robust topological connectivity in the face of large deformations, and is readily refined if deemed necessary during subsequent simulation.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid and object representations

Keywords: tetrahedral mesh generation, BCC lattice, red green refinement, finite element method, level set methods

1 Introduction

Tetrahedral meshes are used in a number of areas including fracture [O'Brien and Hodgins 1999; O'Brien et al. 2002], haptics [Debunne et al. 2001], solid modeling [Cutler et al. 2002], surgical simulations [Ganovelli et al. 2000], and the modeling of biological tissue including the brain [Grinspun et al. 2002], the knee [Hirota et al. 2001] and even fatty tissue [James and Pai 2002]. Robust methods for generating these meshes are in high demand. Of particular interest to us are simulations of highly deformable bodies such as the muscle and fatty tissues commonly encountered in graphics, biomechanics or virtual surgery.

*e-mail: npmolino@stanford.edu
†e-mail: rbridson@stanford.edu
‡e-mail: fedkiw@cs.stanford.edu

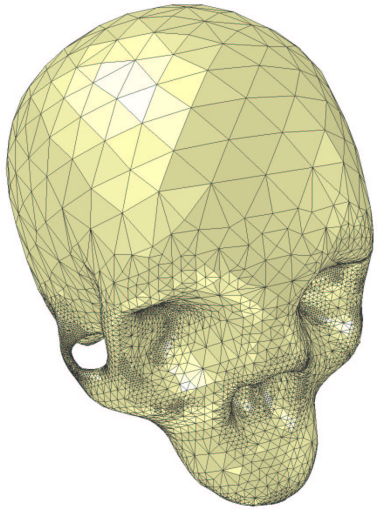

Figure 1: **Tetrahedral mesh of a cranium (80K elements).**

Mesh generation is not only a broad field, but is in some sense many fields each concerned with the creation of meshes that conform to quality measures specific to the application at hand. *The requirements for fluid flow and heat transfer where the mesh is not deformed, and for small deformation solids where the mesh is barely deformed, can be quite different from those for simulating soft biological tissue that may undergo large deformations.* For example, while an optimal mesh for a fluid flow simulation should include anisotropically compressed elements in boundary layers, e.g. [Garimella and Shephard 1998; Lohner and Cebral 1999], these highly stretched cells tend to be ill-conditioned when a mesh deforms significantly as is typical for soft bodies. Either the mesh is softer in the thin direction and the cell has a tendency to invert, or the mesh is stiffer in the thin direction and the simulation becomes very costly as the time step shrinks with higher stiffness and smaller element cross-section. Thus, although our method has been designed to provide a high degree of adaptivity both to resolve the geometry and to guarantee quality simulation results, we neither consider nor desire anisotropically stretched elements. Also, since highly deformable bodies tend to be devoid of sharp features such as edges and corners, we do not consider boundary feature preservation.

Our main concern is to generate a mesh that will be robust when subsequently subject to large deformations. For example, although we obviously want an adaptive mesh with smaller elements in areas where more detail is desired, it is even more important to have a mesh that can be adapted during the simulation since these regions will change. Motivated by crystallography, we use a body-centered cubic (BCC) mesh (see e.g. [Burns and Glazer 1990]) that is highly structured and produces similar (in the precise geometric sense) tetrahedra under regular refinement. This allows us to adaptively refine both while generating the mesh and during the subsequent simulation.

We start with a uniform tiling of space and use a signed dis-

tance function representation of the geometry to guide the creation of the adaptive mesh, the deletion of elements that are not needed to represent the object of interest, and the compression of the mesh necessary to match the object boundaries. This compression stage can be carried out using either a mass spring system, a finite element method or an optimization based approach. One advantage of using a physically based compression algorithm is that it gives an indication of how the mesh is likely to respond to the deformations it will experience during simulation. This is in contrast to many traditional methods that may produce an initial mesh with good quality measures, but also with hidden deficiencies that can be revealed during simulation leading to poor accuracy or element collapse. Moreover, our novel topological considerations (discussed below) were specifically designed to address these potential defects present in most (if not all) other mesh generation schemes.

2 Related Work

Delaunay methods have not been as successful in three spatial dimensions as in two, since they admit flat sliver tetrahedra of negligible volume. [Shewchuk 1998] provides a nice overview of these methods, including a discussion of why some of the theoretical results are not reassuring in practice. Moreover, he discusses how the worst slivers can often be removed. [Cheng et al. 2000] also discusses sliver removal, but states that their theorem gives an estimate that is "miserably tiny".

Advancing front methods start with a boundary discretization and march a "front" inward forming new elements attached to the existing ones, see e.g. [Schöberl 1997]. While they conform well to the boundary, they have difficulty when fronts merge, which unfortunately can occur very near the important boundary in regions of high curvature, see e.g. [Garimella and Shephard 1998; Lohner and Cebral 1999]. In [Radovitzky and Ortiz 2000], the authors start with a face-centered cubic (FCC) lattice defined on an octree and use an advancing front approach to march inward constructing a mesh with the predetermined nodes of the FCC lattice. They choose FCC over BCC because it gives slightly better tetrahedra for their error bounds. However, after any significant deformation the two meshes will usually have similar character. Moreover, since we keep our BCC connectivity intact (as opposed to [Radovitzky and Ortiz 2000]), we retain the ability to further refine our BCC mesh during the calculation to obtain locally higher resolution for improved accuracy and robustness. On the other hand, their approach is better at resolving boundary features and is thus most likely superior for problems with little to no deformation.

[Shimada and Gossard 1995] packed spheres (or ellipsoids [Yamakawa and Shimada 2000]) into the domain with mutual attraction and repulsion forces, and generated tetrahedra using the sphere centers as sample points via either a Delaunay or advancing front method. However, *ad hoc* addition and deletion of spheres is required in a search for a steady state, and both local minima and "popping" can be problematic. This led [Li et al. 1999] to propose the removal of the dynamics from the packing process, instead marching in from the boundary removing spherical "bites" of volume one at a time.

[Debunne et al. 2001] carried out finite element simulations on a hierarchy of tetrahedral meshes switching to the finer meshes only locally where more detail is needed. They state that one motivation for choosing this strategy is the difficulty in preserving mesh quality when subdividing tetrahedral meshes. [Grinspun et al. 2002] proposes basis refinement pointing out that while T-junctions are easily removed in two spatial dimensions, it is more involved in three spatial dimensions. Our method alleviates both of these concerns since subdivision of our BCC mesh leads to a similar (the tetrahedra are congruent up to a dilation) BCC mesh, and the particular red green strategy that we use can be implemented in a straightforward manner.

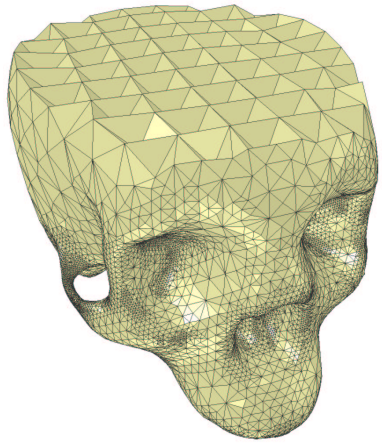

Figure 2: **Tetrahedral mesh of a cranium (cutaway view) showing the regularity and coarseness of the interior tetrahedra (80K elements).**

[Cutler et al. 2002] starts with either a uniform grid or a graded octree mesh and subdivides each cell into five tetrahedra. Then tetrahedra that cross an internal or external interface are simply split, which can form poorly shaped elements for simulation. Neighboring tetrahedra are also split to avoid T-junctions. The usual tetrahedral flips and edge collapses are carried out in an attempt to repair the ill-conditioned mesh.

Our compression phase moves the nodes on the boundary of our candidate mesh to the implicit surface providing boundary conformity. Related work includes [Bloomenthal and Wyvill 1990] who attracted particles to implicit surfaces for visualization, [Turk 1991; Turk 1992] who used particles and repulsion forces to sample and retriangulate surfaces, [de Figueiredo et al. 1992; Crossno and Angel 1997] who used the same idea to triangulate implicit surfaces, [Szeliski and Tonnesen 1992] who modeled surfaces with oriented particles, and [Witkin and Heckbert 1994] who used particle repulsion to sample (and control) implicit surfaces.

In some sense, this wrapping of our boundary around the level set is related to snakes [Kass et al. 1987] or GDM's [Miller et al. 1991] which have been used to triangulate isosurfaces, see e.g. [Sadarjoen and Post 1997]. [Neugebauer and Klein 1997] started with a marching cubes mesh and moved vertices to the centroid of their neighbors before projecting them onto the zero level set in the neighboring triangles' average normal direction. [Grosskopf and Neugebauer 1998] improved this method using internodal springs instead of projection to the centroid, incremental projection to the zero isocontour, adaptive subdivision, edge collapse and edge swapping. [Kobbelt et al. 1999] used related ideas to wrap a mesh with subdivision connectivity around an arbitrary one (other interesting earlier work includes [Hoppe et al. 1993; Hoppe et al. 1994]), but had difficulty projecting nodes in one step, emphasizing the need for slower evolution. Similar techniques were used by [Bertram et al. 2000; Hormann et al. 2002] to wrap subdivision surfaces around implicit surfaces. To improve robustness, [Wood et al. 2000] adaptively sampled the implicit surface on the faces of triangles, redistributed the isovalues times the triangle normals to the vertices to obtain forces, and replaced the spring forces with a modified Laplacian smoothing restricted to the tangential direction. [Ohtake and Belyaev 2002] advocated moving the triangle centroids to the zero isocontour instead of the nodes, and matching the triangle normals with the implicit surface normals.

Although we derive motivation from this work, we note that our

problem is significantly more difficult since these authors move their mesh in a direction normal to the surface, which is orthogonal to their measure of mesh quality (shapes of triangles tangent to the surface). When we move our mesh normal to the surface, it directly conflicts with the quality of the surface tetrahedra. [de Figueiredo et al. 1992] evolved a volumetric mass spring system in order to align it with (but not compress it to) the zero isocontour, but the measure of mesh quality was still perpendicular to the evolution direction since the goal was to triangulate the zero isocontour. Later, however, [Velho et al. 1997] *did* push in a direction conflicting with mesh quality. They deformed a uniform-resolution Freudenthal lattice to obtain tetrahedralizations using a mass spring model, but were restricted to simple geometries mostly due to the inability to incorporate adaptivity. In two spatial dimensions, [Gloth and Vilsmeier 2000] also moved the mesh in a direction that opposed the element quality. They started with a uniform Cartesian grid bisected into triangles, threw out elements that intersected or were outside the domain, and moved nodes to the boundary in the direction of the gradient of the level set function using traditional smoothing, edge-swapping, insertion and deletion techniques on the mesh as it deformed.

3 A Crystalline Mesh

We turn our attention to the physical world for inspiration and start our meshing process with a body-centered cubic (BCC) tetrahedral lattice. This mesh has numerous desirable properties and is an actual crystal structure ubiquitous in nature, appearing in vastly different materials such as soft lithium and hard iron crystals, see e.g. [Burns and Glazer 1990].

The BCC lattice consists of nodes at every point of a Cartesian grid along with the cell centers. These node locations may be viewed as belonging to two interlaced grids. Additional edge connections are made between a node and its eight nearest neighbors in the other grid. See figure 3 where these connections are depicted in red and the two interlaced grids are depicted in blue and in green. The BCC lattice is the Delaunay complex of the interlaced grid nodes, and thus possesses all properties of a Delaunay tetrahedralization. Moreover, all the nodes are isomorphic to each other (and in particular have uniform valence), every tetrahedron is congruent to the others, and the mesh is isotropic (so the mesh itself will not erroneously induce any anisotropic bias into a subsequent calculation). The BCC lattice is structured and thus has advantages for iterative solvers, multigrid algorithms, and both computational and memory requirements.

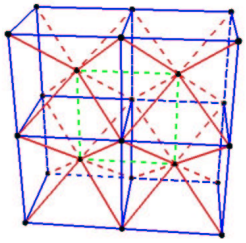

Figure 3: **A portion of the BCC lattice. The blue and the green connections depict the two interlaced grids, and the eight red connections at each node lace these two grids together.**

There are a number of ways to improve the accuracy of a finite element simulation. R-refinement relocates existing mesh nodes to better locations, h-refinement locally refines the mesh to obtain smaller elements where needed, and p-refinement increases the polynomial order of the interpolating functions—see e.g. [Cheng et al. 1988]. A significant advantage of the BCC mesh is that it is easily refined initially or during the calculation. Each regular BCC tetrahedron can be refined into eight tetrahedra, shown in red in figure 4, with a one to eight (or 1:8) refinement. When the shortest of the three possible choices for the edge internal to the tetrahedron is taken, the newly formed tetrahedra are *exactly* the BCC tetrahedra that result from a mesh with cells one half the size. Thus, these eight new tetrahedra are geometrically similar to the tetrahedra of the parent mesh and element quality is guaranteed under this regular 1:8 refinement.

4 Red Green Refinement

Many applications do not require and cannot afford (due to computation time and memory restrictions) a uniformly high resolution mesh. For example, many graphical simulations can tolerate low accuracy in the unseen interior of a body, and many phenomena such as contact and fracture show highly concentrated stress patterns, often near high surface curvature, outside of which larger tetrahedra are acceptable. Thus, we require the ability to generate adaptive meshes.

As the BCC lattice is built from cubes, one natural approach to adaptivity is to build its analog based on an octree. We implemented this by adding body centers to the octree leaves, after ensuring the octree was graded with no adjacent cells differing by more than one level. The resulting BCC lattices at different scales were then patched together with special case tetrahedra. For more on octrees in mesh generation, see e.g. [Shephard and Georges 1991; Radovitzky and Ortiz 2000].

However, we found that red green refinement is more economical, simpler to implement, and more flexible, see e.g. [Bey 1995; Grosso et al. 1997; de Cougny and Shephard 1999]. The initial BCC lattice tetrahedra are labeled red, as are any of their eight children obtained with our 1:8 subdivision shown in red in figure 4. Performing a red refinement on a tetrahedron creates T-junctions at the newly-created edge midpoints where neighboring tetrahedra are not refined to the same level. To eliminate these, the red tetrahedra with T-junctions are irregularly refined into fewer than eight children by introducing some of the midpoints. These children are labeled green, and are of lower quality than the red tetrahedra that are part of the BCC mesh. Moreover, since they are not BCC tetrahedra, we never refine them. When higher resolution is desired in a region occupied by a green tetrahedron, the entire family of green tetrahedra is removed from its red parent, and the red parent is refined regularly to obtain eight red children that can undergo subsequent refinement.

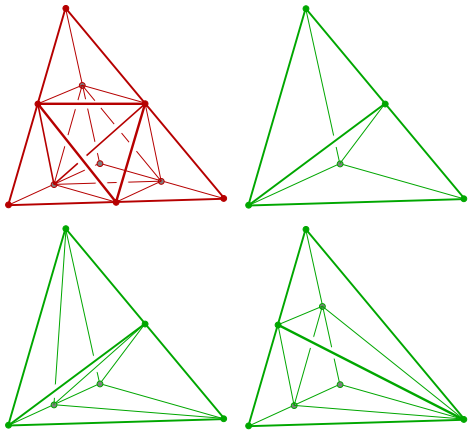

Figure 4: **The standard red refinement (depicted in red) produces eight children that reside on a BCC lattice that is one-half the size. Three green refinements are allowed (depicted in green).**

A red tetrahedron that needs a green refinement can have between one and six midpoints on its edges (in the case of six we do red refinement). We reduce the possibilities for green refinement to those shown in figure 4, adding extra edge midpoints if necessary. This restriction (where all triangles are either bisected or quadrisected) smooths the gradation further and guarantees higher quality green tetrahedra. While there can of course be a cascading effect as the extra midpoints may induce more red or green refinements, it is a small price to pay for the superior mesh quality and seems to be a minor issue in practice.

Any criteria may be used to drive refinement, and we experimented with the geometric rules described in the next section. A significant advantage of the red green framework is the possibility for refinement during simulation based on *a posteriori* error estimates, with superior quality guarantees based on the BCC lattice instead of an arbitrary initial mesh. Note that the lower quality green tetrahedra can be replaced by finer red tetrahedra which admit further refinement. However, one difficulty we foresee is in discarding portions of green families near the boundary (see section 6), since part of the red parent is missing. To further refine this tetrahedron, the green family has to be replaced with its red parent which can be regularly refined, then some of the red children need to be discarded and the others must be compressed to the boundary (see sections 7–8). A simpler but lower quality alternative is to arbitrarily relabel those green boundary tetrahedra that are missing siblings as red allowing them to be directly refined. We plan to address this issue in future work.

5 Representing the Geometry

We represent the geometry with a signed distance function defined on either a uniform grid [Osher and Fedkiw 2002] or an octree grid [Frisken et al. 2000]. In the octree case, we constrain values of fine grid nodes at gradation boundaries to match the coarse grid interpolated values, see e.g. [Westermann et al. 1999]. When the signed distance function has a resolution much higher than that of our desired tetrahedral mesh, we apply motion by mean curvature to smooth the high frequency features and then reinitialize to a signed distance function, see e.g. [Osher and Fedkiw 2002] (and [Desbrun et al. 1999] where curvature motion was applied directly to a triangulated surface).

At any point in space, we calculate the distance from the implicitly defined surface as ϕ which is negative inside and positive outside the surface. To obtain a finer mesh near the boundary, one simply refines tetrahedra that include portions of the interface where $\phi = 0$. If a tetrahedron has nodes with positive values of ϕ and nodes with negative values of ϕ, it obviously contains the interface and can be refined. Otherwise, the tetrahedron is guaranteed not to intersect the interface if the minimum value of $|\phi|$ at a node is larger than the longest edge length (tighter estimates are available

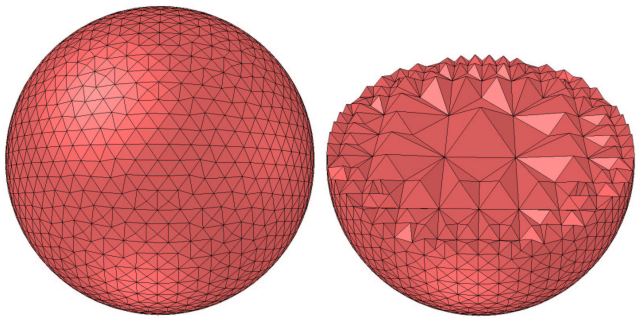

Figure 5: **Tetrahedral mesh of a sphere (18K elements). The cutaway view illustrates that the interior mesh can be fairly coarse even if high resolution is desired on the boundary.**

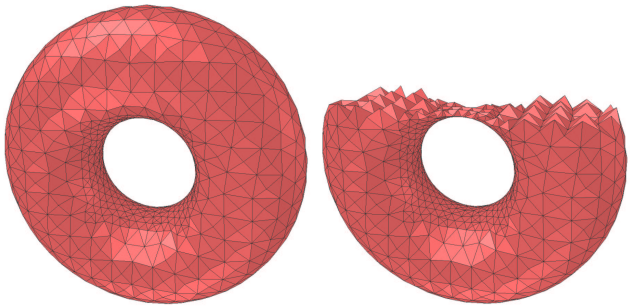

Figure 6: **Tetrahedral mesh of a torus (8.5K elements). The principal curvatures were used to increase the level of resolution in the inner ring.**

of course). The remaining cases are checked by sampling ϕ appropriately (at the level set grid size $\triangle x$), allowing refinement if any sample is close enough to the interface ($|\phi| < \triangle x$). Figure 5 shows a sphere adaptively refined near its boundary. Note how the interior mesh can still be rather coarse.

The outward unit normal is defined as $N = \nabla\phi$ and the mean curvature is defined as $\kappa = \nabla \cdot N$. One may wish to adaptively refine in regions of high curvature, but the mean curvature is a poor measure of this since it is the average of the principal curvatures, $(k_1 + k_2)/2$, and can be small at saddle points where positive and negative curvatures cancel. Instead we use $|k_1| + |k_2|$. The principal curvatures are computed by forming the Hessian,

$$H = \begin{pmatrix} \phi_{xx} & \phi_{xy} & \phi_{xz} \\ \phi_{xy} & \phi_{yy} & \phi_{yz} \\ \phi_{xz} & \phi_{yz} & \phi_{zz} \end{pmatrix},$$

and projecting out the components in the normal direction via the projection matrix $P = I - NN^T$. Then the eigenvalues of $PHP/|\nabla\phi|$ are computed, the zero eigenvalue is discarded as corresponding to the eigenvector N, and the remaining two eigenvalues are k_1 and k_2. See e.g. [Ambrosio and Soner 1996]. To detect whether a tetrahedron contains regions of high curvature, we sample at a fine level and check the curvature at each sample point. Figure 6 shows a torus where the inner ring is refined to higher resolution even though the principal curvatures there differ in sign.

6 Constructing a Candidate Mesh

In the previous three sections we described the BCC lattice, the red green refinement strategy and the representation of the geometry of interest with a signed distance function. This section describes how we tie these ideas together to obtain a candidate tetrahedral mesh of our object. This candidate mesh will be compressed to the boundary using one of the methods described in the next two sections.

The first step is to cover an appropriately sized bounding box of the object with a coarse BCC mesh. Then we use a conservative discard process to remove tetrahedra that are guaranteed to lie completely outside of the zero isocontour: tetrahedra with four positive ϕ values all larger than the maximum edge length are removed.

In the next step, the remaining tetrahedra are refined according to any user defined criteria, such as *a posteriori* error estimates, indicator variables or geometric properties. We have experimented with using both the magnitude of ϕ and various measures of curvature as discussed in the previous section. Using simply the magnitude of ϕ produces large tetrahedra deep inside the object and a uniform level of refinement around the surface, which can be useful since objects interact with each other via surface tetrahedra. A more sophisticated method uses the surface principal curvatures, better resolving

complex geometry and allowing for more robust and efficient simulation when subject to large deformation. We refine any tetrahedron near the interface if its maximum edge length is too large compared to a radius of curvature measure, $1/(|k_1| + |k_2|)$, indicating an inability to resolve the local geometry. We refine to a user-specified number of levels, resolving T-junctions in the red green framework as needed.

From the adaptively refined lattice we will select a subset of tetrahedra that closely matches the object. However, there are specific topological requirements necessary to ensure a valid mesh that behaves well under deformation: the boundary must be a manifold; no tetrahedron may have all four nodes on the boundary; and no interior edge may connect two boundary nodes. Boundary forces can readily crush tetrahedra with all nodes on the boundary, or that are trapped between the boundary and an interior edge with both endpoints on the boundary. To satisfy the conditions, we select all the tetrahedra incident on a set of "enveloped" nodes sufficiently interior to the zero isocontour. This guarantees that every tetrahedron is incident on at least one interior node, and also tends to avoid the bad interior segments for reasonably convex regions, i.e. regions where the geometry is adequately resolved by the nodal samples. We specifically choose the set of nodes where $\phi < 0$ that have all their incident edges at least 25% inside the zero isocontour as determined by linear interpolation of ϕ along the edge.

Additional processing is used to guarantee appropriate topology even in regions where the mesh may be under-resolved. Any remaining interior edges and all edges incident on non-manifold nodes are bisected, and the red green procedure is used to remove all T-junctions. If any refinement was necessary, we recalculate the set of enveloped nodes and their incident tetrahedra as above. As an option, we may add any boundary node with surface degree three to the set of enveloped nodes (if these nodes were to remain, the final surface mesh would typically contain angles over 120°). We also add any non-manifold node that remains and the deeper of the two boundary nodes connected by a bad interior edge. We check that these additions do not create more problems, continuing to add boundary nodes to the set of enveloped nodes until we have achieved all requirements. This quickly and effectively eliminates all topological problems, resulting in a mesh that approximates the object fairly closely (from the viewpoint of an initial guess for the compression phase of the algorithm) and that has connectivity well suited for large deformation simulations.

7 Physics Based Compression

We outfit our candidate mesh with a deformable model based on either masses and springs or the finite element method, and subsequently compress the boundary nodes to conform to the zero isocontour of the signed distance function. The compression is driven using either a force or velocity boundary condition on the surface nodes. Applying forces is more robust as it allows the interior mesh to push back, resisting excessive compression while it seeks an optimal state. If the internal resistance of the mesh becomes larger than the boundary forces, the boundary will not be matched exactly. Instead of adjusting forces, we switch from force to velocity boundary conditions after an initial stage that carries out most of the needed compression. At each boundary vertex, we choose the direction of the force or constrained velocity component as the average of the incident triangles' normals. No force (or velocity constraint) is applied in other directions so the mesh is free to adjust itself tangentially. The magnitude of the force or velocity constraint is proportional to the signed distance from the level set boundary.

For time integration we use the central difference scheme advocated in [Bridson et al. 2002] that treats the nonlinear elastic forces explicitly and the damping forces implicitly. This allows larger time steps to be chosen based only on the elastic (and not the damping) forces. Moreover, since all our damping forces are linear and

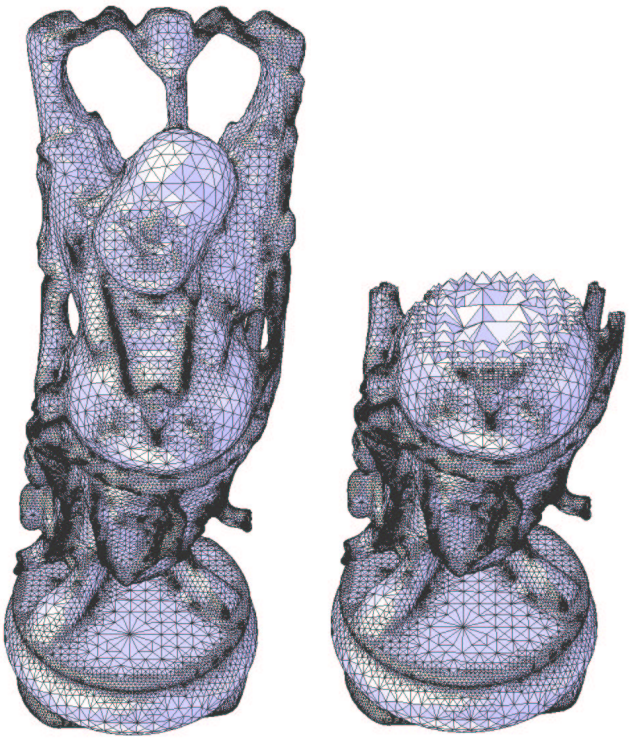

Figure 7: **Tetrahedral mesh of a model Buddha (800K elements). The principal curvatures provide refinement criteria that readily allow the resolution of features on multiple scales.**

symmetric negative semi-definite, we can use a conjugate gradient solver for the implicit step as in [Baraff and Witkin 1998]. We also limit the time step so that no tetrahedron altitude can deform by more than 10% during the time step as motivated by [Baraff and Witkin 1998; Bridson et al. 2002]. In addition, we use the velocity modification procedure discussed in [Bridson et al. 2002] to artificially limit the maximum strain of a tetrahedral altitude to 50% for both compression and expansion, and to artificially limit the strain rate of a tetrahedral altitude to 10% per time step. Since altitudes do not connect two mesh nodes together, all of these operations are carried out by constructing a virtual node at the intersection point between an altitude and the plane containing the base triangle. The velocity of this point is calculated using the barycentric coordinates and velocities of the triangle, and the mass is the sum of the triangle's nodal masses. The resulting impulses on this virtual node are equally redistributed to the triangle nodes, conserving momentum.

7.1 Mass Spring Models

The use of springs to aid in mesh generation dates back at least to [Gnoffo 1982]. [Bossen and Heckbert 1996] points out that internodal forces that both attract and repel (like springs with nonzero rest lengths) are superior to Laplacian smoothing where the nodes only attract each other. Thus, we use nonzero rest lengths in our springs, i.e. simulating the mesh as if it were a real material. All edges are assigned linear springs obeying Hooke's law, and the nodal masses are calculated by summing one quarter of the mass of each incident tetrahedron.

Edge springs are not sufficient to prevent element collapse. As a tetrahedron gets flatter, the edge springs provide even less resistance to collapse. Various methods to prevent this have been introduced, e.g. [Palmerio 1994] proposed a pseudo-pressure term, [Bour-

guignon and Cani 2000] used an elastic (only, i.e. no damping) force emanating from the barycenter of the tetrahedron. [Cooper and Maddock 1997] showed that these barycentric springs do not prevent collapse as effectively as altitude springs. In this paper, we propose a novel extension of their model for altitude springs by including damping forces which are linear and symmetric negative semi-definite in the nodal velocities. This allows the damping terms to be integrated using a fast conjugate gradient solver for implicit integration, which is important for efficiency especially if stiff altitude springs are used to prevent collapse.

Every tetrahedron has four altitude springs each attaching a node to its opposite face. Consider a particular node labeled 1 with nodes 2, 3, 4 on the opposite face with outward normal $\hat{n}$. If the current perpendicular distance is l and the rest length is l_0, then we add the elastic force $F_1 = k^e[(l-l_o)/l_o]\hat{n}$ to node 1 and subtract one third of that from each of the opposite face nodes. If the nodal velocities are $v_1,\ldots,v_4$, we add the damping forces: $F_1 = f_{12}+f_{13}+f_{14}$, $F_2 = -f_{12}$, $F_3 = -f_{13}$ and $F_4 = -f_{14}$ where $f_{1i} = k^d((v_1-v_i)\cdot\hat{n})\hat{n}$ for $i = 2,3,4$. Multiplying the f_{1i} by the barycentric weights of the triangle that determine the location of the virtual node at the base of the altitude spring gives $f = k^d((v_1-v_B)\cdot\hat{n})\hat{n}$ which is the usual damping force for a spring. Thus there is nothing special or unusual about the damping force from the viewpoint of the altitude spring. However, after calculating this force, it needs to be redistributed to the nodal locations at the vertices of the triangle. Our formulas do this so that the Jacobian matrix of damping forces with respect to the velocities is symmetric negative semi-definite.

[Van Gelder 1998] proposed scaling the spring stiffness as the sum of the volumes of the incident tetrahedra divided by the length of the edge. We provide a similar, but much shorter argument for scaling of this nature. The frequency of a spring scales as $\sqrt{k/ml_o}$ (note our "spring constant" is k/l_o) so the sound speed scales as $l_o\sqrt{k/ml_o} = \sqrt{kl_o/m}$. We simply require the sound speed to be a material property implying that k must scale as m/l_o. This result is consistent with [Van Gelder 1998], since the mass and the volume both scale identically with their ratio equal to the density. However, the mass is more straightforward to deal with since one can simply use the harmonic mass of a spring. For example, we set the spring constant for our altitude springs using the harmonic average of the nodal mass and the triangle mass.

7.2 Finite Element Method

It is well known that finite element methods are more robust than their mass spring counterparts, and thus we propose using the finite element method for mesh generation (as well as simulation). While any number of constitutive models could be used, an interesting strategy is to use the real constitutive model of the material when generating its mesh. In this sense, one might hope to predict how well the mesh will react to subsequent deformation during simulation, and possibly work to ensure simulation robustness while constructing the mesh.

We use the nonlinear Green strain tensor, $\varepsilon = 1/2[(\partial x/\partial u)^T(\partial x/\partial u) - I]$ where $x(u)$ represents a point's position in world coordinates as a function of its coordinates in object space. Isotropic, linearly-elastic materials have a stress strain relationship of the form $\sigma_e = \lambda tr(\varepsilon)I + 2\mu\varepsilon$ where λ and μ are the Lamé coefficients. Damping stress is modeled similarly with $\sigma_d = \alpha tr(v)I + 2\beta v$ where $v = \partial\varepsilon/\partial t$ is the strain rate. The total stress tensor is then $\sigma = \sigma_e + \sigma_d$.

We use linear basis functions in each tetrahedron so that the displacement of material is a linear function of the tetrahedron's four nodes. From the nodal locations and velocities we obtain this linear mapping and its derivative and use them to compute the strain and the strain rate, which in turn are used to compute the stress tensor. Finally, because the stress tensor encodes the force distribution inside the material, we can use it to calculate the force on the nodes as in for example [O'Brien and Hodgins 1999; Debunne et al. 2001].

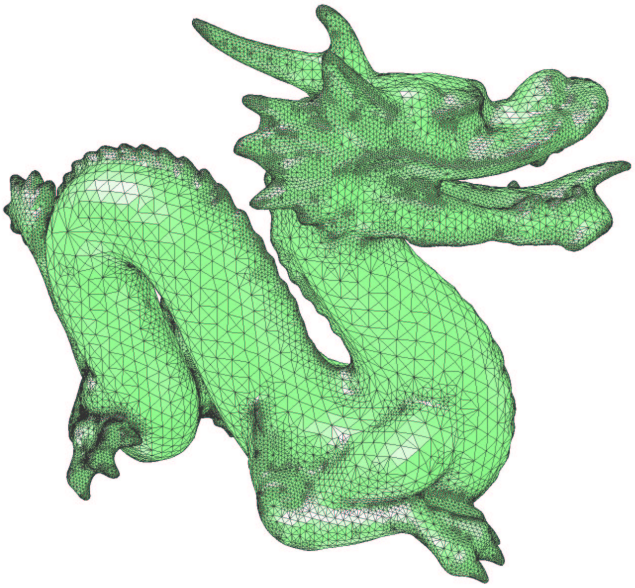

Figure 8: **Tetrahedral mesh of a model dragon (500K elements). The final compression stage can be carried out with masses and springs, finite elements, or an optimization based method.**

In their finite element simulation, [Picinbono et al. 2001] added a force in the same direction as our altitude springs. But since that force was the same on all nodes and based on the volume deviation from the rest state, it does not adversely penalize overly compressed directions and can even exacerbate the collapse. Moreover, they do not discuss damping forces. Therefore, we instead artificially damp the strain and strain rate of the altitudes of the tetrahedra as discussed above.

8 Optimization Based Compression

As an alternative to physical simulation, one can directly optimize mesh quality metrics such as aspect ratios. This doesn't provide the same feedback on potential problems for subsequent simulation, but can give better quality measures since they are directly pursued with each movement of a node. Coupled with our robust connectivity (see section 6), this produces excellent results.

[Freitag and Ollivier-Gooch 1997] demonstrated that optimizing node positions in a smoothing sweep, i.e. considering one node at a time and placing it at a location that maximizes the quality of incident elements, is far superior to Laplacian smoothing in three spatial dimensions. We combine this optimization sweeping with boundary constraints by first moving boundary nodes in the incident triangles' average normal direction by an amount proportional to the local signed distance value. Then the optimization is constrained to only move boundary nodes in the tangential direction. It is important to move boundary nodes gradually over several sweeps just as with physical simulation, since otherwise the optimization gets stuck in local extrema. We also found it helpful to order the nodes in the sweep with the boundary nodes first, their interior neighbors next, and so on into the interior. Then we sweep in the reverse order and repeat. This efficiently transfers information from the boundary compression to the rest of the mesh. Typically we do five sweeps of moving the boundary nodes $1/6$ of the signed distance in the mesh normal direction, and then $2/6,\ldots,5/6$. Finally, we perform five more sweeps moving boundary nodes the full signed distance to ensure a tight boundary fit.

While more efficient gradient methods may be used for the nodal optimization, we found a simple pattern search (see e.g. [Torczon 1997]) to be attractive for its robustness, simplicity of implementation, and flexibility in easily accommodating any quality metric. We implemented the normal direction constraint on boundary nodes simply by choosing pattern directions orthogonal to the mesh normal at the node.

It was helpful to combine incident boundary triangle quality with the incident tetrahedron quality, not just for producing better boundary meshes but also for guiding the optimization away from local extrema as we compressed the boundary. One promising avenue of research is to alternate optimization with physical simulation to further avoid local extrema and to better condition the mesh for the subsequent physical simulations.

9 Results

We demonstrate several examples of tetrahedral meshes that were generated with our algorithm. The number of tetrahedra ranges from 8k to 800K. We used mass spring models, the finite element method and optimization based techniques on all of our available data in order to discern some trends. The results are comparable for all three compression techniques, with the FEM simulations taking slightly longer (ranging from a few minutes to a few hours on the largest meshes) than the mass spring methods, but producing a higher quality mesh. The fastest method, by far, is the optimization based compression which tends to be about 10 times faster.

We track a number of quality measures including the maximum aspect ratio (defined as the tetrahedron's maximum edge length divided by its minimum altitude), minimum dihedral angle, and maximum dihedral angle during the compression phase. The aspect ratios of our candidate mesh start at about 3.5 regardless of the degree of adaptivity, emphasizing the desirability of our combined red green adaptive BCC approach. This number comes from the green tetrahedra whereas the red tetrahedra have aspect ratios of $\sqrt{2}$. In the more complicated models, the worst aspect ratio in the mesh tends to increase to around 6-7 for the physics based compression methods and to around 5 for the optimization based compression. Dihedral angles typically range from 15° to 150°, although one can often do better, e.g. the optimization based technique obtains angles between 19° and 145° for the cranium.

Of course, these results are dependent on the types and strengths of springs, the constitutive model used in the FEM, and the quality measures used in the optimization based technique. It is easier to get good quality with the optimization technique since one simply optimizes based on the desired measure, as opposed to the physics based techniques where one has to choose parameters that indirectly lead to a quality mesh. However, we stress that the measure of mesh quality *is* the measure of the worst element at any point of dynamic simulation. It does little good to have a perfect mesh that collapses immediately when the simulation begins. For meshes that undergo little to no deformation (fluid flow, heat flow, small strain, etc.) this quality measure is either identical to or very close to that of the initial mesh. However, for large deformation problems this is not the case, and the physics based compression techniques hold promise in the sense that the resulting mesh may be better conditioned for simulation.

The general theme of our algorithm—tile ambient space as regularly as possible, select a subset of elements that are nicely connected and roughly conform to the object, then deform them to match the boundary—is applicable in any dimension and on general manifolds. Figure 9 shows the result of triangulating a two-dimensional manifold with boundary. We began with a surface mesh of the dragon actually created as the boundary of a tetrahedral mesh from our method (with additional edge-swapping and smoothing), and a level set indicating areas of the surface to trim away. We kept a subset of the surface triangles inside the trimming level set and compressed to the trimming boundary, making sure to stay on the surface. Note that quality two-dimensional surface meshes with boundary are important for cloth simulation, especially when considering collisions.

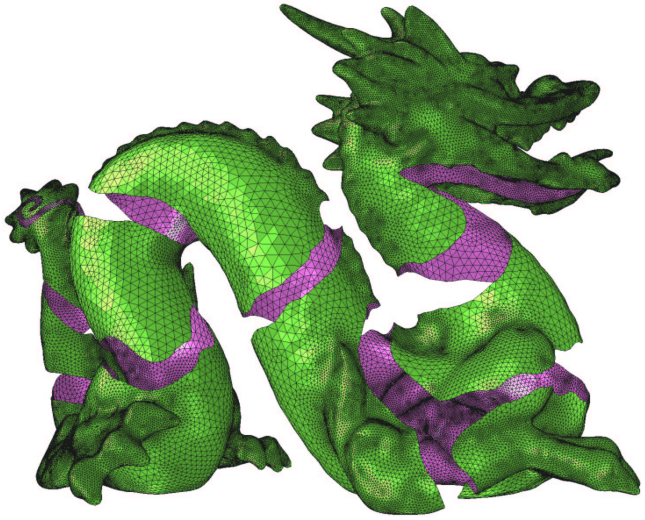

Figure 9: **Our method is easily extended to create a triangle mesh of a manifold with boundary. The peeled away regions of the dragon were modeled with a second level set.**

10 Conclusions and Future Work

Meshes generated using this algorithm have been successfully used to simulate highly deformable volumetric objects with a variety of constitutive models, see [Anonymous us 2003a], as well as two dimensional shells [Anonymous us 2003b]. This is not surprising because these meshes have withstood the fairly strenuous deformation of compressing them to the zero isocontour, where every surface node is subjected to external forces.

We considered discrete ways to improve the final mesh such as edge-swapping methods where an edge with N adjacent tetrahedra is replaced with $2N-4$ tetrahedra, see for example [Freitag and Ollivier-Gooch 1997; Joe 1995]. But in general, we avoid swapping in order to preserve our red green structure. Moreover, swapping tends to have a negligible effect on our mesh quality measures indicating that we have achieved near optimal connectivity with our red green adaptive BCC structure.

For simplification of tetrahedral meshes, edge collapse techniques can be used, see e.g. [Staadt and Gross 1998; de Cougny and Shephard 1999; Trotts et al. 1999; Cignoni et al. 2000] (see also [Hoppe 1996]), and we plan to pursue combinations of edge collapse with level set projection techniques as future work.

Boundary features and internal layers (as in [Cutler et al. 2002]) may be matched in our algorithm with additional forces or constraints.

References

AMBROSIO, L., AND SONER, H. M. 1996. Level set approach to mean curvature flow in arbitrary codimension. *Journal of Differential Geometry 43*, 693–737.

ANONYMOUS US. 2003. Finite volume methods for the simulation of muscle tissue.

ANONYMOUS US. 2003. Simulation of detailed clothing with folds and wrinkles.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. *Computer Graphics (SIGGRAPH Proc.)*, 1–12.

BERTRAM, M., DUCHAINEAU, M., HAMANN, B., AND JOY, K. 2000. Bicubic subdivision-surface wavelets for large-scale isosurface representation and visualization. In *Proceedings Visualization 2000*, 389–396.

BEY, J. 1995. Tetrahedral grid refinement. *Computing*, 55, 355–378.

BLOOMENTHAL, J., AND WYVILL, B. 1990. Interactive techniques for implicit modeling. *Computer Graphics (1990 Symposium on Interactive 3D Graphics) 24*, 2, 109–116.

BOSSEN, F. J., AND HECKBERT, P. S. 1996. A pliant method for anisotropic mesh generation. In *5th International Meshing Roundtable*, 63 – 76.

BOURGUIGNON, D., AND CANI, M. P. 2000. Controlling anisotropy in mass-spring systems. In *Eurographics*, 113–123.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Gr. (SIGGRAPH Proc.) 21*, 594–603.

BURNS, G., AND GLAZER, A. M. 1990. *Space groups for solid state scientists*. Academic Press, Inc. (2nd ed.).

CHENG, J. H., FINNIGAN, P. M., HATHAWAY, A. F., KELA, A., AND SCHROEDER, W. J. 1988. Quadtree/octree meshing with adaptive analysis. *Numerical Grid Generation in Computational Fluid Mechanics*, 633–642.

CHENG, S.-W., DEY, T. K., EDELSBRUNNER, H., FACELLO, M. A., AND TENG, S.-H. 2000. Sliver exudation. *Journal of the ACM 47*, 5, 883–904.

CIGNONI, P., COSTANZA, C., MONTANI, C., ROCCHIN, C., AND SCOPIGNO, R. 2000. Simplification of tetrahedral meshes with accurate error evaluation. In *Visualization*.

COOPER, L., AND MADDOCK, S. 1997. Preventing collapse within mass-spring-damper models of deformable objects. In *The Fifth International Conference in Central Europe on Computer Graphics and Visualization*.

CROSSNO, P., AND ANGEL, E. 1997. Isosurface extraction using particle systems. In *Proceedings Visualization 1997*, 495–498.

CUTLER, B., DORSEY, J., MCMILLAN, L., MÜLLER, M., AND JAGNOW, R. 2002. A procedural approach to authoring solid models. *ACM Trans. Gr. (SIGGRAPH Proc.) 21*, 302–311.

DE COUGNY, H. L., AND SHEPHARD, M. S. 1999. Parallel refinement and coarsening of tetrahedral meshes. *International Journal for Numerical Methods in Engineering 46*, 1101–1125.

DE FIGUEIREDO, L. H., GOMES, J., TERZOPOULOS, D., AND VELHO, L. 1992. Physically-based methods for polygonization of implicit surfaces. In *Proceedings of the conference on Graphics interface*, 250–257.

DEBUNNE, G., DESBRUN, M., CANI, M., AND BARR, A. H. 2001. Dynamic real-time deformations using space & time adaptive sampling. *Computer Graphics (SIGGRAPH Proc.) 20*.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. *Computer Graphics (SIGGRAPH Proc.)*, 317–324.

FREITAG, L., AND OLLIVIER-GOOCH, C. 1997. Tetrahedral mesh improvement using swapping and smoothing. *International Journal for Numerical Methods in Engineering 40*, 3979–4002.

FRISKEN, S. F., PERRY, R. N., ROCKWOOD, A. P., AND JONES, T. R. 2000. Adaptively sampled distance fields: a general representation of shape for computer graphics. *Computer Graphics (SIGGRAPH Proc.)*, 249–254.

GANOVELLI, F., CIGNONI, P., MONTANI, C., AND SCOPIGNO, R. 2000. A multiresolution model for soft objects supporting interactive cuts and lacerations. In *Eurographics*.

GARIMELLA, R., AND SHEPHARD, M. 1998. Boundary layer meshing for viscous flows in complex domains. In *7th International Meshing Roundtable*, 107–118.

GLOTH, O., AND VILSMEIER, R. 2000. Level sets as input for hybrid mesh generation. In *Proceedings of 9th International Meshing Roundtable*, 137–146.

GNOFFO, P. 1982. A vectorized, finite-volume, adaptive-grid algorithm for Navier-Stokes calculations. *Numerical Grid Generation*, 819–835.

GRINSPUN, E., KRYSL, P., AND SCHRÖDER, P. 2002. CHARMS: A simple framework for adaptive simulation. *ACM Trans. Gr. (SIGGRAPH Proc.) 21*, 281–290.

GROSSKOPF, S., AND NEUGEBAUER, P. J. 1998. Fitting geometrical deformable models to registered range images. In *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments (SMILE)*, 266–274.

GROSSO, R., LÜRIG, C., AND ERTL, T. 1997. The multilevel finite element method for adaptive mesh optimization and visualization of volume data. In *Visualization*, 387–394.

HIROTA, G., FISHER, S., STATE, A., LEE, C., AND FUCHS, H. 2001. An implicit finite element method for elastic solids in contact. In *Computer Animation*.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1993. Mesh optimization. *Computer Graphics (SIGGRAPH Proc.)*, 19–26.

HOPPE, H., DEROSE, T., DUCHAMP, T., HALSTEAD, M., JIN, H., MCDONALD, J., SCHWEITZER, J., AND STUETZLE, W. 1994. Piecewise smooth surface reconstruction. *Computer Graphics (SIGGRAPH Proc.)*, 295–302.

HOPPE, H. 1996. Progressive meshes. *Computer Graphics (SIGGRAPH Proc.) 30*, 98–108.

HORMANN, K., LABSIK, U., MEISTER, M., AND GREINER, G. 2002. Hierarchical extraction of iso-surfaces with semi-regular meshes. In *Symposium on Solid Modeling and Applications*, 58–58.

JAMES, D. L., AND PAI, D. K. 2002. DyRT: Dynamic response textures for real time deformation simulation with graphics hardware. *ACM Trans. Gr. (SIGGRAPH Proc.) 21*.

JOE, B. 1995. Construction of three-dimensional improved-quality triangulations using local transformations. *SIAM Journal of Scientific Computing 16*, 6, 1292–1307.

KASS, M., WITKIN, A., AND TERZOPOULOS, D. 1987. Snakes: Active contour models. In *International Journal of Computer Vision*, 321–331.

KOBBELT, L. P., VORSATZ, J., LABSIK, U., AND SEIDEL, H.-P. 1999. A shrink wrapping approach to remeshing polygonal surfaces. In *Eurographics*, 119–130.

LI, X., TENG, S., AND ÜNGÖR, A. 1999. Biting spheres in 3d. In *8th International Meshing Roundtable*, 85–95.

LOHNER, R., AND CEBRAL, J. 1999. Generation of non-isotropic unstructured grids via directional enrichment. In *2nd Symposium on Trends in Unstructured Mesh Generation*.

MILLER, J., BREEN, D., LORENSEN, W., O'BARA, R., AND WOZNY, M. 1991. Geometrically deformed models: A method for extracting closed geometric models from volume data. *Computer Graphics (SIGGRAPH Proc.)*, 217–226.

NEUGEBAUER, P., AND KLEIN, K. 1997. Adaptive triangulation of objects reconstructed from multiple range images. In *IEEE Visualization*.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. *Computer Graphics (SIGGRAPH Proc.)*, 137–146.

O'BRIEN, J. F., BARGTEIL, A. W., AND HODGINS, J. K. 2002. Graphical modeling of ductile fracture. *ACM Trans. Gr. (SIGGRAPH Proc.) 21*.

OHTAKE, Y., AND BELYAEV, A. G. 2002. Dual/primal mesh optimization for polygonized implicit surfaces. In *Proceedings of the seventh ACM symposium on solid modeling and applications*, ACM Press, 171–178.

OSHER, S., AND FEDKIW, R. 2002. *Level set methods and dynamic implicit surfaces*. Springer-Verlag. New York, NY.

PALMERIO, B., 1994. An attraction-repulsion mesh adaption model for flow solution on unstructured grids.

PICINBONO, G., DELINGETTE, H., AND AYACHE, N. 2001. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *IEEE International Conference Robotics and Automation*.

RADOVITZKY, R. A., AND ORTIZ, M. 2000. Tetrahedral mesh generation based in node insertion in crystal lattice arrangements and advancing-front Delaunay triangulation. *Computer Methods in Applied Mechanics and Engineering 187*, 543–569.

SADARJOEN, I. A., AND POST, F. H. 1997. Deformable surface techniques for field visualization. In *Eurographics*, 109–116.

SCHÖBERL, J. 1997. Netgen - an advancing front 2d/3d mesh generator based on abstract rules. *Computing and Visualization in Science 1*, 41–52.

SHEPHARD, M. S., AND GEORGES, M. K. 1991. Automatic three-dimensional mesh generation by the finite octree technique. *International Journal of Numerical Methods Engineering 32*, 709–739.

SHEWCHUK, J. 1998. Tetrahedral mesh generation by Delaunay refinement. In *Proc. Fourteenth Annual Symposium on Computational Geometry*, 86–95.

SHIMADA, K., AND GOSSARD, D. 1995. Bubble mesh: Automated triangular meshing of non-manifold geometry by sphere packing. In *ACM Third Symposium on Solid Modelling and Applications*, 409–419.

STAADT, O. G., AND GROSS, M. H. 1998. Progressive tetrahedralizations. In *Visualization*, 397–402.

SZELISKI, R., AND TONNESEN, D. 1992. Surface modeling with oriented particle systems. *Computer Graphics (SIGGRAPH Proc.)*, 185–194.

TORCZON, V. 1997. On the convergence of pattern search algorithms. *SIAM J. Opt. 7*, 1, 1–25.

TROTTS, I. J., HAMANN, B., AND JOY, K. I. 1999. Simplification of tetrahedral meshes with error bounds. *Visualization 5*, 3, 224–237.

TURK, G. 1991. Generating textures on arbitrary surfaces using reaction-diffusion. *Computer Graphics (SIGGRAPH Proc.) 25*, 289–298.

TURK, G. 1992. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH Proc.)*, 55–64.

VAN GELDER, A. 1998. Approximate simulation of elastic membranes by triangulated spring meshes. *Journal of Graphics Tools 3*, 2, 21–42.

VELHO, L., GOMES, J., AND TERZOPOULOS, D. 1997. Implicit manifolds, triangulations and dynamics. *Journal of Neural, Parallel and Scientific Computations 15*, 1–2, 103–120.

WESTERMANN, R., KOBBELT, L., AND ERTL, T. 1999. Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Visual Computer 15*, 2, 100–111.

WITKIN, A., AND HECKBERT, P. 1994. Using particles to sample and control implicit surfaces. *Computer Graphics (SIGGRAPH Proc.) 28*, 269–277.

WOOD, Z., DESBRUN, M., SCHRÖDER, P., AND BREEN, D. 2000. Semi-regular mesh extraction from volumes. In *Visualization 2000*, 275–282.

YAMAKAWA, S., AND SHIMADA, K. 2000. High quality anisotropic tetrahedral mesh generation via packing ellipsoidal bubbles. In *The 9th International Meshing Roundtable*, 263–273.