Removing Tetrahedra from a Manifold Mesh

C. Forest H. Delingette N. Ayache
Epidaure Research Project, INRIA Sophia Antipolis
2004 route des Lucioles, 06902 Sophia Antipolis, France
Clement.Forest@inria.fr, Herve.Delingette@inria.fr, Nicholas.Ayache@inria.fr

Abstract

One of the most important task in surgical simulation is the ability to cut volumetric organs. Several algorithms have already been described but none of them can actually maintain a specific and important topological property of the mesh called manifoldness*. In this article we define the notion of* manifoldness *and we explain why it is important to preserve it. We propose a new algorithm that maintains* manifoldness *and that implements a very simple cut strategy: the removing of soft tissue material, which is an efficient way to simulate the action of an ultrasound cautery. Finally we present experimental results which show the efficiency of this algorithm in a surgery simulation system.*

Keywords: Surgery Simulation, Tetrahedral Meshes, Interactive Cut, Manifoldness, Topological Singularities

1 Cutting in surgical simulation

Cutting is one of the most important gesture in surgery and most of surgical simulation systems offer or wish to offer cutting facilities for at least some of the represented organs [12]. Cutting has been quite extensively studied for surface models[16, 14] but has received little attention for volumetric meshes.

In a surgical procedure, *cutting* means two different actions: incising which is performed with a scalpel, and removing soft tissue materials which is performed with an ultrasound cautery.

Incising algorithms are the most commonly studied in the literature. Most of them are based on subdivision algorithms whose principle is to divide each tetrahedron across a virtual surface defined by the path of a tool cutting edge[1, 8]. Although they are very accurate in the rendering of the cut these algorithms suffer from two limitations. First, they tend to create a large number of small tetrahedra having a low shape quality which can be a major problem for the implementation of finite element models. Recently, Nienhuys[9] proposed a 3D adaptive mesh algorithm which tries to avoid the creation of tetrahedra by moving mesh vertices along the cut surface and by splitting additional triangles. However, this approach limits the number of additional vertices at the cost of creating tetrahedra of poor quality.

Second, these algorithms suppose that the cut surface created by the motion of a surgical tool is very smooth or is locally planar. However, in real-time surgical simulation systems, there are no constraints on the motion of surgical tools and most of the time the cut surface is not even close to be a smooth surface. Furthermore, surgical cutting gestures do not usually consist of a single large gesture, but are made of repeated small incisions. Therefore, these algorithms are not of practical use for cutting volumetric meshes, at least when simulating an hepatectomy.

Cutting by removing soft tissue material can be efficiently represented by a simple removal of tetrahedra and since it does not modify the quality of mesh elements, it is well suited for real-time soft tissue simulation. Although it usually reduces to a self-evident "`remove this tetrahedron from the list of tetrahedra`", it often requires that the size of removed tetrahedra is small enough to obtain a realistic cut and therefore it implies to manage meshes of relatively large size (especially when compared with the previous approach). In this article, we suppose that this problem is solved by using a mesh locally refined at parts where the cut occurs[11].

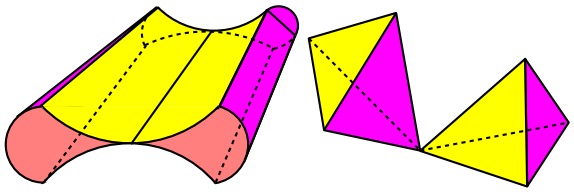

Figure 1. Example of non manifold objects

However, the obvious algorithm "`remove this tetrahedron from the list of tetrahedra`", cannot be used for realistic cutting simulation

because it does not constrain the mesh to be a manifold volume. The notion of manifold is defined in section 2 but Figure 1 gives an example of non manifold volumetric volumes.

2 Manifoldness

The notion of manifold surface or volume, or *manifoldness*, has been studied mainly in the case of triangulated surfaces [4] but very few papers have focused on volumetric tetrahedral meshes. This is mainly due to the fact that most authors are interested in either creating volumetric mesh from surface meshes[10] or in performing local remeshing where they suppose the initial mesh is a manifold. Only authors that have addressed the problem of cutting into tetrahedral meshes have reported problems while preserving the manifold nature of the mesh[9, 7] without solving it.

Manifoldness differs from conformality which is necessary for all of the finite elements algorithms, by adding a topological requirement that is the absence of singularities, as described in [2]. A mesh composed of two tetrahedra linked by a single vertex or a single edge are not manifold meshes.

Definition 2.1 *A mesh is a **conformal mesh** if*

- *all tetrahedra have a non-empty interior*
- *the intersection of two tetrahedra are either an empty set, a common vertex, a common edge or a common face.*

Definition 2.2 *A mesh is a **manifold mesh** if:*

- *it is conformal mesh*
- *all vertices have a neighborhood that is homeomorphic to a topological sphere for inner vertices and to a topological half-sphere for surface vertices.*
- *all edges have a connected adjacency.*

Manifoldness for a tetrahedral mesh is very important because it is a requirement for performing many low-level tasks including:

- **computing vertex normal** Normals are especially useful for the rendering of the mesh surface with Gouraud shading or with the PN triangles[15]. In the case of surgical simulation, vertices normals are also necessary to compute the mesh deformation and the reaction force sent to the force-feedback device.
- **mesh smoothing** Many algorithms such as Laplacian smoothing are based on finding all adjacent vertices to a given vertex. Finite element methods do not require manifoldness (but only conformality) although non manifold mesh often leads to ill-conditioned linear systems of equations.

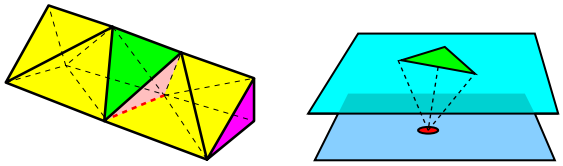

Figure 2. Removing the tetrahedron can cause a singularity on an edge (left) or on a vertex (right)

- **mesh subdivision** Subdivision surfaces and volumes [13] require a regular connectivity between vertices.

Finally and most importantly, manifold meshes allow to access all adjacency relationships between vertices, edges, triangles and tetrahedra with a minimum amount of information.

3 Cutting Algorithm

3.1 Problem position

Our surgical simulator is based on a Laparoscopic Impulse Engine[5] which simulates the handle of a laparoscopic instrument. The collisions between the tool and the organs are detected with an efficient hardware accelerated method [6] and several approaches based on an original biomechanical modeling of soft tissues have been implemented and tested [3, 11]. In the case of the ultrasound cautery all the tetrahedra detected close to the tool extremity are simply removed.

As demonstrated in section 4, it is frequent to find a tetrahedron that cannot be removed without creating a non-manifold mesh. Indeed, it can be easily shown that it is impossible to pierce right through the mesh (Figure 2) or to separate two connected components (Figure 2) by removing tetrahedra one by one and without ever creating any singularities.

3.2 Strategies for solving topological singularities

Several approaches can be designed to prevent topological singularities. For instance, when a potential topological problem has been identified, one could refine locally the mesh around this singularity and remove both the refined tetrahedron and the singularity neighborhood. Indeed, this approach is interesting because it always allows to remove the targeted tetrahedron, but it can not prevent future new singularities around newly created vertices thus leading to cascading refinements. Furthermore, it tends to create many new, small and poorly shaped tetrahedra.

A second possibility would be to split up vertices, edges and triangles around each singularity to remove adjacency

between elements. However, a split operation at a vertex can potentially causes singularities on any edge or vertex adjacent to the initial vertex, thus leading to cascading splits. Furthermore, by splitting triangles and vertices, we can easily create self-intersecting meshes which greatly reduce the visual realism of the simulation.

In fact, we have partially implemented these two approaches but we decided to abandon them because they lead to rather complex and computationally expensive solutions. It should be emphasized that the difficulty in removing topological singularities significantly increases from triangulated surface meshes to volumetric ones. For 2-manifolds for instance, singularities can be easily solved by splitting vertices and edges. As mentioned before, it is no longer the case for 3-manifolds as singularities can propagate far away from the initial tetrahedron.

The idea of the algorithm we propose in this article is the following: starting from a tetrahedron T, we provide a set $\mathcal{T}_{removable}$ of tetrahedra containing T which can be removed while preserving the manifold nature of the tetrahedral mesh. In some cases, this set is reduced to the initial tetrahedron, or, in very few cases, to an empty set when no local solution has been found.

We now describe in more details this new approach.

3.3 Testing removability

A basic component of the algorithm presented below is to test if a single tetrahedron or a set of tetrahedra can be removed without creating topological problems.

3.3.1 Case of a single tetrahedron

Testing the removability of a single tetrahedron is quite easy. This is done according to the number of faces of that tetrahedron which belong to the mesh surface. Let T be the tetrahedron to be removed.

- If T has no faces on the mesh surface (T is then inside the mesh), it can be safely removed iff none of its four vertices are part of the mesh surface.
- If T has exactly one face on the mesh surface, it can be safely removed iff the vertex opposite to that face does not belong to the surface.
- If T has exactly two faces on the surface (T makes an angle in the mesh surface), it can be safely removable iff the edge opposite to the two faces does not lye on the surface
- If T has three or four faces on the surface (case of a pyramid put down on the mesh case or of a single floating tetrahedron) it can always be removed.

3.3.2 Case of a set of tetrahedra

Testing the removability of a whole set of tetrahedra is much more complicated than for a single tetrahedron Basically, we have to check that the mesh surface that would result from the removal of that set is a 2-manifold (the neighborhood of all its vertices is simply connected and each edge is only adjacent to two and only two triangles). Let $\mathcal{T}$ be the set of tetrahedra to be removed. The test is performed in three stages:

1. retrieving the sets $\mathcal{T}_v$, $\mathcal{T}_e$ and $\mathcal{T}_t$ of all vertices, edges and triangles of the new surface that are adjacent to the set $\mathcal{T}$ of tetrahedra. This is done easily by skimming through of the tetrahedra of $\mathcal{T}$
2. testing each edge of $\mathcal{T}_e$ by scanning their neighborhood and checking that the latter has not been disconnected by one of the tetrahedra of $\mathcal{T}$.
3. testing each vertex of $\mathcal{T}_v$ by retrieving all the new surface triangles adjacent to each vertex and checking that triangles are connected through their edges in a single connected component.

3.4 Algorithm Description

Let T be the tetrahedron to be removed. We are trying to determine a set $\mathcal{T}_{removable}$ of tetrahedra that can be removed (while preserving manifoldness) such that $T \in \mathcal{T}_{removable}$. There are obviously many removable sets, the most trivial one being the whole mesh. Therefore, we need to give additional criteria for defining the optimal set $\mathcal{T}_{removable}$.

Our first criterion is to minimize the cardinality of $\mathcal{T}_{removable}$ (i.e. the number of tetrahedra to remove). Indeed, it is very reasonable to restrict as much as possible the side effect of preserving the manifold nature of the mesh. In term of surgery simulation, it simply tries to avoid the removal of much larger area by the lancet than expected. Note that we could have replaced this criterion by minimizing the overall volume of $\mathcal{T}_{removable}$ rather than its cardinality.

The second criterion consists in restricting the tetrahedra of $\mathcal{T}_{removable}$ to be adjacent to (i.e. to contain) one of the four vertices of T. The reasons are twofold. First, we want that $\mathcal{T}_{removable}$ to be located around T to enforce the visual realism of the cutting. Second, we want to limit the computation time of $\mathcal{T}_{removable}$ since we are targeting real-time applications. Unfortunately limiting spatially the search of $\mathcal{T}_{removable}$ implies in some cases the absence of local solutions. Figure 3 shows an example in 2D of the absence of solutions. In 3D, such cases are possible, though not very frequent in practice. When the search fails, we decide to return an empty set $\mathcal{T}_{removable} = \{\}$.

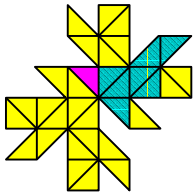

Figure 3. In blue, the minimum cardinality removable set containing the green triangle

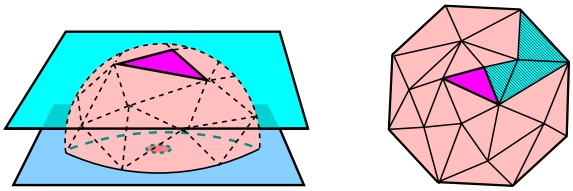

Figure 4. A vertex adjacency and the corresponding flattened map

3.4.1 Classification

The first step of the algorithm is to test the removability of tetrahedron T. Using the test described above, we can differentiate between three cases:

- The tetrahedron can not be removed and has exactly one face on the surface, the problem is then located on the vertex opposite to that face,
- The tetrahedron can not be removed and has exactly two faces on the surface, the problem is then located on the edge opposite to the two faces,
- The tetrahedron can be removed.

When the tetrahedron T cannot be removed, we apply two different algorithms depending on the singularity location.

3.4.2 Vertex singularity

A typical flattened map of the adjacency of such a problematic vertex is shown Figure 4 which can be seen as the representation of the planisphere seen by that vertex. The removal of the tetrahedron, in green in the figure, would create a hole in that planisphere and the vertex adjacency would no longer be homeomorphic to an half sphere. The idea is to suppress that hole by searching a path starting from the removed tetrahedra and going out of the vertex adjacency. This search is performed in a width-first manner in order to get the shortest path and is further arbitrarily limited to a depth of 10 in order to restrict the time spent in this search. When a path is found, its removability is tested and, on success the set $\mathcal{T}_{removable}$ is determined.

If the test fails, we use an additional heuristics. It appears to be often the case that the path we found, divides the adjacency of one of the tetrahedron vertices into two connected components. Therefore, we also test the removability of the given path extended with each one of those two connected components. If one of those two sets appears to be actually removable it is temporally stored and will be eventually used if no smaller removable set are determined.

If no paths are found to be removable then, we output an empty set $\mathcal{T}_{removable} = \{\}$.

3.4.3 Edge singularity

When the singularity is located at an edge, we test all tetrahedra surrounding that edge. Because, this edge is on the surface, we can divide these tetrahedra into two sets, corresponding to the leftmost and rightmost tetrahedra. We first test the removale for each of these two sets and in case none can be removed we also test if all tetrahedra adjacent to that edge can be removed. If not, we output an empty set.

This algorithm is very simple and can be implemented in a very efficient manner. We could also add more heuristics in case of failure to try to solve more configurations. However, because edge singularities do not occur very often, we have decided to keep it very simple.

4 Results

In order to test this method we have simulated the action of an ultrasound cautery on several meshes of different size, geometry and shape quality. At each of the simulation steps we try to remove the tetrahedra located next to the tool extremity. We then measure the percentage of resolved cases and the average cardinality of the removed set. To be more significant, if a non-removable tetrahedron is encountered several times it is only reported once.

The meshes that were used are:

1. A simple cylinder
2. A thin sheet with an average thickness of two tetrahedra
3. A rough mesh of a liver (3970 tetrahedra) obtained with a 3D-Mesher
4. A locally refined mesh of the same liver (8454 tetrahedra), obtained with a 3D-Mesher
5. A locally refined of a liver (9933 tetrahedra for a third of the liver) obtained with a home-made edge splitting method

The results appear to be quite stable from one mesh to the other and independent of the number of tetrahedra removed or of the shape quality of the meshing. The only differences appear with meshes showing strong topological singularity (thin sheet and cylinder).

- **Importance of the algorithm**

Mesh	nb. of operations	removable tetrahedra	pbs. on vertices	pbs. on edges
1	130	89.3%	9.2%	1.5%
2	265	62.3%	31.7%	6.0%
3	460	84.1%	13.5%	2.4%
4	1448	83.8%	14.4%	1.8%
5	2268	86.1%	13.3%	0.6%

Those figures show for each one of the meshes the number of attempted removing operations and the percentages of the latter which correspond respectively to a removable tetrahedron, to a tetrahedron whose removal would have caused a topologic problem on a vertex and to one that would have caused a problem on an edge. On average, 15% of the encountered tetrahedra could lead to non-manifoldness if removed. However, if those tetrahedra are not actually removed that ratio increases sharply. Among those 15%, 80-90% are due to singularities on vertices and 10-20% to singularities on edges

- **Algorithm on the vertices**

Mesh	simple path	extended path	non resolved	average card. of removed set
1	50.0%	41.7%	8.3%	3.1
2	63.1%	22.6%	14.3%	3.1
3	71.0%	22.6%	6.4%	2.9
4	64.4%	25.5%	10.1%	3.2
5	63.1%	21.3%	15.6%	3.3

A solution is found in about 85% of the cases and the average cardinality of the encountered sets is 3.3. The additional heuristic is also an important part of the methodology since it gives a solution in almost a quarter of cases.

- **Algorithm on the edges**

Mesh	half adjacency	whole adjacency	non resolved	average card. of removed set
1	0.0%	50.0%	50.0%	6.0
2	18.7%	37.5%	43.8%	4.7
3	18.2%	27.3%	54.5%	5.0
4	46.2%	19.2%	34.6%	4.0
5	42.9%	21.4%	35.7%	3.8

The algorithm we proposed for solving this case is not as efficient and solves only half of the encountered cases. However, this is only of few importance this case appearing in less than 2% of the removal operations.

- **Global figures**

Mesh	General		Problems	
	non solved	average card. of removed set	non solved	average card. of removed set
1	1.5%	1.2	14.2%	3.5
2	7.1%	1.8	19.0%	3.3
3	2.2%	1.3	13.7%	3.0
4	2.1%	1.3	12.8%	3.3
5	2.3%	1.3	16.5%	3.3

The algorithm solves more than 97% of the encountered cases, the removed set has an average cardinality of 1.4. Only considering problematic cases, those figures are respectively 85% and 3.3.

The ability to solve 97% of cases is satisfying for the purpose of surgery simulation. Indeed at each step of the simulation several (5 or 6) tetrahedra are tested for removal and in practice there is always at least one for which the removal can actually be performed.

References

[1] D. Bielser and M. H. Gross. Interactive simulation of surgical cuts. In *Proc. Pacific Graphics 2000*, pages 116–125. IEEE Computer Society Press, October 2000.

[2] J.-D. Boissonnat and M. Yvinec. *Algorithmic Geometry*. Cambridge University Press, UK, 1998. Translated by Hervé Brönnimann.

[3] S. Cotin, H. Delingette, and N. Ayache. A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. *The Visual Computer*, 16(8):437–452, 2000.

[4] A. Guéziec, G. Taubin, F. Lazarus, and B. Horn. Cutting and stitching: Converting sets of polygons to manifold surfaces. *IEEE Transactions on Visualization and Computer Graphics*, pages 136–151, 2001.

[5] Immersion Corp. Laparoscopic Impulse Engine webpage. http://www.immersion.com/products/custom/laproimpulse.shtml.

[6] J.-C. Lombardo, M. Cani, and F. Neyret. Real-time collision detection for virtual surgery. In *Computer Animation*, Geneva Switzerland, May 26-28 1999.

[7] C. Mendoza, C. Laugier, and F. Boux de Casson. Virtual reality cutting phenomena using force feedback for surgery simulations. In *Proc. of the IMIVA workshop of MICCAI*, Utrecht(NL), 2001.

[8] A. B. Mor and T. Kanade. Modifying soft tissue models: Progressive cutting with minimal new element creation. In *MICCAI*, pages 598–607, 2000.

[9] H.-W. Nienhuys and A. F. van der Stappen. Supporting cuts and finite element deformation in interactive surgery simulation. In W. Niessen and M. Viergever, editors, *Procs. of the Fourth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'01)*, pages 145–152. Springer Verlag, October 2001.

[10] F. P. and G. P.-L. *Maillages, applications aux éléments finis*. Hermes Sciencs, 1999.

[11] G. Picinbono, H. Delingette, and N. Ayache. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE International Conference Robotics and Automation*, Seoul Korea, May 2001. Best conference paper award.

[12] G. Picinbono, J.-C. Lombardo, H. Delingette, and N. Ayache. Anisotropy, interaction and extrapolation for surgery simulation. *Journal of Visualisation and Computer Animation*, 2001. to appear (accepted for publication).

[13] H. Qin. Fem-based dynamic subdivision surfaces. *8th Pacific Conference on Computer Graphics and Applications*, pages 184–191, October 2000. ISBN 0-7695-0868-5.

[14] D. Serby, M. Harders, and G. Székely. A new approach to cutting into finite element models. In W. Niessen and M. Viergever, editors, *Procs. of the Fourth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'01)*, pages 425–433. Springer Verlag, October 2001.

[15] A. Vlachos, J. Peters, C. Boyd, and J. Mitchell. Curved PN triangles. In *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, August 2001.

[16] G. Voss, J. K. Hahn, W. Muller, , and R. W. Lindeman. Virtual cutting of anatomical structures. In I. Press, editor, *Medicine Meets Virtual Reality (MMVR '99)*, January 1999.