Tissue Resection using Delayed Updates in a Tetrahedral Mesh

Kishalay KUNDU [a] and Marc OLANO [a]

[a] *Department of Computer Science, University of Maryland Baltimore County*

Abstract. In open surgery simulations, cuts like incisions and resections introduce irreversible changes to underlying geometry. In such circumstances, updating tetrahedral meshes for sophisticated physical modeling methods like finite elements becomes computationally intensive.

We present an algorithm that does not need to update every time there is an incision. It allows multiple incisions and only performs subdivision after resection. We will show that this leads to lesser subdivisions and increases the interactivity of the simulation.

Keywords. geometric model, surgical simulation, cuts

Introduction

Computer based simulation of surgeries provide safe and repeatable environments for surgeons and residents to hone their skills. One of the chief challenges of surgical simulation is accurate representation of deformable organs. Many existing simulations focus on minimally invasive procedures which avoid the need to cut or limit the types of cuts.

Simulating cuts is a difficult process in *open surgery* simulation because they change the underlying data. Simulation systems that use physics-based deformation methods like mass-spring or finite elements (FEM) are particularly vulnerable to this. Cuts force introduction of new nodes and recomputations of the stiffness matrix, which makes the system slow and non-interactive.

Many FEM-based simulations model deformable organs as a mesh of tetrahedra. Bielser [1] describes a geometric data structure that preserves state information for a tetrahedra-based FEM model. In their system, a tetrahedron being cut by a line tool exists in one of twenty-four states, based on the status of their cuts. Nienhuys and van der Stappen [3] describe a cutting algorithm that constantly deforms tetrahedra so that the cut-trajectory aligns with the tetrahedra face or edge. This method is reduces the need to introduce new nodes but can produce many degenerate tetrahedra.

Newer methods like XFEM, work around the node-growth problem by enriching nodal shape functions instead of introducing new nodes [4]. We present a geometric model that preserves cut information. We assume that our cuts are piecewise-planar (3D) instead of piecewise-linear (2D). Planar cuts do not trigger tetrahedral subdivision unless resection occurs, that is, a part of the mesh is completely severed from the rest. This reduces the rate of element-growth.

1. Methods and Tools

We have implemented a novel geometric data-structure where every tetrahedron maintains its state information including the number and position of cuts. The top level mesh maintains an overall information of the state of the mesh, like the number and state of non-contiguous cuts.

The mesh is divided into an octree-based hierarchical structure to improve the speed of collision detection. The top-level mesh maintains a set of cuts, each of which may be comprised of several polygons. A new cut is defined when a new incision occurs, or when an old cut changes trajectory and is no longer on its previous plane. Logical cut information is maintained by tetrahedra. Visual and haptic feedback are derived from these logical cuts.

Resection does not occur when a tetrahedron is cut through but only when a portion of the mesh is completely severed from the rest. When this happens, multiple cuts are merged and the affected tetrahedra are subdivided along the cut plane.

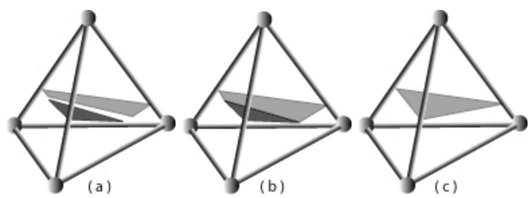

Figure 1. Merging cuts. Two separate cuts become one cut plane before subdivision

Figure 1 shows how two cuts can potentially merge into a single cut. It is to be noted that at the end of state (Figure 1c,) subdivision occurs only if the rest of the top-level cut causes resection. Our method leads to potentially fewer degenerate tetrahedra and thus provides stability to the overall system.

Bielser [1] proposed a state-machine-based tetrahedral cutting algorithm that maintains state information about tetrahedra and performs subdivision based on state changes. We have modified the concept to include multiple cuts and delayed subdivision based on mesh state instead of tetrahedron state.

2. Results

Figure 2 shows the various stages of cutting a tetrahedral mesh. Figure 2a shows the wireframe image of a tetrahedralmesh that is partially cut. While Bielser's method prompts tetrahedral subdivision at this stage, we suspend subdivision till later. Figure 2b shows our cut architecture's capacity to store multiple cuts. The bottom-right tetrahedron has been completely resected, yet remains intact.

Figure 2c shows complete mesh resection and tetrahedral subdivision. Figure 2d shows a partially cut mesh that is textured using a 3D texture map. Our lazy subdivision results in significantly lower tetrahedral subdivision for multiple cut procedures.

Our collision detection algorithm performed interactively at a rate of 28 fps with a mesh model of 1024 tetrahedra. We were able to sustain a maximum of 4 cuts, each of which spanned more than a quarter of the mesh, for the above model at a frame rate of 15 fps.

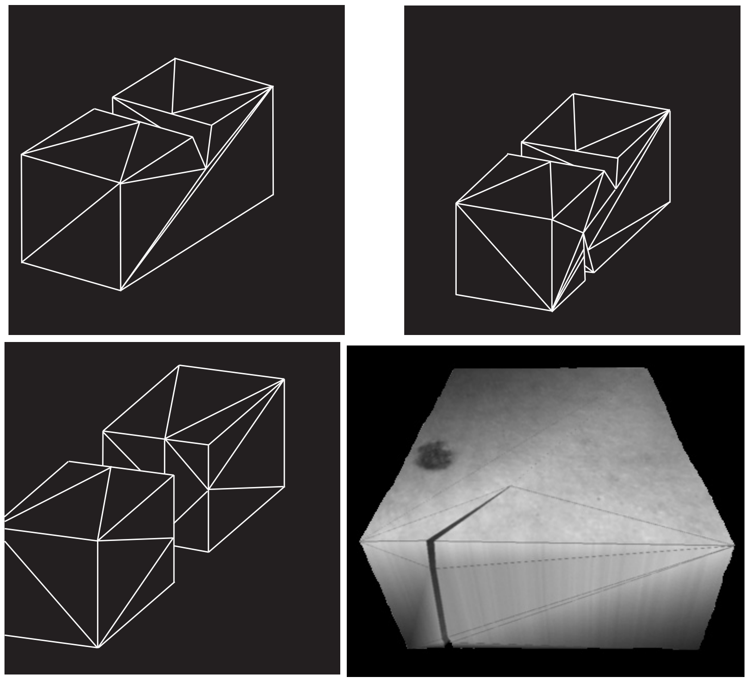

Figure 2. a. Single partial cut in mesh, b. Multiple partial cut, c. Resection and subdivision, d. 3D-texture mapped volume

3. Conclusion

We have designed and implemented a data-structure to reduce overall tetrahedral subdivision in open surgery simulations. This in turn, makes our system more robust and scalable. We do not restrict our cuts to be 2-dimensional. They may be planar in nature. We also allow multiple cuts to exist in the model. Subdivision only occurs in the event of resection.

In addition, our scheme is a logical fit with XFEM-based tissue modeling methods. We hope that these methods alleviate some of problems that traditional mesh-based models have had with cuts in open surgery surgical simulation.

References

[1] D. Bielser, *A framework for open surgery simulation*, Ph.D. thesis, Swiss Federal Institute of Technology, ETH, Zurich, 2003.

[2] N. Moës, J. Dolbow and T. Belytschko, A finite element method for crack growth without remeshing, *International Journal for Numerical Methods in Engineering*, **46**(1999), 131–150.

[3] H-W. Nienhuys and A.F. van der Stappen, Supporting cuts and finte element deformation in interactive surgery simulation, Technical Report, Universiteit Utrecht, 2001.

[4] L.M. Vigneron, J.G. Verly and S.K. Warfield, Modelling surgical cuts, retractions and resections via extended finite method, *Proceedings of the* 7^{t}*h International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 311–318, 2004.