Simulating cuts in triangulated objects

A. Frank van der Stappen
Universiteit Utrecht (joint work with Han-Wen Nienhuys)

April 4, 2003

The motivation of our work on simulating cuts is formed by interactive surgery simulations. Such simulations combine interactive deformation of virtual tissue with simulations of surgical procedures. The Finite Element Method (FEM) seems suited for computing deformation since it is physically accurate. Meshes form the basis of this technique: the domain of the problem is subdivided in geometric primitives ('elements') such as tetrahedra or triangles. With this subdivision the governing partial differential equation (PDE) can be transformed in a system of numerical equations. For the rest of the discussion, we will assume that the PDE is linear and derives from an elasticity problem. The equations can then be discretized into a linear system, where the matrix is called stiffness matrix.

Finding a solution is a two step process: first, the stiffness matrix is constructed, then the unknowns (a set of deformations) are computed from the stiffness matrix and given loads. The characteristics of a mesh influence both steps. The accuracy of the discretization process is determined by element shapes. For triangular/tetrahedral meshes, large elements and large angles decrease the accuracy of the FEM discretisation. The accuracy of the solution process in finite precision arithmetic is determined by the condition number of the stiffness matrix, where a lower condition number is better. The convergence speed of iterative solution methods also depends on the condition number. This holds for both the conjugate gradient algorithm (a static approach), and for damped dynamic relaxations using explicit time integration. The condition number of a complete stiffness matrix can be bounded by the condition numbers for separate elements. In contrast to the FEM discretization error, high condition numbers are caused by elements with small sizes and small angles. So, the requirements for optimal accuracy in the discretisation and for optimal speed and accuracy in the numerical process are contradictory. However, it is safe to say that a mesh with 'round' (non-flat) elements and bounded element sizes is a good general purpose mesh.

Iterative algorithms seem well suited as numerical solution strategies for surgery simulations, since they allow on-line meshchanges. If such an iterative approach is used, then mesh quality becomes an issue of vital importance: the maximum size of simulations is largely determined by mesh characterics: better meshes yield faster convergence, enabling larger and more accurate simulations. It follows that simulated surgical manipulations (such as needle insertions, cuts, and cauterizations) should be designed to keep mesh quality high. Secondly, the manipulations should also keep the complexity of the mesh low, since the cost of a single iteration step is proportional to the size of the mesh.

We state the general mesh cutting problem as follows: given a starting mesh, and positions of a user-controlled scalpel, modify the mesh at every moment to show an incision that represents the past trajectory of the scalpel, and ends exactly at the scalpel. The challenge is to do so while maintaining the quality and size of the mesh.

Simulation of cuts in surgery simulation is related to simulation of other destructive surgical procedures. The first operation to have been simulated on volumetric meshes is cauterization. This was done by removing elements in contact with a virtual cauterization tool. A disadvantage when applied to cutting is that it produces a jagged surface on the virtual tissue.

For cutting, subdivision methods have been the norm: elements that are in contact with the scalpel are subdivided to produce a cut conforming to the scalpel position. Subdivision methods always increase the size of the mesh. Moreover, these methods tend to produce degeneracies: mesh modification is done only within a fixed region of the mesh, and if the scalpel moves close to the boundary of that region, poorly-shaped elements are inevitable. Research has been done to counter the the degeneracies—caused by subdivision cutting—by collapsing short edges of the mesh. This approach does improve the quality of the mesh, but this solution does not repair all inconsistencies: not all edges may be contracted, and flat triangles and tetrahedra, which do not contain short edges but are still degenerate, are not dealt with.

Our first approach to cutting in tetrahedralized objects keeps the mesh size at the initial level by performing cuts only along faces of the mesh, which are selected according to a heuristic. Nodes of the mesh are relocated to align these faces with the trajectory of the virtual scalpel. We have implemented this approach and the results confirm that the mesh size remains small. In addition, the method also creates only few short edges. There are, however, also some disadvantages. Since no new nodes are created, the resolution of the cut is bounded by the mesh resolution, which results in a less accurate representation of the cut in the unlikely case of sharp turns in the scalpel path. A more serious problem is that snapping can result in degenerate elements in the mesh. Such degeneracies are dealt with by subdividing flat elements, and collapsing the resulting short edges, effectively removing the flat element. Unfortunately, not all edges can be collapsed. Using only existing mesh features as a basis for mesh modification is problematic: when the scalpel is not especially close to a mesh feature, it may not be possible to match the mesh topology to the scalpel path without introducing degeneracies.

Our second approach applies to triangulated surfaces. The technique could be applied to surgery simulation for membrane-like structures, such as skin or intestine. The method keeps the mesh size and quality at the initial level through a careful scheme of edge flips, node deletions, and node insertions. When cutting, a node of the mesh is attached to the virtual scalpel. Roughly speaking, our method deletes nodes that are to close to this (moving) active node to prevent degeneracies, it flips edges around the active node to increase the mesh quality, and inserts nodes behind the active node to adequately approximate the past trajectory of the scalpel. We have implemented our approach and it turns out that it maintains the mesh quality and keeps the mesh size low, without introducing degeneracies.

Finally, we report some initial results on our approach to needle insertion.