Algorithms for Cutting Surfaces Composed by Simplex Meshes

Luis Cruz and Luis G. de la Fraga

Cinvestav. Computer Science Departament.
Av. Instituto Politécnico Nacional 2508. 07300 México, D.F.
fraga@cs.cinvestav.mx

Abstract. In this work we develop three new operators to cut a volume surface composed by a simplex mesh that envelops such volume. These operators are based on remove the cut edges and reorganizing the mesh connectivity which, in turn, implies inserting new edges and reordering the faces' edges and edges' vertices. In order to test these operators we perform two simulations: in the first one a cylinder is cut, and in the second simulation a non-linear deformation with rupture is calculated over a sphere, thus the sphere is broken when are applied external forces. We also discuss the non-linear deformation model that we used in the second experiment.

Keywords: surface cutting, simplex meshes, surface deformation, non-elastic model, physically based simulation.

1 Introduction

The simplex meshes are a relatively new form to represent surfaces. Using simplex meshes allows some operators [1] for gluing other simplex meshes to construct more complicated models; such operators only change the links among the different elements that composes the mesh (vertices, edges and faces).

In this work three new operators are introduced, the *split operator*, that changes the genus of the mesh (i.e. the number of "holes" in the model) and outputs two distinct meshes; the *connectivity operator*, that checks if the mesh is already split; and the *cutting operator*, which, in turn, removes an edge from the mesh and re-orders its vertices and faces (and it possibly modifies the mesh genus). In order to implement these operators, we develop four algorithms.

Also, we apply external forces to deform the mesh using a non elastic deformation model over every edge in the mesh. In previous works [1,2] only was used a completely elastic model to deform a simplex mesh. We develop a non-elastic model with rupture based on a linear function and an exponential part which acts as the plastic region for the material, and with the rupture part itself.

To test our results we present two experiments: for the first one, a cylinder is built and some edges are removed until the model is split in two halves and enables the manipulation by the user of each part, one independently of the

A. Gelbukh, S. Suárez, H. Calvo (Eds.)
Advances in Computer Science and Engineering
Research in Computing Science 29, 2007, pp. 90-99

Received 30/06/07
Accepted 19/10/07
Final version 24/10/07

other; the second experiment uses the non-elastic model in a sphere, an internal force is applied internally to deform it until the sphere is broken.

2 Previous Works

The representation of a volume's surface has been done with one of three kinds [3]: a triangular mesh [4], implicit representations (such as splines or hyperquadrics), and simplex meshes [1,5,6]. The research had been primarily concentrated on the first two, leaving the simplex meshes to a side method.

Principally the research with simplex meshes had been developed by Delingette [7], fundamentally to model some human [8,2] and mouse [9] organs. Delingette proposed some operators to modify the vertex or the edges connectivity to obtain a new mesh gluing basic forms.

Some research was performed about the problem of cutting the volume tetrahedrization, but not had used the simplex mesh from its surface [10]. Another work had been conducted towards triangle meshes, modeling the 3D surface and its properties and manipulation [9].

3 Description of the new operators

A simplex mesh has a constact connectivity. In this work we use a 2-simplex mesh in which every vertex has only three neighboring vertices.

Now, we will describe the algorithms which conforms the developed operators, the algorithms 1 and 3 are used by the edge remove operator (and its corresponding faces), the algorithm 2 represents the connectivity operator, and the algorithm 4 represents the split operator.

The algorithm 1 shows the steps taken to remove one edge from the mesh. In Fig. 1 we can see the used notation.

Algorithm 1 Edge remove

Require: A simplex mesh (S), and the edge to remove (A)
Ensure: Removed adjacent faces to the removed edge
Find out the E's adjacent vertices (V_1 and V_2)
Find out the V_1 and V_2's adjacent vertices (V_3, V_4, V_5 and V_6)
Remove adjacent mesh faces to E (F_1 and F_2) using algorithm 2
Verify if there is some special case, if so, proceed to normalize the edges, so there is not a triangle adjacent to the edge E
Remove E
Add the two new edges (E_1 and E_2)
Modify the opposite faces to E (F_3 and F_4)
Remove V_1, V_2, E_3, E_4, E_5 and E_6

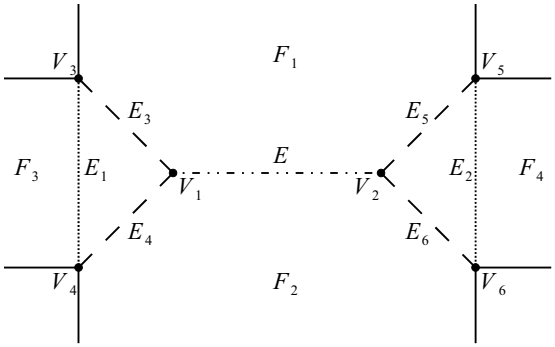

Fig. 1: Notation used in algorithm 1, F_1 to F_4 are the faces involved in the process, E the edge to be removed, E_1 and E_2 the new edges to be added, E_3 to E_6 the edges that need to be removed, V_1 and V_2 the vertices to be removed and V_3 to V_6 the vertices involved in the process but not removed

Algorithm 2 checks if the connectivity is preserved between two mesh's edges. This algorithm is used whenever an edge is removed from the mesh (it cuts from the geometrical model) to verify if the iterative process stops. A model who has been split in two parts will generate two different meshes and each one will be independently iterated.

Fig. 2 shows the edges where it is necessary to check the connectivity (these edges are E_1 and E_2). If both edges are conected then there is a path between them. For cheking this path one vertex from the edge E_1, say V_2, is chosen. Then the algorithm runs going in the next order: to the edge E_3, vertex V_3 is chosen, go to the edge E_4, to vertex V_4, to the edge E_5, to vertex V_5 and, finally, to the edge E_2, therefore a path exists and E_1 and E_2 are connectted.

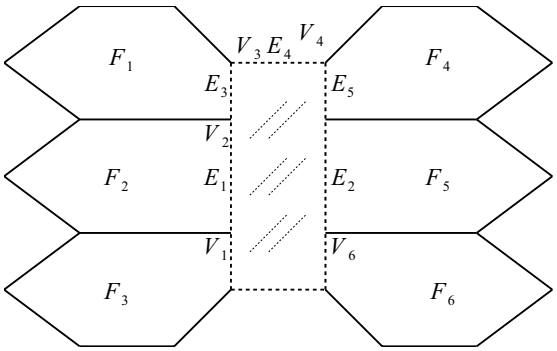

Fig. 2: Following edges to know if the mesh is already split

Algorithm 2 Verify the connectivity between edges

Require: Simplex mesh's edges (S^E) and the two edges to test (E_1 and E_2)
Ensure: Verify if the mesh was split in two parts
  Set E to edge E_1
  Choose a vertex, V_1, from E
  repeat
    Find the opposite vertex from E, V_2
    Find the adjacent edge to V_2, which does not have a face and is not E
    Set this edge to E
    Set V_2 to vertex V_1
  until Reach the edge E_1 or E_2
  if Reached the edge E_2 **then**
    The mesh was not split
  else
    The mesh was split
  end if

The algorithm 3 shows how the face between two meshes is deleted. Fig. 3 shows the notation used, and Fig 4 shows the mesh after the face has been removed, the mesh could be split in two separate meshes, at that point we use algorithm 2 to check this fact. Finally, if the mesh is already disconnected, we use algorithm 4 to compute the two new meshes.

Algorithm 3 Remove jointed face

Require: Simplex mesh (S, faces $-S^F-$, edges $-S^E-$ and vertices $-S^V-$) and the face to remove (F)
Ensure: Removed face and one, or two disjointed, meshes
  Remove the face edges (E_1 and E_2) which are adjacent to empty faces (faces already removed before before)
  Use algorithm 2 to verify if the connectivity is preserved, using the new created edges
  if Connectivity was lost **then**
    Use algorithm 4 to get the new two meshes
  end if

The algorithm 4 shows how we get two disconnected meshes from a single one. This algorithm takes a complexity order of $O(nm)$, where n is the number of faces in the original mesh (S^F) and m is the mean edges in each face (once every time approximately 6) because it processes each face and edge.

Three special cases must be treated separately for the case when a edge is deleted. These cases always involve triangles in the mesh (see Fig. 5), and the algorithm collapses in triangles until it finds a configuration without them, or get an empty mesh.

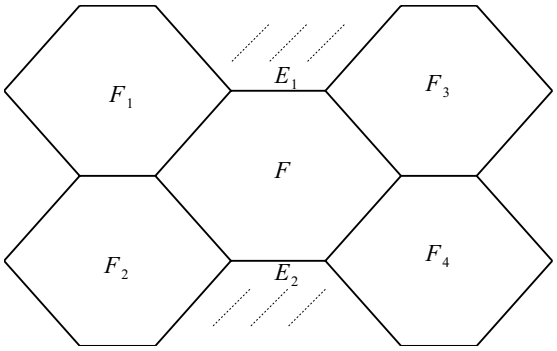

Fig. 3: The mesh with the face to be removed (edges E_1 and E_2 will be deleted also)

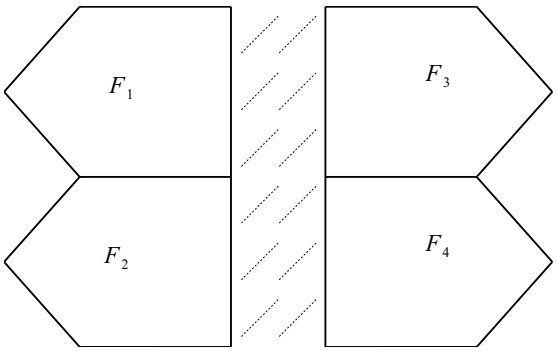

Fig. 4: The mesh without the deleted face

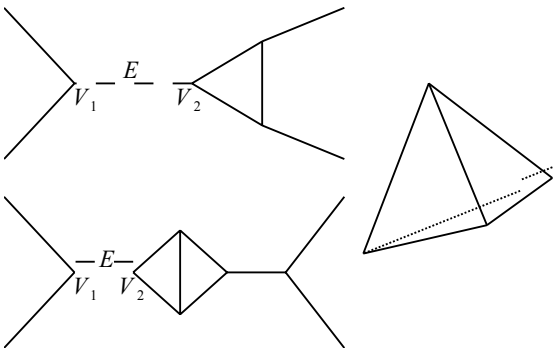

Fig. 5: The three special cases in the algorithm to remove an edge

Algorithm 4 How to get two disjoint meshes

Require: Simplex mesh (S, faces $-S^F-$, edges $-S^E-$ and vertices $-S^V-$)
Ensure: Compute one or more disjoint meshes
  Let L^F be a face's empty set, L^E an edge's empty set, and L^V a vertex's empty set
  Set all faces of S^F as *non-processed*
  while There are non-processed faces **do**
    Borrow the next non-processed face, F_i, from S^F
    Put face in a new face list, $L^F_{F_i}$, do the same with its edges, $L^E_{E_i}$, and its vertices, $L^V_{V_i}$
    Put face in the processing queue, P_f
    while The processing queue, P_f, is not empty **do**
      Get the next face, F_j, from the processing queue, P_f
      for all Adjacent face to F_j, that are not in any list ($F_{j,k}$, edges, $E_{j,k}$ and vertices, $V_{j,k}$) **do**
        Put $F_{j,k}$ in the F_j list ($L^F_{F_j}$)
        Put $E_{j,k}$ in the E_j list ($L^E_{E_j}$)
        Put $V_{j,k}$ in the V_j list ($L^V_{V_j}$)
        Put $F_{j,k}$ in the processing queue, P_f
      end for
    end while
  end while

4 Experiment cutting a cylinder

We used a cylinder to test our new operators, this cylinder is show in Fig. 6. The cutting process starts by removing the first edge from the cylinder's central strip (see the upper left of Fig. 7), the algorithm 1 is applied to delete the edge and refresh the vertex coordinates and the configuration of the affected faces.

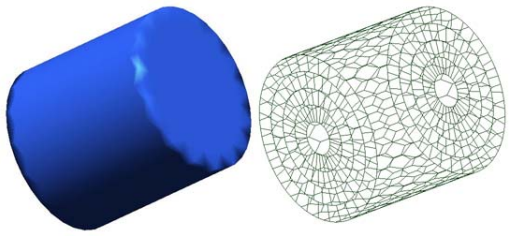

Fig. 6: Original cylinder

Once the first edge has been removed, we proceed to remove the second edge (see upper right of Fig. 7), here two faces were removed and one of them is adjacent to the intended edge, so this fact must be taken into account to avoid delete the non-existent face. Also, the corresponding vertices have one less adjacent face and it is necessary to take care of that, indicating that the face is

no more in its list of neighbor faces. In Fig. 7 we can see one thin central section produces by the cutting process, this is a consequence of the process itself and, also, of the vertex moving to the barycenter of its three neighbors.

The process goes ahead, deleting edges until the last one from the central strip must be removed; then the mesh clearly must be split in two parts, as can be seen in lower left Fig. 7.

Once the mesh has been cut in two meshes, it is possible to independently manipulate both (see lower right of Fig. 7).

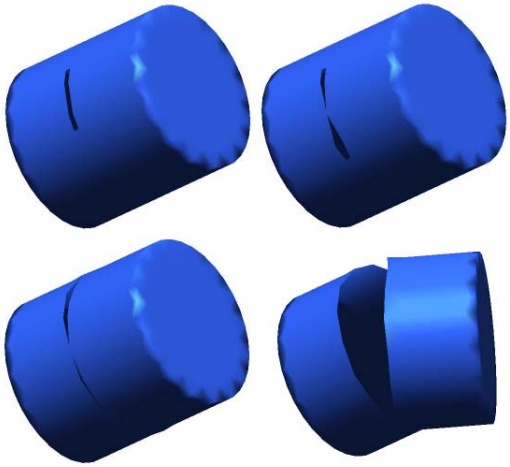

Fig. 7: The cutting process

5 Deformation model with rupture and second experiment

To deform a simplex mesh we attach a simple mechanical system composed by a mass, a spring, and a dashpot, to each mesh edge. This system is represented by Eq. ((1)):

$$m\ddot{x} + b\dot{x} + kx = f_{\text{ext}} \tag{1}$$

where m is the mass, b is called damping coefficient, and k is the spring stiffness coefficient. Eq. ((1)) is solved numerically to obtain the elongation x by using the finite differences method, because it takes less operations and its result is enough for our purposes. The system is in a steady stable at rest, so there are not external (f_{ext}) or internal forces applied to it.

This elastic model represented by Eq. (1) models a perfectly elastic material, i.e., a material whose graph of elasticity is linear. We develop a non-linear model using a completely inelastic material, with one segment linear, another exponential, and if eventually the force is too much, the material breaks. The

linear model is represented $f_{\text{int}} = k \cdot x$, where f_{int} is the internal force stored in the spring. The non-elastic model is:

$$f_{\text{int}}(x) = \begin{cases} k \cdot x & \text{if } x \leq l_1, \\ k \cdot l_1 + e^{\gamma(x-l_1)} - 1.0 & \text{if } l_1 < x \leq l_2, \\ 0 & \text{if } l_2 < x. \end{cases} \tag{2}$$

where γ is a parametric value (if we want a soft transition between the linear and exponential parts γ must be equal to k, so the first derivative in the point where $x = l_1$ would be equal), l_1 is the desired elongation limit (which depends on γ) and l_2 is the elongation at which the edge breaks.

At Fig. 8 we show, when the value of $\gamma = 0.05$, how the l_1 affects the maximum elongation the element can take (the model parameters are $m = 0.01$, $d = 0.01$, $k = 0.01$, $f_{\text{ext}} = 0.01$), as we can see with this value of γ the maximum elongation is effectively limited, but, is greater than l_1, if we increase γ this value would be closer to l_1.

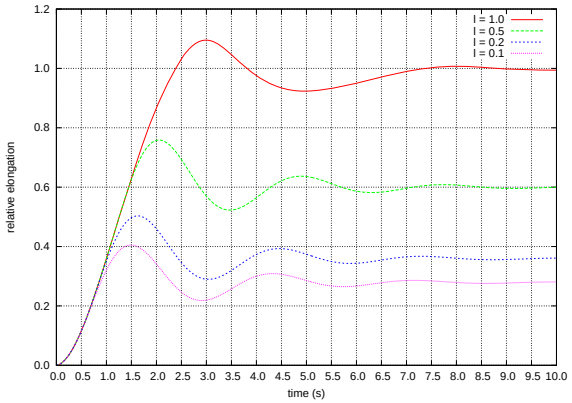

Fig. 8: Comparing the deforming limits at different values of l_1

The break limit (l_2) is defined as a factor that represents the maximum length that an element can have without rupture, at that length the element breaks; that factor represents the maximum stress the element can sustain in the point of rupture, above which the internal force is zero.

The second experiment is apply an internal force, as an internal pressure, to a deformable sphere until it breaks. The sphere used is shown in Fig. 9. In Fig.e 10 we can see the sphere when it breaks and the deforming process is stopped. We can see when the applied force deforms the model more or less in a symmetrical way, except at six rectangles located in the middle strip of the sphere.

The external force in each vertex is computed based on the influence area of it, as shown in the Eq. ((3)), where ρ is a constant which represents the inside

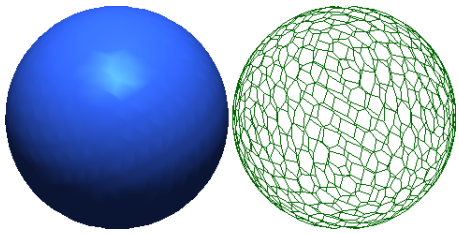

Fig. 9: Original sphere model

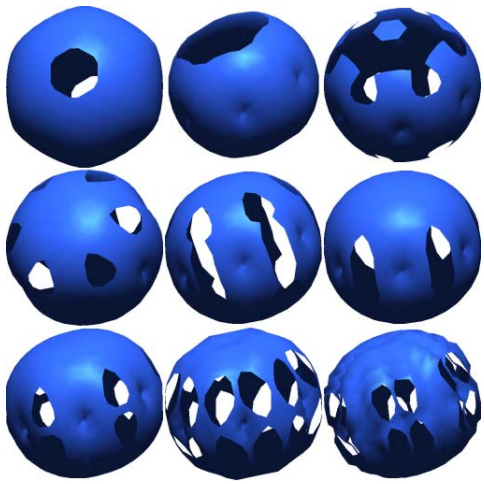

Fig. 10: The model when the internal pressure breaks it.

pressure, $\mathcal{A}_v$ is the influence area of the vertex v and $\mathcal{A}_T = \sum_{\forall v} \mathcal{A}_v$. Each value of ρ yields a different breaking pattern, as shown in Fig. 10, where the values are 8.7, 15.2, 16.2, 16.4, 17.3, 17.4, 23.0, 50.0 and 150.0, respectively.

$$f_{ext} = \rho \frac{\mathcal{A}_v}{\mathcal{A}_T} \tag{3}$$

6 Conclusions and future work

We have developed three new operators that allow cut volumes built with a simplex mesh. The first operator computes if a simplex mesh has lost its connectivity, the second operator split a simplex mesh in two halves, and, the third operator deletes an edge from the mesh and their corresponding faces.

The way we compute the simplex mesh connectivity has an acceptable computational complexity level, because we only check if it is keep between the two new added edges, so it only has to check a small set of edges. The split operator case needs to operate over all the simplex mesh, because we have no information about the faces connectivity, only its adjacency, so we could not optimize it well,

but we are trying to develop a new algorithm to directly split the mesh when removing the edge, without travel through all the faces.

The described operators have deficiencies (the cutting operator needs to take care of the face's and vertices' ordering, the connectivity operator needs to operate in edges with faces removed, and the split operator needs only a performance boost), over which we are already working and there is hope we get a complete definition of them, so they could operate in whatever simplex mesh we throw at it.

Our non–elastic deformation model has a good behavior as we expected, when it is applied to a surface simplex mesh.

As future work, we will extend operators to cut volumetric models based also on simplex meshes, so they can operate in this new domain, and, apply this same elastic model to it to see if it fits nicely in such structure or needs considering some other features to shows a well behavior.

We are thinking in apply the developed work to visualize and to simulate human organs in surgery processes.

References

1. H. Delingette. General object reconstruction based on simplex meshes. *International journal of computer vision*, 32(2):111–146, September 1999.
2. H. Delingette and N. Ayache. Hepatic surgery simulation. *Communications of the ACM*, 48(2):31–36, February 2005.
3. J. Montagnat, H. Delingette, and N. Ayache. A review of deformable surfaces: topology, geometry and deformation. *Image and vision computing*, 19(14):1023–1040, December 2001.
4. Jiuxiang Hu, Anshuman Razdan, Gregory M. Nielson, and Gerald E. Farin. Improved geometric constraints on deformable surface model for volumetric segmentation. *IEEE Geometric modeling and processing*, 04:237–248, 2004.
5. H. Delingette and J. Montagnat. General deformable model approach for model based reconstruction. *IEEE international workshop on model-based 3D image analysis*, 98, January 1998.
6. H. Delingette and J. Montagnat. Shape and topology constraints on parametric active contours. *Computer vision and image understanding*, 83(2):140–171, September 2001.
7. J. Montagnat and H. Delingette. Volumetric medical image segmentation using shape constrained deformable models. *Proceedings of computer vision, virtual reality and robotics in medicine*, 97:13–22, March 1997.
8. J. Montagnat and H. Delingette. Globally constrained deformable models for 3d object reconstruction. *Signal Processing*, 71(2):173–186, December 1998.
9. G. Hamarneh, H. Delingette, and M. Henkelman. 3d segmentation of mouse organs from mr images using deformable simples mesh models. *Proceedings of the International Society of Magnetic Resonance Medical*, page 779, 2003.
10. Stéphane Cotin, Hervé Delingette, and Nicholas Ayache. A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. *Visual Computer journal*, 16(8):437–452, 2000.