ICRA '03, Taipei (Taiwan), Sept 2003

Cutting using Haptics and Non Linear Finite Element Models

C. Mendoza, C. Laugier
INRIA Rhône-Alpes & GRAVIR , SHARP Project,
ZIRST-655, Avenue de l'Europe, 38330 Montbonnot Saint Martin, FRANCE.
Email: [Cesar.Mendoza-Serrano, Christian.Laugier]@inrialpes.fr

Abstract—**This paper presents a methodology to simulate cuts in deformable objects. It proposes a simple physical model of cutting in combination with a large displacement (Green) strain tensor formulation. Explicit finite elements are used to allow real-time simulations and fast topology updates during cutting procedures. A geometrical and haptic approaches are also proposed and merged to the physical cutting model.**

Keywords— **Cutting, medical simulators, physical models, deformable objects, virtual reality, haptic interaction.**

I. Introduction

CUTTING simulations of human tissue is an essential component in virtual reality surgery simulators. Such simulations require real-time interactions, accurate geometric models of anatomical structures and realistic dynamic models. Including haptic technology gives the surgeon the sensation of touching and manipulating the simulated virtual organs. However many problems are found while trying to implement cutting in a surgery simulator : (a) obtaining physically realistic models of the different human organs that can be simulated in real time, (b) changing the complexity of the topology during cutting, (c) physical modeling of the cutting interactions between the tool and the human tissue, (d) detecting collisions of virtual objects for real-time, (e) non-stable numerical resolution of the model dynamics, (f) obtaining stable haptic systems, etc.

Previous works address cutting by only taking into account the geometric and visual interactions between the tool and the virtual object. Only a few works model haptic cutting and a physical model of a cut is still missing. This paper presents an hybrid approach in which geometry, physical interaction and haptics are taken into account.

A. Related works

Most of the work done in cutting operations can be grouped in two main approaches:

A.1 *The geometry approach*

In the geometry approach we have that researchers choose between destroying the physical elements or dividing them. *Destroying elements* consists on removing the elements that collide with the virtual tool [1]. To reflect this change they update the inverse of the stiffness matrix K of their finite element model. However removing elements from the model violates the physical principle of mass conservation. Besides, for large meshes, updating K^{-1} can be computationally expensive. Cotin [2] and Piccinbono [3] improved performance computation and realism using a mass-tensor model and fine discretizations in cutting areas, however physical properties are still lost. *Dividing elements* proposes to create new tetrahedrons from a set of tetrahedrons that are in collision with the tool. Bielser et al. [4] use mass-springs mapped onto a tetrahedrical mesh. They subdivided any tetrahedron intersected by the virtual tool and store in a database a set of precut tetrahedrons. This method allows mass conservation and the physical properties remain unchanged. However, cuts add lots of small volume elements to the simulation loop. This might cause problems to reach real-time interactivity and numerical unstability problems may arise in the integration scheme. Mor et Kanade [5] simulate cutting using explicit finite element methods. They propose to cut a tetrahedron progresivelly and minimize the creation of new ones. They create some temporary elements while the tool is inside the tetrahedron and once the tool has left the tetrahedron, this one is destroyed and replaced by the new ones. However a big constraint is the disparity in the size of the new created elements since they causes big forces that might render the resolution method to become unstable. In any case, subdivision increases the simulated elements and real-time risk to be lost.

A.2 *Haptic rendering of cutting*

Few research has been done about haptic rendering of a cut. Mahvash and Hayward [6] based their research in an energy approach rather than in a stress approach. They distinguish three modes of interaction: deformation, fracture and cutting. During deformation, the work done by a tool is recoverable, during rupture, this work is zero and during cutting the work is equal to the irreversible work spent by fracture deformation. However, they efforts are only concentrated in the haptic interaction. No considerations are done in the dynamics of the cut and on the physical and geometrical model of the object. In fact, they obtained the haptic force from given linear, quadratic or cubic functions and not from a physical model. To our knowledge, Bielser and Gross [7] were the first to model and put together haptics cutting and a physical simulation. They model the forces of the scalpel for a haptic device. Nevertheless, they used a mass-spring model, physically less accurate than a finite element model.

B. Paper organization

This paper is organized as follows: first we describe the Green deformable model and its implementation using explicit finite elements. Next, we obtain a physical model of the cutting procedure. Section IV presents the geometrical algorithm. In section V we describe how we render the haptic sensation of a cut and how we merge the physical model, the geometrical algorithm and the haptics rendering. In section VI we present some of our results. Finally some conclusions and future research are done in section VII.

II. Physical Model

IN Gibson and Mirtich [8] much of the work done until 1997 in physical modeling is described. We don't use *Particle and mass modes* [9] since they are missing of a strong physical foundation even though they are easy to implement and fast to simulate. Besides these models depends on the connectivity of the springs which is clearly a handicap since cutting implies a change in the connectivity properties. *Boundary Element Methods* [10] are not suitable for cutting since only the boundary displacements and forces are computed neglecting internal geometry required for cutting. Recent *Long Element Methods* [11], which are based in Pascal's Principle and the conservation of laws, are not still well developed to support topology modifications. *Finite element methods* from elasticity theory are more physically accurate models. This methods [12] [13] obtain a huge sparse system of the type $u = K^{-1}f$ that is normally lengthy to solve for real-time interactions. Pre-inversion techniques allows real-time [1] but performing cutting might change the composition of stiffness matrix K invalidating the pre-inverted matrices. Another approach using elasticity theory are *explicit finite elements* [2][14] or enhanced with a multi-resolution scheme [15]. The way the animation is performed is quite similar to the mass-spring case, since each node of FEM dynamically integrates its motion from the positions of the neighboring nodes. These methods give more accurate results that mass-spring systems. In our approach we use an explicit formulation from a Green deformable model.

A. The Green Deformable model

This non linear model [16] is based in the Green strain model in which at each point of the material, local deformations are modeled using a *strain tensor*, ϵ. The Green strain is expressed by a 3 x 3 matrix. Its (i, j) coefficient is:

$$\epsilon_{ij} = (\frac{\partial \vec{x}}{\partial u_i}\frac{\partial \vec{x}}{\partial u_j} - \delta ij) \tag{1}$$

where the Kronecker delta is $\delta_{ij} = 1$ if $i = j$ or zero otherwise. The vector $\vec{u} \in \Re^3$ represents a location in the material coordinates and $\vec{x}(u)$ denotes the point's position in world coordinates as a function of its material coordinates. The *stress tensor*, σ, represents the force distribution inside the material. It can also be represented with a first order approximation as a 3 x 3 matrix. The force, f, acting on an elementary surface dA with an outward unit normal, $\vec{n}$, is $f = \sigma\vec{n}dS$. The stress tensor, σ has two components: the elastic stress due to the strain, $\sigma^{(\epsilon)}$, and the viscous stress due to the strain rate, $\sigma^{(v)}$. The total internal stress, is the sum of these two components: $\sigma = \sigma^{(\epsilon)} + \sigma^{(v)}$. If we assume that our material is isotropic and considering linear elasticity to link stress and strain, then we have:

$$\sigma_{ij}^{(\epsilon)} = \sum_{k=1}^{3} \lambda\epsilon_{kk}\delta_{ij} + 2\mu\epsilon_{ij}. \tag{2}$$

The material's rigidity is determined by the value of μ, and the resistance to changes in volume (dilation) is controlled by λ (incompressible materials should have an infinite λ). They are also known as Lame constants. Similarly, the strain rate, $\sigma^{(v)}$, is:

$$\sigma_{ij}^{(\nu)} = \sum_{k=1}^{3} \phi\nu_{kk}\delta_{ij} + 2\psi\nu_{ij}. \tag{3}$$

The parameters ϕ and ψ will control how quickly the material dissipates internal kinetic energy.

B. Explicit finite element models

Classical finite elements require to invert a global *stiffness* matrix:

$$u = K^{-1}f \tag{4}$$

This can be very lengthy for real-time interactions. Explicit finite element techniques [2] [14] also subdivides the object into subvolumes and obtain physical equations from them. But, instead of merging all these equations in a large matricial system, explicit finite elements solve the physics of each element independently. The procedure is as follows: first, we use the balance equation of each element to obtain the force at each node in function of the displacement of neighbor nodes. Then, instead of obtaining the equilibrium position by solving the matricial system 4, we only integrate the force at each node to obtain the new position for the node.

B.1 *Implementation*

We choose to use tetrahedral elements as they offer the best trade-off between speed and efficiency. A node i in the tetrahedron is described by its position in the material coordinates, $\vec{\mathbf{m}}$, its position in the world coordinates, $\vec{\mathbf{p}}$ and a velocity in world coordinates, $\vec{\mathbf{v}}$. Therefore, in the world coordinates the tetrahedron may be described by its nodes using $\vec{\mathbf{P}} = [\vec{\mathbf{p}}_{[1]}\ \vec{\mathbf{p}}_{[2]}\ \vec{\mathbf{p}}_{[3]}\ \vec{\mathbf{p}}_{[4]}]$ and $\vec{\mathbf{V}} = [\vec{\mathbf{v}}_{[1]}\ \vec{\mathbf{v}}_{[2]}\ \vec{\mathbf{v}}_{[3]}\ \vec{\mathbf{v}}_{[4]}]$. Following O'Brien et all. [14] we can compute the world position and velocity in terms of material coordinates using:

$$\vec{\mathbf{x}}(\vec{\mathbf{u}}) = \vec{\mathbf{P}}\vec{\beta}\begin{bmatrix} \vec{\mathbf{u}} \\ 1 \end{bmatrix}^{-1}; \quad \dot{\vec{\mathbf{x}}}(\tilde{\mathbf{u}}) = \tilde{\mathbf{V}}\tilde{\beta}\begin{bmatrix} \vec{\mathbf{u}} \\ 1 \end{bmatrix}^{-1} \tag{5}$$

where the inverse barycentric matrix, β, is defined by

$$\vec{\beta} = \begin{bmatrix} \vec{\mathbf{m}}_{[1]} & \vec{\mathbf{m}}_{[2]} & \vec{\mathbf{m}}_{[3]} & \vec{\mathbf{m}}_{[4]} \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1}. \tag{6}$$

To compute the values of ϵ and ν within the element requires the first partials of $\vec{x}$ with respect to $\vec{u}$:

$$\tfrac{\partial \vec{\mathbf{x}}}{(\vec{\mathbf{u}})} = \vec{\mathbf{P}}\beta\delta_i; \quad \tfrac{\partial \dot{\vec{\mathbf{x}}}}{(\vec{\mathbf{u}})} = \vec{\mathbf{V}}\beta\delta_i \tag{7}$$

where

$$\delta_i = [\delta_{i1}\ \ \delta_{i2}\ \ \delta_{i3}\ \ 0]^T \tag{8}$$

Conversely, the total internal force that an tetrahedrons exerts on a node is

$$\vec{\mathbf{f}}_{[i]}^{el} = -\frac{vol}{2}\sum_{j=1}^{4}\vec{\mathbf{p}}_{[j]}\sum_{k=1}^{3}\sum_{l=1}^{3}\beta_{jl}\beta_{ik}\sigma_{kl} \tag{9}$$

and the total internal force acting on the node is obtained by summing the forces exerted by all elements that are attached to the node.

C. The resolution method

We use a simple modified Euler integration scheme that is modified to an semi-implicit scheme due to the local nature of our explicit finite element method.

$$\vec{v}+ = \tfrac{\vec{f}}{m}dt \quad v* = (I_3 - \tfrac{dt^2}{m}H)^{-1} \quad x+ = vdt \tag{10}$$

where the Hessian matrix $H = \frac{\partial \vec{f}}{\partial \vec{x}}$ represents a first order approximation of the force applied on a node when it moves.

III. Physical Model of Cutting

We model a cut using a finite element deformable model and fracture mechanics. Since the interaction between the virtual tool and the object are focused in surgical applications, we consider that the cuts produce the least damage possible on the objects. This depends on the *sharpness* of the virtual cutting tool. We define **the sharpness**, κ, of a virtual cutting tool as the capability of a rigid object to produce a *cutting phenomenon* during an interaction with another object. We also define **a cut** as the process of *intentionally breaking* an object due to the application of external loads or forces created by a cutting tool (generally a rigid object). We consider the body to be initially free of external loads and having global elastic deformations, except for the small contact region between the tool and the object where the crack occurs and where a plastic behavior is considered. We distinguish three well defined states during the interaction between the cutting tool and the elastic object:

A. Deformation state

During this state the cutting tool does not cause a fracture on the surface of the object. The scalpel only produce deformations on the body that leads the local region in contact with the scalpel to be close to the plastic zone.

B. Cracking state

This is originated due to a the violent and sudden rupture of a material by external and internal loads. The location in which a fracture begins is known as a *crack*. To determine when a crack is produced we analyze the forces acting in the contact area between the object and the tool, considering this contact area as a single point (assuming that cutting tools have a contact area as minimum as possible). The force acting at this point is decomposed into its different components: perpendicular to the surface or tangent to the surface. They are used to determine a *rupture threshold*. Thus, to initialize a crack in a body we test if the tensile forces in the contact point are greater to a given *rupture threshold* considering the sharpness, κ, of the cutting tool and the damage, ζ, on the cut section. To start the analysis it is necessary to recall that in fracture mechanics there are two different type of failures:

- *Tensile failure:* This corresponds to loading *normal* to the failure surface. If the failure is produced by pushing rather than pulling then we can have a *compressive failure*.
- *Shear failure* This corresponds to loading *tangential* to the failure surface.

We analyze the forces that cause these failures. Let us consider the contact between the tool and the local region of the object as shown in figure 1. The internal forces equilibrate the external load produced by the blade of the cutting tool. The internal force distribution can be represented by an equivalent set of resultants, F and moments, M. Conversely, the contact surface dA depends on the shape of the tool. Actually, the sharpness of the blade, κ, that we have defined before will determine the size of dA. In the other hand, from classical mechanics we know that the traction, T, is defined as:

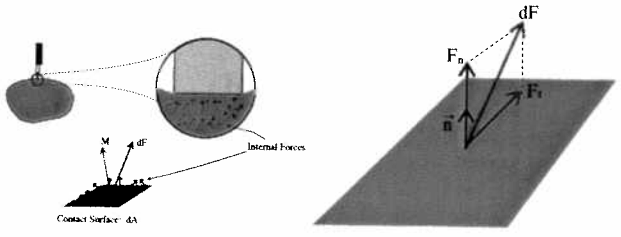

Fig. 1. Internal forces during contact between cutting tool and object

$$T = \lim_{dA \to 0} \frac{dF}{dA}. \tag{11}$$

It provides a measure of the direction and intensity of the loading at a given point. We can define our *cutting traction* by:

$$T_c = \frac{1}{\kappa} \lim_{dA \to 0} \frac{dF}{dA} \tag{12}$$

where κ is the sharpness of the blade.

Let us decompose the force into normal and tangential components, see right of figure 1. The normal component, F_n is simply the projection of dF along the normal $\vec{n}$. The projection is easily calculated using the dot product. Therefore, the scalar magnitude of the normal component is:

$$|F_n| = dF \cdot \vec{n} \tag{13}$$

and since the direction of F_n is given by the normal, then:

$$F_n = (dF \cdot \vec{n})\vec{n}. \tag{14}$$

Note that $dF = F_n + F_t$, then we have

$$F_t = dF - F_n \tag{15}$$

and its magnitude defined by $|F_t| = |dF - (dF \cdot \vec{n})\vec{n}|$. Substituting F_n and F_t into equation 12 and using the limit properties we can rewrite the *cutting traction* vector as:

$$Tc = \frac{1}{\kappa} \left[(\lim_{dA \to 0} \frac{|F_n|}{dA})\vec{n}_1 + (\lim_{dA \to 0} \frac{|F_t|}{dA})\vec{n}_2 \right] \tag{16}$$

where n_1 is the normal to the plane and n_2 is the tangent to that same plane,(i.e. a normal in another perpendicular plane). Then, to determine when a crack appears on the object it seems to be easy to obtain the normal and tangential forces and compare them with a force threshold. However, most of the measurable parameters available are given using the *fracture toughness*, K_I, of the material which is the critical stress intensity required to produce a failure in a given material. Therefore we are required to find an expression representing our *cutting traction vector*, T_c, in terms of stress. From our previous analysis, we identify the *normal* stress, σ, and the *shear stress*, τ :

$$\sigma = \lim_{dA \to 0} \tfrac{F_n}{dA}; \quad \tau = \lim_{dA \to 0} \tfrac{F_t}{dA}. \tag{17}$$

Then, our *cutting traction* vector may be expressed as:

$$T_c = \frac{1}{\kappa}(\sigma \vec{n}_1 + \tau \vec{n}_2). \tag{18}$$

To extend this analysis to the 3D case, we consider an infinitesimal cube located at point O and calculate the traction vector T in any plane with normal, $\vec{n}_i$, enclosing the infinitesimal cube of the contact area as shown in figure 2. Then our *cutting traction* vector may be expressed as:

$$\begin{bmatrix} t_{e_x} \\ t_{e_y} \\ t_{e_z} \end{bmatrix} = \frac{1}{\kappa} \begin{bmatrix} \sigma_{e_x x} & \tau_{e_x y} & \tau_{e_x z} \\ \tau_{e_y x} & \sigma_{e_y y} & \tau_{e_y z} \\ \tau_{e_z x} & \tau_{e_z y} & \sigma_{e_z z} \end{bmatrix} \begin{bmatrix} n_{e_x} & n_{e_y} & n_{e_z} \end{bmatrix}. \tag{19}$$

Or in a abbreviated expression,

$$T_c = \Gamma N \tag{20}$$

where T_c is the cutting traction vector, N is the set of normals and Γ is the set of normal and shearing stresses. The problem of fracturing is then solved by detecting the maximum stress that might

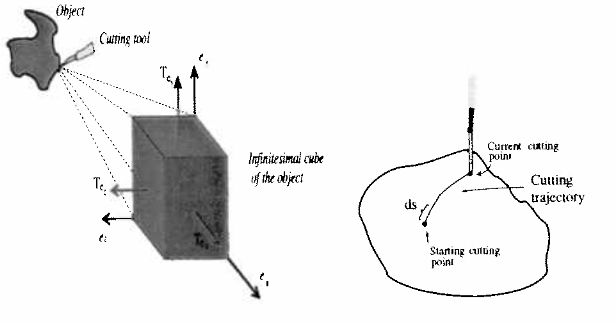

Fig. 2. (right) Infinitesimal cube taken from an object enclosing an arbitrary point (left) Cumulative cutting

over-pass the given threshold imposed by the material toughness, K_I. Classical mechanics states that the maximum normal stress, σ_{max} can be calculated using the maximum principal stresses. The principal stresses are actually the eigenvalues of Γ: σ_1,σ_2 and σ_3. The maximum shearing stress is computed from:

$$\tau_{max} = \frac{1}{2}max\{|\sigma_1 - \sigma_2|, |\sigma_1 - \sigma_3|, |\sigma_2 - \sigma_3|\}. \tag{21}$$

Furthermore, we introduce a coefficient $\zeta \in [0.1\ 1]$ to give a *damage* feature to the cutting area. For example a small value of ζ means that the section has been already cut and damaged and that it might be more easy to cut this region rather than a section that has not been cut, $\zeta = 1$. We define our *cutting stress*, σ_c, as:

$$\sigma_c = \frac{1}{\zeta}\min\left(\sigma_{max}, \tau_{max}\right). \tag{22}$$

Finally, we consider the trajectory of the cutting tool on the surface of the object. Assume small movements and neglect dynamic friction, then we may let ds be a displacement of the tool from the first collision point $s(0)$ to the current position of the tool, $s(t)$. The cumulative cutting stress is then given by:

$$K_c = \int_{s(0)}^{s(t)} \sigma_c(s)ds \tag{23}$$

DEFINITION 1: The **cutting crack** of an elastic material is defined as the fracture produced on the contact region, assumed as plastic, between the object and the cutting tool and that satisfies the following relation:

$$K_c \geq K_I \tag{24}$$

where σ_c is our cutting stress and K_I is the material toughness of the object.

C. Deformation after cut

Once a cut has been carry out, the continuous properties of the model change. The cohessive forces between the splitted section dissapear. The use of the explicit finite element let us set up this change by simply changing the barycentric coordinates of the object. Equation 9 allows to compute the force on a vertex depending on the tetrahedrons that share it. If a tetrahedron is separated then its influence in the given vertex is not taken into account any longer.

IV. The geometrical algorithm

Instead of destroying or subdividing the simulation elements we have decided to separate them. This approach has been treated in our previous work using mass-spring [17] and recently by Nienhuys [18]. Notice that if the elements are destroyed realism is lost and if the elements are subdivided then the number of elements increased making the simulation slower at each cutting operation.

A. The algorithm

We have divided the cutting procedure into different sub-tasks:

- Collision Detection
- Interpretation of the user's behavior
- Cutting test
- Selecting cutting surface
- Topology modification
- Singularity treatment

A.1 *Collision detection*

For fast collision detection computations we utilize a hierarchical bounding box tree that reduces the number of collisions checks and enable us to find the collision point in a fastest way. This procedure is further explained in [19]. Furthermore, since we consider surgery applications, in which the cutting tool is displaced in slowly movements by the surgeon, before using the hierarchical tree method we verify the last colliding facet and its neighbors.

A.2 *Interpretation of the user's behavior*

We determine if the user execute an attempt to cut or not. We consider slow and small movements as for minimal invasive surgery procedures. We proposes the following criteria based on the displacements of the virtual tool on the object surface. Let $C_p(t)$ be the colliding point at time t between the virtual tool and a facet. If the virtual tool touches the facet and remains in that position without moving, i.e. $C_p(t-1) = C_p(t) = C_p(t+1) = ...$, then this is only interpreted as a touching manipulation, see figure 3 (top left). Let $v_c(t)$ the closest vertex to the colliding point, $C_p(t)$, at time t. Then, if the user moves the virtual tool inside the same facet, meaning that $C_p(t) \neq C_p(t-1)$, we will consider

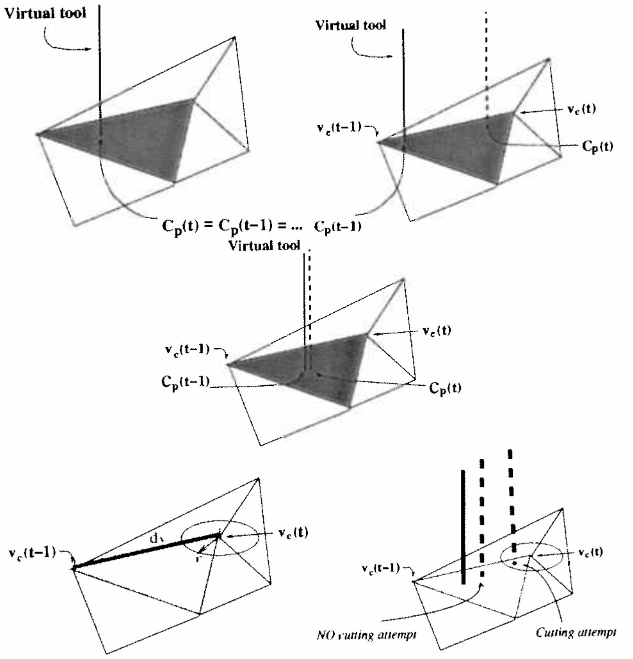

Fig. 3. Determining cutting attempts: based on the movement of the tool on the colliding facet and its neighbors.

a *cut attempt* if the closest vertex to the colliding point, $v_c(t)$, is different to that one in the previous step time, i.e.

$$v_c(t) \neq v_c(t-1), \tag{25}$$

see figure 3 (top right). However, this condition might be verified in some non realistic circumstances. For example, if the virtual tool is placed in the geometric center of the facet at time $t-1$, then by a very small movement of the virtual tool at time t, the condition $v_c(t) \neq v_c(t-1)$ can be easily easily verified, see figure 3 (middle) . We assume that small movements of the virtual tool with respect to the size of the facet are not *cut attempts* but only touching manipulations. For this reason, we constraint a bit more our condition by defining a neighborhood around the vertex $v_c(t)$ as shown in figure 3 (bottom left). Thus, to consider a *cutting attempt* the tool has to verify the condition of equation 25 and $Cp(t)$ has to lie in the neighborhood, i.e. the virtual tool has to be very close to the vertex $v_c(t)$. For simplicity we have chosen the neighborhood to be a circular region with radius r. Let d_v be the distance between $v_c(t)$ and $v_c(t-1)$ then we compute r as $r = \alpha d_v$ where $\alpha \in [0\ 1]$. The value of α indicates the size of the circular neighborhood and parameterizes the *cutting attempts*. For example, if $\alpha \approx 0$ the tool requires to travel on the facet for a longer period to consider the displacement as a cut, see figure 3 (bottom right). Analytically, consider d_c the distance between $C_p(t)$ and $v_c(t)$. Then, a *cutting attempt* takes place if $d_c < r$. Furthermore, if the user moves the virtual tool to a different facet, we will consider a *cut attempt* only if the closest vertex to the colliding point $v_c(t)$ is different to that in the previous step time, i.e. $v_c(t) \neq v_c(t-1)$, see figure 4.

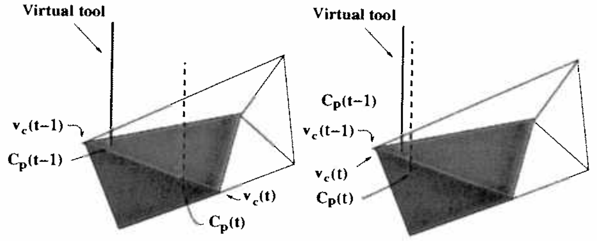

Fig. 4. (left) Cut attempt: The virtual tool has moved to a point in which the closest vertices at times t and $t-1$ are different. (right) A change of facet does not always means a cut. For example, if the closest vertex, v_c, is always the same.

A.3 *Cutting test*

Two main tests are considered: the physical (analyzed in the previous section) and the geometrical aspect. Since our deformable model lies into a tetrahedral finite element discretization, we take into account the possible changes in the tetrahedron to determine if a cut has to be carry out. In our approach, instead of [2] who takes into account the variation of length of the six edges of a tetrahedron ($l^k - l_o^k/l_o^k$, $k = 0, ..., 6$), we only compare the edges of the tetrahedron that are on the surface of the object. This consideration decreases the number of edges to compare and it is best for computer simulation purposes.

A.4 *Selecting cutting surface*

To select the tetrahedrons to be separated we identify two cutting procedures:

- **Conventional cutting procedure** This procedure focus in cutting virtual objects whose internal structure does not require an special care and the size of the cutting blade produces deep cuts. Most of previous works are based in this procedure.
- **Boundary cutting procedure** This procedure acts specially in the surface of the virtual object and the cuts are done slowly and progressively towards the internal part of the virtual body. This comes to be important when the internal structure of the object require special care like in human organs (e.g. liver cuts follow this approach to avoid cutting important vessels). The cutting contact is usually small and the cutting tool is unable to carry out deep cuts.

Since we focus our research in surgical procedures we selected the *boundary cutting approach* as the starting point of our work. Considering this and that our model lies in a tetrahedral mesh, we assume that the boundary cutting procedure takes into account only the tetrahedra that belong to the surface of the virtual body. Besides, the minimum number of vertices required to *open* the mesh is three, see figure 5. The line joining these set of vertices will be called the *cutting line* and it starts at $v_c(t-1)$ (closest vertex to colliding point at $t-1$) while the intermediate point will be $v_c(t)$. Thus in order to obtain the ending vertex, v_2 of the cutting line that best fits a realistic cut we project all the vertices that belongs to the neighbor facets of the first touched facet onto the planned spanned for this facet. We consider that the *cut attempt* is done in the direction, $\vec{s}_1$, from $C_p(t-1)$ to $C_p(t)$. We also consider that the cut is as straight as possible, then we define $\vec{s}_i$ as the vectors from the possible projected vertices to $C_p(t)$; v_2 will be chosen as the vertex v_i whose vector $\vec{s}_i$ is minimum with respect to $\vec{s}_1$ (see figure 6):

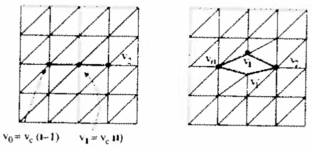

Fig. 5. In a *separation cutting approach* we need at least 3 vertices to represent a cut.

$$v_2 = v_i \quad \text{such that} \quad \min\{\angle(\vec{s}_1, \vec{s}_i)\}. \tag{26}$$

Let $\vec{s}_1$ be the vector from v_0 to v_1 and $\vec{n}$ the normal to the facet as shown in figure 7. Define P as the plane spanned by $\vec{n}$ and $\vec{s}_1$. Set $\mathbb{T}$ as the set of tetrahedrons, T, sharing the vertex v_1. Then, $\forall T \in \mathbb{T}$, that are in one side of the plane, are separated from those who belongs to the other side of the plane by only splitting

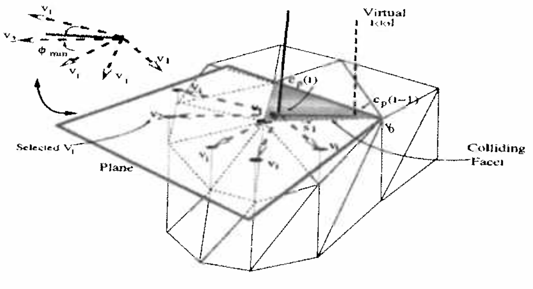

Fig. 6. Cutting line: ending vertex is obtained by the minimal angle between the projected vertices v_i onto the plane and the vector $\vec{s}_i$.

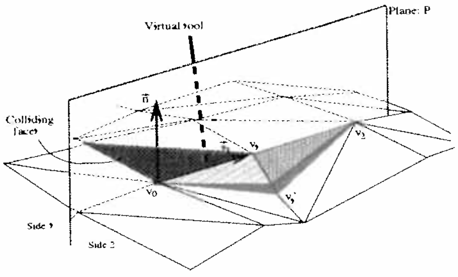

Fig. 7. A set of tetrahedra is put in one side of the plane (Side 1) and another set is put in the other (Side 2). The plane is spanned by vectors $\vec{s}_1$ and $\vec{n}$.

v_1. When a tetrahedron is divided by the plane, P, the tetrahedron will lie in the same side of the plane of its furthest vertex.

A.5 *Topology modification*

To reflect the cut we reposition the vertices of the cutting line by translating v_0 and v_2 to $C_p(t-1)$ and $C_p(t)$ respectively. The vertex v_1 is orthogonally projected to the plane spanned by the virtual tool as shown in figure 8. After reshaping the tetrahedrons, we need to keep the validity of the explicit finite element by updating the β matrix and the volume of the involved tetrahedrons. Degeneracy of tetrahedrons during the cutting procedure is not treated.

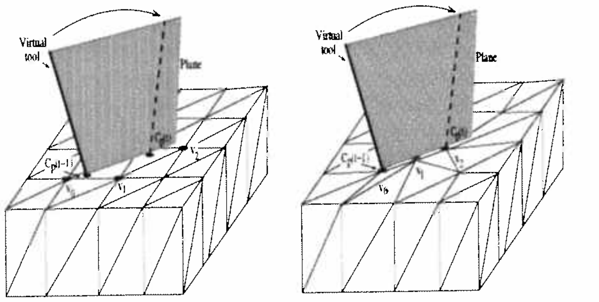

Fig. 8. (left) The virtual tool spans a plane depending on its current and previous position and orientation. (right) The nodes of the tetrahedrons are projected on the plane.

To split apart the two sets of tetrahedrons, a new vertex is created as shown in figure 7. The tetrahedrons placed in *side 1* of the plane keep using the vertex v_1 and the tetrahedrons that are in *side 2* will use the new created vertex v_1'.

A.6 *Singularity detection and treatment*

It is possible that some connectivity properties are lost while cutting and leading to a non consistent mesh.

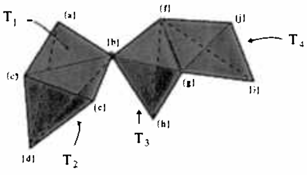

Fig. 9. Vertex $\{b\}$ is singular, edge $\{f, b\}$ is singular.

DEFINITION 2: **Singularity:** A volumetric mesh used for physical simulation is said to be singular when the simulation elements (e.g. tetrahedrons) present geometrical connections without area, see figure 9.

The topological data structure let us identify fast and efficiently these singularities. More precisely, our data structure is organized as an *abstract simplicial complex* K consisting of a set of vertices $1,\ldots,m$ together with a set of non-empty subset of the vertices, called the *simplexes* of K, such that any set consisting of exactly one vertex is a simplex in K, and every non-empty subset of a simplex in K is also a simplex in K. A simplex containing exactly $d+1$ vertices has dimension d and is called a *d-simplex*. For example, the facet with vertices $\{v_1, v_2, v_3\}$ is 2-dimensional, i.e. $d=2$ and it is called a 2-simplex. The *faces* (do not confuse with a geometric facet) of a simplex, s, denoted $\bar{s}$, are the edges and vertices and they are themselves a set of non-empty subsets of s. The *star* of s, denoted $star(s)$, is the set of simplices of which s is a *face* (in the topological sense).

ASSUMPTION 1: Let s^0 be a 0-simplex (vertex) and s^3_n the n 3-simplexes of which s^0 is a *face*. If the simplexes resulting from $star(s^3_1 \cap s^3_2 \cap s^3_3 \ldots s^3_n)$ have dimension $d \leq 3$ then s^0 is *singular*, in the other hand if $d=3$ for all the resulting simplexes then s^0 is not singular.

Instead of formally proving the last assumption we show it by an example. Let the tetrahedrons $T_1 = a,b,c,d$, $T_2 = b,c,d,e$, $T_3 = b,f,g,h$, $T_4 = f,g,i,j$ be 3-simplexes of K. To analyze if the 0-simplex $\{c\}$ in the figure 9 is singular or not we need only to analyze the *star* operation of the intersection of the 3-simplexes of which $\{c\}$ is a face, i.e. T_1 and T_2:

$$star(T_1 \cap T_2) = star(\{b,c,d\}) = \{\{a,b,c,d\},\{b,c,d,e\}\} \tag{27}$$

We see that the result is a set of simplexes of dimension $d=3$ and we conclude that the vertex c is not singular. In the other hand if we analyze the 0-simplex $\{b\}$, we notice that is singular. To see that we take the *star* operation of the 3-simplexes of which $\{b\}$ is a face, i.e. T_1,T_2,T_3: Since one of its subsets is $\{a,b\}$ then we find that $d \leq 3$ and thus $\{b\}$ is singular.

V. Rendering cutting using haptics

In our previous work [20], we have solved the difference rate frequency problem between the physical (about 25 Hz) and the haptic simulations (about 1 Khz) by separating the haptic and the simulation loop and linking them by a *buffer model*. Thus, instead of interacting with the whole physical model we interact with only a local or intermediate representation of it. In the other hand, to render haptic cuts we assume that we cannot render a cut sensation to the user if the visual sense has not detected the cut. We distinguish three modes of interaction: haptic forces obtaining before, after and during cutting. As shown in figure 10 we consider two deformation curves. The upper deformation curve is obtained before a cutting happens and as Mahvash and Hayward [6] we consider that the work done by the tool is recoverable. However when the physical simulation detects a cut, a signal F_c is send to the haptic loop to start the cutting mode. The cutting mode leads to another deformation curve which represents the behavior of the force after a cut considering the work done during the cut as unrecoverable. The deformation curves are obtained from the

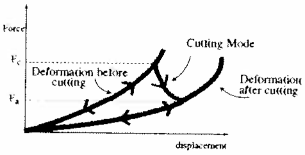

Fig. 10. Behavior of a cut

physical model. In the *cutting mode* the force is computed using a non-linear criteria:

$$F_{haptic}(I) = F_c e^{-\xi I} \qquad (28)$$

where F_{haptic} is the force feedback provided by the haptic device, F_c is the cutting signal sent by the physical model, I represents the number of iterations executed by the haptic loop *after* the cut and the value of ξ represents an elastic or plastic cut ($\xi >> 1$ for plastic cut for a quasi-plastic haptic sensation of the cut). This is valid only for a limited number of haptic iterations, in our case it takes only the time that a cut is simulated by our physical and graphical simulation. The next schema shows better how we have merged the physics of the cut to the geometrical cut and to the haptic rendering of the cut.

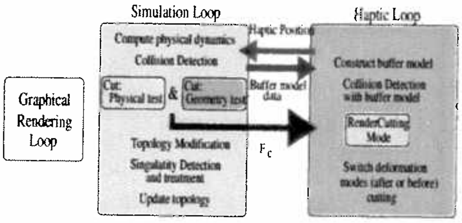

Fig. 11. The cutting schema

VI. Results

In figure 12 we show a physical simulation of a ligament tissue. It is composed of 100 tetrahedrons simulated using explicit finite elements and non-linear Green formalism. A PHANToM device is coupled to the simulation to render the sensation of touching and cutting the object. We use a 800 MHz. processor and we obtain 30 Hz. for the physical and graphical simulation. The haptic rendering reaches the 1000 KHz. However, high deformable objects presented self-collisions after being cut.

VII. Conclusion

We have proposed a physical and geometric model of cutting human tissue based in a stress approach and it has been integrated to a haptic rendering algorithm of a cut. This hybrid approach puts together approaches where either only the geometric part or either only the haptic part were taken into account.

The implementation of our approach is a proof-of-concept of this technique. This work opens new roads of research: object self-collisions caused during a cut need to be treated, degeneracy detection and treatment are also part of our current research.

VIII. Acknowledgements

This work has been supported by the National Council of Science and Technology of Mexico (CONACYT) and by the National Research Institute in Computer Science and Control (INRIA), France.

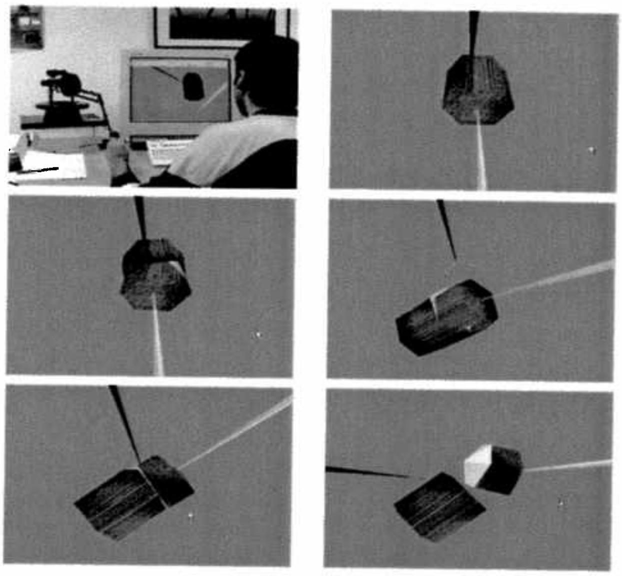

Fig. 12. Cutting a human ligament using a force feedback device

References

[1] M. Bro-Nielsen and S. Cotin, "Real-time volumetric deformable models for surgery simulation using finite elements and condensation," in *Proc. Eurographics'96*, 1996.

[2] S. Cotin, *Modèles anatomiques déformables en temps-réel.*, Thèse de doctorat, INRIA Sophia Antipolis – Université de Nice, Sophia Antipolis, 1997.

[3] G. Picinbono, "Modeles geometriques et physiques pour la simulation d'interventions chirurgicales," in *PhD Thesis, INRIA Sophia-Antipolis, FRANCE*, 2001.

[4] D. Bielser, V. A. Maiwald, and M. H. Gross, "Interactive cuts through 3-dimentional soft tissue," in *EUROGRAPHICS'99*, 1999, vol. 18, pp. C–31–C38.

[5] A. B. Mor and T. Kanade, "Modifying soft tissue models : Progressive cutting with minimal new element creation.," in *MICCAI, Medical Image Computing and Computer Assisted Intervention*, Pittsburg, U.S.A., Oct. 2000, vol. 1.

[6] M. Mahvash and V. Hayward, "Haptic rendering of cutting: A fracture mechanics approach," *Haptics-e, http://www.haptics-e.org*, vol. 2, no. 3, Nov. 2001.

[7] D. Bielser and M. Gross, "Interactive simulation of surgical cuts," *Proceedings of Pacific Graphics, IEEE Computer Society Press*, pp. 116–125, 2000.

[8] F. Gibson and B. Mirtich, "A survey of deformable models in computer graphics," Tech. Rep., MERL, Cambridge, MA, http://www.merl.com/reports/TR97-19/index.html, 1997.

[9] S. Gibson, "3d chainmail: A fast algorithm for deforming volumetric objects," in *Proc. Symp. on Interactive 3D Graphics, ACM SIGGRAPH*, 1997, pp. 149–154.

[10] D. James and K. Pai, "A unified treatment of elastostatic contact simulation for real time haptics," in *Haptics-e Journal, http://www.haptics-e.org*, 2001.

[11] K. Sundaraj, C. Laugier, and Costa I. F., "An approach to lem modelling : Construction, collision detection and dynamic simulation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2001.

[12] D. Terzopoulos and Fleischer K., "Modeling inelasctic deformation: Viscoelasticity, plasticity, fracture," in *Computer Graphics, Proceedings of SIGGRAPH'88, Atlanta, Georgia*, 1988.

[13] D. Gourret, N. Magnenat-Thalmann, and D. Thalmann, "Simulation of object and human skin deformation in a grasping task," *SIGGRAPH*, pp. 21–30, July 1989.

[14] James O'Brien and Jessica Hodgins, "Graphical models and animation of brittle fracture," in *SIGGRAPH Conference Proc.*, 1999.

[15] G. Debunne, M. Desbrun, M. Cani, and A. Barr, "Dynamic real-time deformations using space and time adaptive sampling," in *Computer Graphics, SIGGRAPH, USA*, 2001.

[16] Y.C. Fung, *A First Course in Continuum Mechanics.*, Prentice-Hall, Englewood Cliffs, N.J., 1969.

[17] F. Boux de Casson and C. Laugier, "Simulating 2d tearing phenomena for interactive medical surgery simulators," in *Computer Animation 2000, Pensilvania, USA*, May 2000.

[18] H. Nienhuys and F. Vanderstappen, "Combining finite element deformation with cutting for surgery simulations," in *EUROGRAPHICS 2000*, 2000.

[19] K. Sundaraj and C. Laugier, "Fast contact localisation of moving deformable polyhedras," in *Proc. of the Int. Conf. on Control, Automation, Robotics and Vision,Singapore*, Dec. 2000.

[20] C. Mendoza and C. Laugier, "Realistic haptic rendering for highly deformable virtual objects," in *IEEE Virtual Reality (VR), Yokahama JAPAN*, mar 2001.