Deformable Body Simulation with adaptive subdivision and cuttings

P. Meseure, C. Chaillou
{meseure,chaillou}@lifl.fr
http://www.lifl.fr/GRAPHIX
LIFL - Université de Lille 1 -Bât M3
59655 Villeneuve d'Ascq CEDEX FRANCE

Abstract: *We propose a deformable body model in order to simulate the dynamic behaviour of human organs during surgical operations. It is aimed at simulating organs in training surgical simulators, currently developed in our team. However, since this model is general, it can be used in simulation or animation of other deformable objects. In order to be sufficiently realistic, the model must behave dynamically, that is obey to the physical laws of motion. Moreover, it must be possible to interact with it. This requires impacts and contacts to be detected, and an accurate processing of extreme interactions such as cuttings and tears to be implemented. The model we present is intended to respond in an optimal manner to these constraints in realtime. It is based on a spring surface mesh, fitted with a virtual rigid component which does not interact with the environment and provides the structure with a rigid behaviour. In order to get both contacts precision and realtime, an adaptive subdivision of the surface mesh is used: a small number of nodes are defined at the beginning of the simulation, and more nodes are added dynamically to the mesh in the contact zones.*

1. Introduction

We plan to design a number of training surgical simulators, in areas such as ophthalmology [MKCD95], ultrasound endoscopy [VC95] or gynaecologic laparoscopy. Our goal is simple: we want to build the tools to enable the surgeons to learn how to operate not on actual patients but on virtual ones. Nevertheless, in the medical field, a simulator requires not only to generate highly realistic images but also to reproduce the mechanical interactions between the surgeons and the operated organs. The complete system consists therefore in all the means to measure the motion of the tools manipulated by the surgeons, to compute the behaviour of the manipulated organs and to reproduce their modifications both in a visual and tactile manner.

Realtime simulation of the dynamic behaviour of deformable or rigid bodies is a new challenge in computer graphics today. The simulated organs can be quite different, so that the model we plan to design must be general. This generality makes it usable in other areas requiring deformable body animation, providing that the same preliminary hypotheses are chosen.

To build such a model, three main problems are to be solved:

First, the motion of the objects has to look realistic. This means that the model must be based on the physical laws governing the cinematic and dynamic behaviour of the rigid or deformable bodies. Those equations usually result in second order differential equations, which require a correct integration method. Kinematics is essential for the reproduction of motion, while dynamics is necessary to compute the forces returned to the user. These forces are also called *force feedback*.

Second, the object must interact with its environment. For instance, the surgeon, by means of a tool, can manipulate the organs, come into contact with them, move (push/pull) them and/or deform them. Hence, these interations can produce both local or global deformations and rigid motions (translations, rotations).

Finally, the interactions can be drastic and result in structure modifications: cuttings and tears for example.

In this paper, we first analyse the previous works in the field of organ modelling. Then, the purposes and constraints of the design of our model are exposed, as well as the necessary preliminary hypotheses to simplify the problem. Our model is then presented. This model includes a refinement process, which enables precise contacts handling without compromising realtime simulation.

Proceedings of the *WSCG'97 Conference*, Plzen, 10-14 february 1997, pp 361-370.

2. Previous Works

Until now, only few models for realtime dynamic simulation of organs have been proposed. Some of them are based on motion and deformation models, but not always on physical laws. Ad-hoc deformations in contact zone can indeed be used and simple models applied to compute the force feedback. For instance, [HTB95] interpolates actual ultrasound images to produce deformations. [SBM94] propose an outstanding model of an eye, with many visual details. However, only the cornea is mechanically defined. A finite element model for incompressible deformations in case of large displacements is used, but, unfortunately, the simulation process is not detailed. In the same way, [SPHL95] have carried out an eye surgery simulator including force feedback. The model is not, strictly speaking, based on physics: measurements of the physical properties of the eye have been taken, and these values are applied to the force feedback system. Though this process does not guarantee the right behaviour, the resulting forces stay in a correct range of magnitude. Finally, another model, based on the Finite Element Method (FEM) and proposed by [CDBA96], solves the realtime problem by computing in a first step the deformations generated by the displacement of each node. By applying the superposition theorem, the deformations are then summed up during the simulation. The realism of liver simulation they obtain is surprisingly good. Beside this research, condensation techniques followed by the inversion of the stiffness matrix are proposed by [BC96] to reduce computation time.

However, these two finite element methods are limited, because they assume that the organ is elastic and the displacements are small. Moreover, any modifications in the structure induce a new computing of the matrices of the system. It is therefore hard to make the first solution compatible with cuttings. The FEM is so much time-consuming that it usually requires simplifications and drastic hypotheses to make it realtime. In the simplest case (that is to say, using linear elements, linear elasticity theory and small displacements), the FEM is based on the resolution of a linear system, whose complexity is quadratic with respect to the number of nodes.

All the methods described before, ad-hoc deformations or FEM, are not dynamic. That means that the equation governing their motion depends only upon the positions of the nodes, neither on their speed nor their acceleration. Even the spring model of [CEO93] tries to guess by heuristics where the equilibrium positions of the nodes are located, and therefore, does not simulate the return to this position. In static environment, each computation step is considered as an equilibrium state. This makes the computation of the transitional states impossible without the addition of an extra model to process the return to equilibrium state. The usefulness of dynamic environments for the modelling of organs in surgical simulation is still an open question. [BC96] recommend to use static equations as soon as the number of nodes become too high. For our concern, we do think that static equations are too limited for the simulation of organs. If a body with a high damping, or on the contrary a low damping, has to be simulated, transitional states are necessary to reproduce, in the first case, the slowness of the return to the equilibrium, and in the second case, the vibrations of the structure before stabilization. Static models are suitable only when the damping is close to the state between slow returns and vibrations.

Static model are often preferred, on one hand, because of equation simplifications and faster computation, on the other hand, to avoid instability in the equation integration phase. Two kinds of integration methods exist: explicit and implicit methods [Gou94]. Explicit methods are simpler and therefore more interesting for realtime applications. Their complexity are $o(n)$, with n the number of nodes. However, these methods are conditionally stable. Shannon theorem requires that the system should be sampled at, at least, the double of the highest frequency of the system. If the sample rate is not sufficient, the model can diverge: it can then exhibit vibrations of increasing amplitude which disturb the geometry of the object.

On the contrary, implicit methods can be used to ensure convergence even if the Shannon condition is not enforced. However, these methods usually result in a non linear equation system. Moreover, even if these equations are approximated by linear equations, the complexity remains the same as a linear system, that is $o(n^2)$, for n nodes).

Spring models or the more general CORDIS/ANIMA [LJFC91] model seem more attractive for our purpose. For a completely connected graph (each mass is connected to all the other masses), a spring model described by dynamics equations and integrated by explicit methods, has a quadratic complexity: for each of the *n* masses, the force exerted on the mass is obtained in *o(n)*, and then is integrated. Moreover, it is possible to reduce this complexity by decreasing the number of links. To sum up, the maximum complexity of a (completely connected) spring model is the same as the minimum complexity of a finite element model (linear elasticity, linear elements), and is still compatible with high displacements and transitional states.

Another problem, as crucial as deformation modelling, relates to collisions and more specifically contacts. Most previously described simulations only detect which nodes of the finite element or spring mesh come into contact. Hence, all contacts are approximated to the nearest node in the mesh. Therefore, the collision process directly depends upon the sampling of the mesh. Spacial sampling must be increased in order to get a precise contact and avoid visual aberration. More over, some deformation models make the collision detection very time consuming. For example, [KKKH95] propose to model organs with a spring model. The masses are the control point of nurbs surface which define the shape of the organ. The complexity of the organ geometry makes it difficult to compute contact in realtime.

Finally, most deformation models only take into account displacements compared to an unmovable rest configuration. However, in a number of cases, the objects can be moved, translated and/or rotated. During a surgical operation, it is not seldom when an organ has to be pushed because it prevents the surgeon from accessing the operated zone. In the same idea, gravity has to be simulated. It is indeed in charge of any motions, except those due to the operator. These *spontaneous* motions disturb the surgeon during the operation. This means that the behaviour of each deformable object has a rigid component which have to be taken into account.

To sum up, our aim is to find a model giving correct displacements and deformations, reproducing transitional states, and able to handle different possible contacts, without any hypothesis about the sampling of the surface. This doesn't mean that we want a perfect physical model of the simulated object, but a simple model able to reproduce the plausible visual and tactile aspect of the object. For realtime reasons, the total amount of computation should stay reasonable and constant time algorithms should be preferred. In other words, we want this object to behave naturally, like in the real world. This model is intended to simulate deformable objects in large environment, as it is the case in endoscopic operations.

3. Preliminary hypotheses

3.1. Synchronicity

First of all, the model is intended to run in realtime, on usual workstations, even on up-market PC. The challenge is to find the best compromise between realtime, realism and memory. In the dynamic model simulation field, the realtime concept is relatively imprecise, and means most of the time a degree of interactivity. If the model is dynamic, *synchronicity* must be ensured [ULC94]. The main idea is that the mechanical simulation is computed and the result displayed, in, at most, the time interval the actual action takes to complete. Synchronicity and realtime concepts are independent, as seen in the following example. Let's assume that the simulation time step is *1ms* but the calculation time is *10ms*. If a point moves with a speed of *1m/s*, it will move on a distance of *1mm* during a simulation time step. However, the user will have the impression that the point has the same motion but during *10ms*, and seems to move with a speed of *0.1m/s*, so is 10 times slower. In the same way, any user's motion will be 10 times faster in the computation frame. Thus, a contact will be considered as a tough impact.

To choose the right sampling rate, the free hand gesture frequency, *300Hz*, has to be taken into account. The drawing should have a frame rate of, at least, 10 images/s for minimum visual interactivity. Realism is related both to display (light, textures) and mechanical simulation (deformations, force feedback).

3.2. Interaction Types

In this paper, we assume that the interaction is performed between a rigid tool and a deformable organ which is supposed to be *elastic*. The global simulation scheme, as described on the *Figure 1.* is a simplification of [LJFC91]: sensors measure the tools motion, send the tools position to the organ model which will deform accordingly. Two kinds of nodes appear on the deformable body. The nodes which are in the contact zone, whose position is given by the tool and on which the exerted forces are unknown. These forces will be computed by evaluating the deformation internal strains and will be sent to the force feedback system. The position of the other nodes is unknown, but will be determined by summing all the forces acting on them and integrating. This partition of nodes is well known and has already been described in [GTT89]. This model appears as a module with position values as input and both deformations and forces as output, to produce respectively the image and the force feedback.

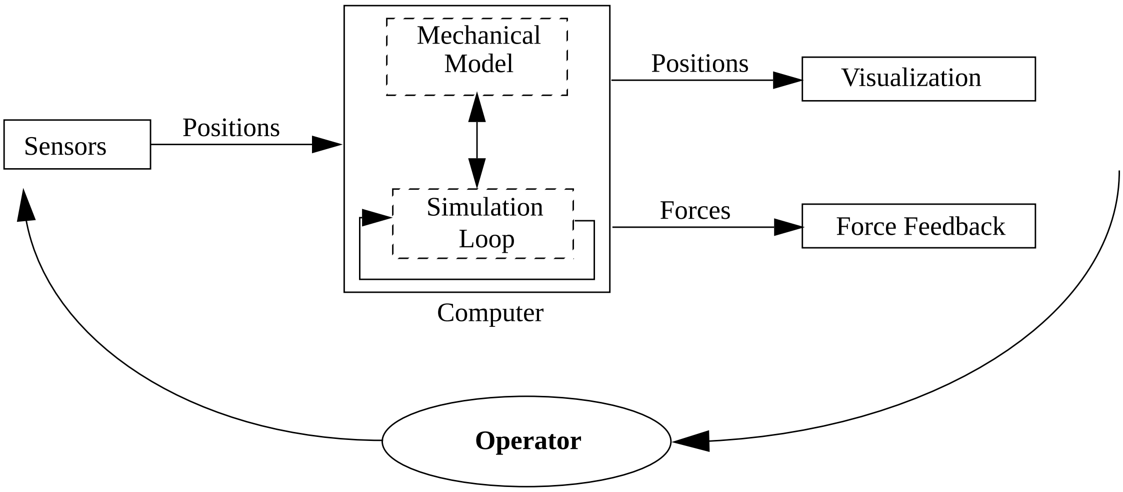


Figure 1. Global Simulation Scheme

4. Mechanical Model

4.1. A spring model with a rigid component

As seen in the previous analysis, it seems that spring and damper models are of great interest for our simulation, since they offer a good compromise between realtime, realism and memory. In a medical context, all gesture are moderate and calm, not tough, and consequently, of low frequency. Unfortunately, the expression of the strain exerted by a spring with no constant axis is not linear and requires a square root. A model with too many springs is no more interesting. Thus, the number of springs has to be kept small. Only a surface mesh could be considered, where only the neighbouring nodes are joined together by damped springs, since we are only interested in the position of the surface node for the drawing. Such a model is very simple since the number of springs is then a linear function of the number of nodes (if we assume that a node has, at most, a given number of neighbours). However, it lacks a volume coherency: if the model is put onto a plane into a gravitational field, it collapses. Thus, a simple surface mesh is not sufficient: the number of springs has to increase, to add links between second (or more) neighbours[1], and/or more complex types of links such as hinges must be used.

A practical way to handle deformable models submitted to rigid motions is to consider that the body is composed of a rigid kernel which is responsible for translations and rotations. It is surrounded by a deformable component. This process has already been proposed previously. In [Gas89], springs with rest distant *L* and constant axis link the rigid kernel and the surface deformable mesh together. This model is unfortunately not adapted to our purpose, because it is not compatible with cuttings. The deformations are indeed computed by a blending between volume and surface preservation. The mesh cannot be altered, because it guarantees the deformation coherency. Furthermore, this rigid component inside the deformable object does not correspond to the mechanical properties of the organs we plan to model.

1.We call “second neighbours” of the point, the neighbours of the neighbours of the point.

We have chosen a model fitted with a rigid *virtual* component, only present for dynamics computation. This rigid component is linked to a damped spring surface mesh (which forms the visible part of the object, that is to say will be used to display the object), by means of zero-length rest springs. Each mesh node is joined to the rigid component with one spring. Thus the number of springs is always a linear function of the number of nodes. Without any interaction, rigid and deformable components stay in the same place.

By saying that this rigid component is *virtual*, it is meant that it cannot come into contact with the other objects in the environment. If it could interact with the other objects, then the deformable object would have almost a rigid behaviour: it would stay at rest in a undeformable manner, and would not deform due to its own weight, if for example, it was put on a rigid plane. The rigid component is only submitted to volumetric forces such as gravity. In case of contact, the rigid and deformable components will separate each from the other. On the contrary, when the interaction stops, the two components get joined again. *Figure 2.* shows the state of the system at rest on a plane. The rigid component does not interact with the plane, so continue to fall. But, since the plane prevents the surface component from falling, the rigid component is held by the surface component, for zero-length springs connect the two components together. Therefore, the surface component deforms because of its own mass and the mass of its volumetric part, the rigid component. However, the rigid component garantees that the surface component does not collapse.

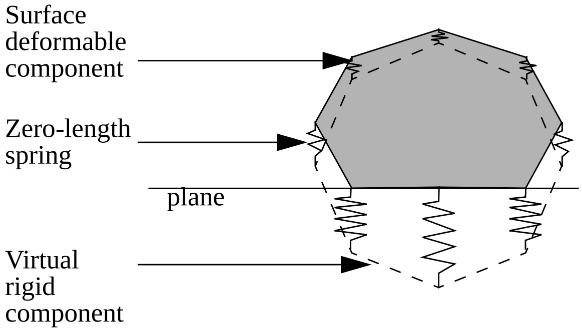


Figure 2. Deformable and rigid components separation in case of contact

Our model is very close to [TW88]. This last splits the equations of motion into three parts: the two first describe the translations and rotations of the overall body, and the third one computes the deformation of the body referred to a undeformable rest state, called *reference component*. This splitting is a result of a mathematical mechanism: the position of any point of the body is expressed referred to the reference component. On the contrary, our model is based on an intuitive idea: the rigid component shows the place where the body should be if there was not any interaction. The behaviour of the two models is then completly different. In a situation as described on *Figure 2.*, the rigid and deformable components would have their centre of mass in the same place, the rigid component would not be sag so much, but would stay around the deformable component.

4.2. Integration of the equations

The Euler method is chosen. It consists in doing a Taylor development in the first order for speed and position. For each node in the mesh, we compute all the external forces $\vec{F}_{ext}$, such as gravity and air viscosity, contact or impact forces $\vec{F}_c$, and elastic (internal) forces $\vec{F}_{int}$. The acceleration $\vec{a}_i$ is then computed and integrated to obtain the speed $\vec{v}_i$ and the position $\vec{x}_i$ of the i^{th} particle:

$$\begin{aligned} \vec{a}_i(t) &= \frac{\sum (\vec{F}_{int}(t) + \vec{F}_{ext}(t) + \vec{F}_c(t))}{m_i} \\ \vec{v}_i(t) &= \vec{v}_i(t-\Delta t) + \Delta t \times \vec{a}_i(t) \\ \vec{x}_i(t) &= \vec{x}_i(t-\Delta t) + \Delta t \times \vec{v}_i(t) \end{aligned} \quad (1)$$

The simulation time step $\Delta t = 1ms$ has been chosen. To determine the rigid component behaviour, the sum of the forces acting onto it is computed. These forces are the external forces (gravity, air viscosity) and the strain of the springs linking it to the deformable component. Acceleration, speed and position of the centre of mass, are obtained by the same way as seen before. The orientation is obtained by computing the torque $\vec{T}$, which allows the program to calculate the instantaneous rotation speed $\vec{\omega}$ by integration, and get both an axis $\vec{u}$ and an angle θ for the instantaneous rotation. This rotation allows the program to update a quaternion representing the absolute orientation of the solid:

$$\begin{aligned} \frac{d}{dt}(I\vec{\omega})(t) &= \sum (\vec{T}_{int}(t) + \vec{T}_{ext}(t)) \\ \vec{\omega}(t) &= \vec{\omega}(t-\Delta t) + \Delta t \times \frac{d}{dt}\vec{\omega}(t) \\ \theta(t) &= \theta(t-\Delta t) + \Delta t \times \omega(t) \\ \vec{u} &= \frac{\vec{\omega}}{\omega} \end{aligned} \qquad (2)$$

4.3. Analysis of the model

The model has been implemented and tested. The implementation has shown some important advantages:

• The number of springs is limited and is a linear function of the number of nodes in the mesh.

• The rigid component is responsible for the rotations and translations of the object, hence a correct rigid behaviour of the object is obtained.

• Surface springs allow the modeller to control the surface deformations of the object: it is hence very easy to define the strength to cuttings for example.

• The springs which link the rigid and deformable components together control the resistance to pressure. It is possible, by choosing the appropriate coefficient of elasticity, to simulate very deformable objects which are elastic in only one or two dimensions and with no volume coherency such as plastic bags or rubbers, or on the contrary, almost rigid objects.

• The rigid component makes the mesh have only one rest position, whatever its topology is. The mesh can naturally be more complex, with the enclosing of hinges for instance. In the same idea, if the topology of the mesh is altered during the simulation, due to a cutting for example, the rigid component will allow the mesh to keep volume coherency. In other words, the rigid component will guarantee the robustness of the mesh.

• By modifying the rigid component, it is possible to simulate plasticity phenomena [TF88].

• The mesh generates polygons which can be used for drawing the structure with a graphics accelerator. These polygons will also define the object shape for the collision detection.

• The masses of the deformable and the rigid component, as well as its inertia matrix, are independent. This is also true for the air viscosity coefficients.

Unfortunately, some critical points still remain, due to limits of the model:

• Synchronicity is crucial. As it has already been exposed, if synchronicity is not obtained, the speed of the user motions will be multiplied and the effect on the model will make it unstable.

• In order to generate a simple model, only local deformations are taken into account. The displacement of each node is computed independently of the displacement of the other nodes. All global behaviour is controlled by the rigid component, so no global transformations can be performed. It is therefore hard for the model to bend, for instance. A bending can however be performed by means of a set of appropriate local deformations or by bending the rigid component.

• The control of the deformations is limited. For instance, it is hard to find forces which can guarantee that the volume or the surface are preserved. It is however possible to add bounds to limit the deformation.

• Deformation are always local to the place where the external force is exerted. This can result in a too high local deformation, and a small perturbation of the springs which are farther. If the force gets more intense, all the structure will move, instead of making the deformation zone bigger.

• The integration method is not always stable. Generally, the damping of the springs or the viscosity of the ambient environment must be increased to reach a critical value. But then, if the damping is too high, the model seems very slow, always slackened, just as it was plunged into a viscous fluid (and indeed it is, since then the viscosity is no more the viscosity of the air).

• Two main reasons make it essential to keep the number of nodes in the mesh low. First, to keep the simulation realtime, the number of springs is reduced, because the computation of the strain exerted by a spring requires to calculate a square root. To reduce the number of springs, the number of nodes has to decrease too. Second, convergence conditions of the integration scheme are closely related to the mass of each node: the natural frequency is indeed a function of $\sqrt{\frac{k}{m}}$. To keep this frequency low, the mass of the nodes must stay high enough. Since the total mass of the surface mesh is constant, the number of nodes must not be too large or the mass of each node will become too small. To sum up, if we have to increase the number of nodes, the natural frequency of the whole system will increase and therefore the sampling rate. Above a limit, it won't be possible any more to ensure synchronicity on the simulation machine.

A nice way to partially solve the two last limits, that is to say too high local deformations and divergence, is to bound the distances between nodes as proposed by [Pro95]. However, this method doesn't guarantee convergence and can lead to self-supported vibrations.

Though these drawbacks must be addressed, the proposed model is currently sufficient for simple simulation with few interactions.

5. Adaptative subdivision

We have seen in the previous section that the number of masses and springs of the mesh had to be reduced for realtime and integration time step reasons. The sampling of the external surface is low, so the contact process is coarse. Following the example of deformation method for clothes [HPH96], the mesh is subdivided to get it more precise in the contact zone.

A complete presentation of our collisions treatment process goes beyond the scope of this article. Indeed, it corresponds to impact detection, detection acceleration and collision response computation (contact or impact). Briefly, we can say that detection is computed between objects trajectories and not by determining 3D intersection between objects at each simulation time step. As described by [Boy79], a collision occurs between two objects *A* and *B*, during time 0 and Δt, if at least one of these conditions is verified:

- *a vertex of A passes through a face of B*
- *a vertex of B passes through a face of A*
- *an edge of A passes through an edge of B*

To compute the collision, we consider that any point has a linear uniform trajectory between time 0 and Δt.

By refining the mesh, the model can handle collisions with resolution as fine as necessary: the mesh is adapted (that is some masses are added to it) to the shape of the object in contact. A subdivision appears in two cases, when a collision occurs on an edge or a face of the deformable object (see *Figure 3.*). At the collision point, a node is added into the mesh. The problem is then to find the physical properties of the new nodes and links. 0 order (total mass), 1 order (centre of mass) and 2 order momenta preservation result in the trivial solution: to add a mass whose value is 0. In other words, it is strictly impossible to add a node into a mesh without disturbing the physical properties of the mesh. The subdivision method proposed by [HPH96] results in a pleasant solution, however, the total mass of the mesh is not kept. In an interactive case, for example, if the surgeon holds the organ, he would be quite disturbed by a sudden increase of the organ mass!

By removing some constraints, it is possible to get a system with infinite number of solutions where only those with positive masses are interesting. We choose to keep the total mass constant and nodes masses positive, that is the 0 and 1 order momenta are kept constant, but 2 order momenta are left unconstained.

Here is the case of a collision of a rigid point onto a deformable triangle *ABC*. The collision point can be expressed (in only one way) as:

$$P = \alpha A + \beta B + \gamma C \text{ with } \alpha + \beta + \gamma = 1 \qquad (3)$$

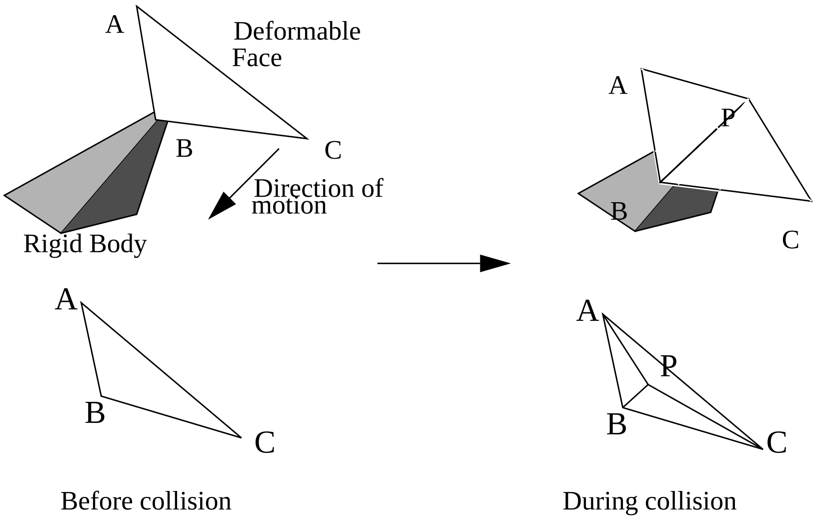


Figure 3. Collision of a point of the rigid body onto a face of the deformable body

The mass of Point *P* is obtained by taking away a part of the masses of points *A*, *B* and *C*:

$$m_t^P = am_{t-\Delta t}^A + bm_{t-\Delta t}^B + cm_{t-\Delta t}^C \tag{4}$$

The total organ mass has to be constant so:

$$m_t^A = (1-a)\, m_{t-\Delta t}^A \qquad m_t^B = (1-b)\, m_{t-\Delta t}^B \qquad m_t^C = (1-c)\, m_{t-\Delta t}^C \tag{5}$$

We suppose that the parts of the masses of points *A*, *B* and *C* transferred to the mass of point *P* depends directly upon the location of *P* in the triangle:

$$\frac{a}{\alpha} = \frac{b}{\beta} = \frac{c}{\gamma} \tag{6}$$

Finally after the mass transfer, a kind of mass coherency is enforced by saying that the point *P* mass is such as:

$$m_t^P = \alpha m_t^A + \beta m_t^B + \gamma m_t^C \tag{7}$$

When the system is solved, it can be found that:

$$\frac{a}{\alpha} = \frac{b}{\beta} = \frac{c}{\gamma} = \frac{1}{1 + \dfrac{\alpha^2 m_{t-\Delta t}^A + \beta^2 m_{t-\Delta t}^B + \gamma^2 m_{t-\Delta t}^C}{\alpha m_{t-\Delta t}^A + \beta m_{t-\Delta t}^B + \gamma m_{t-\Delta t}^C}} \tag{8}$$

It can be easily proved that *a*, *b* and *c* are always bounded by 0 and 1, which guarantees that the new masses are always positive. The same kind of calculation can be used to evaluate the new values of the air viscosity coefficients. It is wise to check the order of magnitude of the new masses of *P*, *A*, *B* and *C*, after the subdivision, because these masses can be quite low, and therefore incompatible with the shannon limits. A good idea would be to compute the system at different time step. The lowest time step corresponds to only a few nodes of low masses, which are in contact, and the largest time step corresponds to the rigid component.

The next step is to compute the stiffness of the springs of the new edges. The stiffness of the spring which links the new node *P* to the rigid component of the deformable object is obtained by averaging the elasticity of the springs of *A*, *B* and *C*, respectively with α, β and γ. For the new springs on the surface mesh, the stiffness values are harder to find. Several possibilities exist, but, none is satisfying. First, a sort of angular elasticity constant may be considered. For instance, the springs linking *A* to *P* is obtained by averaging the stiffness of springs *AC* and *AB* affected by the ratios of the angles *BAP* and *CAP* over *CAB*. Another solution is to assume that the ratio of the stiffness over the rest length of the spring is constant. All these solutions are not satisfying because the behaviour of *A*, *B* and *C* are disturbed by the strain of the new springs stretching on them. [TC90] have proposed a solution to split a spring in two, without changing the behaviour of the system. However, they weren't interested in inserting springs inside a mesh.

6. Implementation - Discussion

Our model is still under construction. It has been designed from the beginning to be compatible with all the aspects of the simulation, because it is not possible to study separately deformations, collisions and cuttings, since everything is controlled by the underlying model.

Simulation is interactive, even if synchronicity is not reached yet (see screenshots on *Figure 5.*). We have tested our model with deformable bodies with different number of nodes. These tests (see table on *Figure 4.*) have been measured on three different platforms to show how powerful the processor has to be. This measures show that collision detection is responsible of 90% of the calculation time.

Even if our model is not finished yet (collisions, rendering...) we plan to adapt it to a simulation of gynaecologic laparoscopy currently under development. This environment include a large number of organs which are likely to be operated such as ovary or womb, and other organs which only disturb the operation. It seems not a good idea to simulate all these numerous organs by complex models, such as Finite elements, which are furthermore not sufficient to describe rigid motions: in abdominal cavity, the organs are under pressure and are submitted, naturally, to gravity. A simple model as described in this paper will make this operation possible to be simulated realistically in realtime.

	R4600, 133MHz	P Pro, 200 Mhz	R10000, 194MHz
12 nodes	630Hz/3500Hz	880Hz/10680Hz	2500Hz/14000Hz
120 nodes	36Hz/330Hz	60Hz/1300Hz	180Hz/1450Hz
258 nodes	18Hz/145Hz	25Hz/610Hz	60Hz/650Hz

Figure 4. Simulation rates
(the two frequencies correspond respectively to simulations with and without collision détection)

7. References

[BC96] Bro-Nielsen, M., and Cotin, S., “Realtime Volumetric Deformable Models for Surgery Simualtion using Finite Elements and Condensation” *Proceedings of the EUROGRAPHICS’96 conference, Computer Graphics Forum,* 15, 3, Poitiers, 28-30 August 1996, pp 57-66.

[Boy79] Boyse, J. W., “Interference Detection among Solids and Surfaces” *Communications ACM*, vol 22, 1, january 1979, pp 3-9.

[CDBA96] Cotin, S., Delingette, H., Bro-Nielsen, M., Ayache, N., Clément, J.M., Tasseti, V., and Marescaux, J., “Geometric and Physical Representations for a Simulator of Hepatic Surgery”, *Proceedings of the Medecine Meets Virtual Reality IV*, San Diego, california, january 17-20 1996.

[CEO93] Cover, A.S., Ezquerra, N.F., O’Brien, J., Rowe, R., Gadacz, T., and Palm, E., “Interactively Deformable Models for Surgery Simulation” *IEEE Computer Graphics and Applications*, 13, 6, november 1993, pp 68-75.

[Gas89] Gascuel, M.P., “OSEA : un nouveau modèle de matière pour traiter les collisions entre objets déformables” (*in french*) *Actes de PIXIM 89,* pp 309-323.

[Gou94] Gourret, J.P., *Modélisation d’Images Fixes et animées*, chapitre VI “Assemblage et Animation d’Objets Déformables : Simulation par Eléments Finis” (*in french*), Masson, Paris, 1994.

[GTT89] Gourret, J.P., Magnenat Thalmann, N., and Thalman, D., “Simulation of Object and Human Skin Deformations in a Grasping Task” *SIGGRAPH’89 Conference Proceedings, Computer Graphics* 23, 3, Boston, 31 july - 4 august 1989, pp 21-30.

[HPH96] Hutchinson, D., Preston, M., and Hewitt, T., “Adaptive Refinment for Mass/Spring Simulations”, *Eurographics Workshop on Animation and Simulation*, Poitiers, 31 August - 1 september 1996, pp 31-45.

[HTB95] Henry, D., Troccaz, J., and Bosson, J.-L., “Virtual Echography: Simulation of an ultrasonic Examination” *Proceedings of Computer Assisted Radiology CAR’95*, berlin, 21-24 june 1995.

[KKKH95] Kühnapfel, U. G., Krumm, H. G., Kuhn, C., Hübner, M., and Neisius, B., “Endosurgery Simulations with KISMET : A Flexible Tool for Surgical Instrument Design, Operation Room Planning and VR Technology Based Abdominal Surgery Training” *Proceedings of “Virtual World”* 1995, pp 165-171.

[LJFC91] Luciani, A., Jimenez, S., Florens, J.L., Cadoz, C., and Raoult, O., “Computational Physics: a Modeler Simulator for Animated Physical Objects” *Eurographics Workshop on Animation and Simulation*, Vienne, 1991, pp 425-437.

[MKCD95] Meseure, P., Karpf, S., Chaillou, C., Dubois, P., and Rouland, J.F., “Low-cost Medical Simulation: a Retinal Laser Photocoagulation Simulator” *Proceedings of the Graphics’Interface 95 Conference*, Québec City, Québec, 17-19 may 1995, pp 155-162.

[Pro95] Provot, X., “Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour” *Proceedings of the Graphics Interface’95 Conference*, Québec City, Québec, 17-19 may 1995, pp 147-154.

[SBM94] Sagar M. A., Bullivant D., Mallinson G. D., Hunter P. J, and Hunter J. W., “A Virtual Environment and Model of the Eye for Surgical Simulation” *SIGGRAPH’94 Conference Proceedings, Computer Graphics annual conference series*, Orlando, 24-29 july 1994, pp 205-212.

[SPHL95] Sinclair, M. J., Peifer, J.W., Haleblian, R., Malcolm, N.L., Green, K., and Hull, D.S., “Computer simulated Eye Surgery - A Novel Teaching Method for Residents and Practitioners” *Ophthalmology* 1995 (102), pp 517-521.

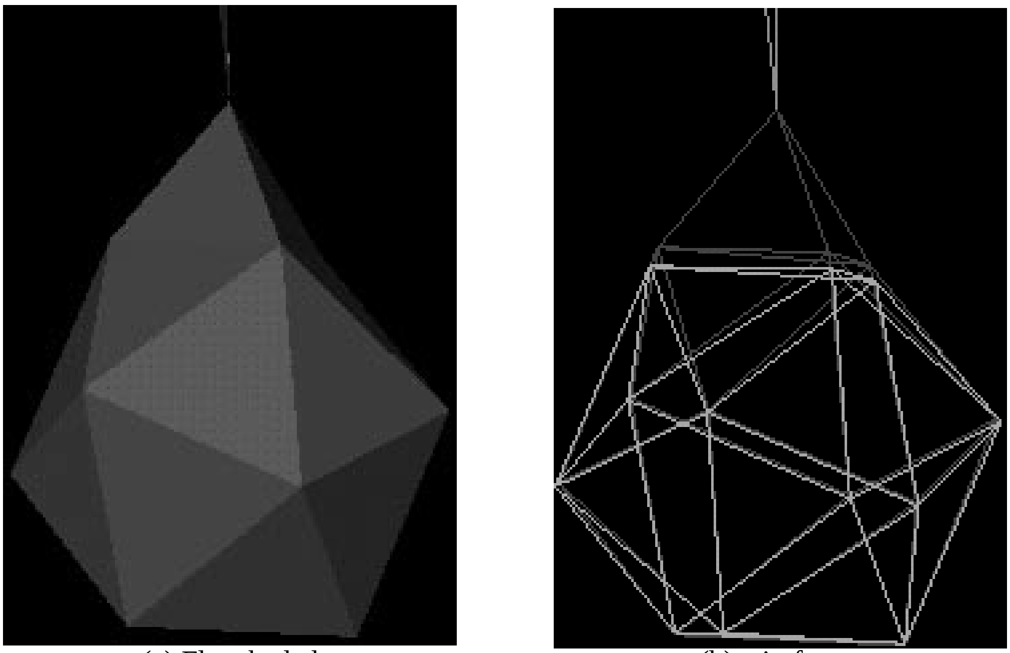
[TC90] Thingvold, J.A., and Cohen, E., “Physical Modeling with B-Spline Surfaces for Interactive Design and Animation” *Proceedings of the Symposium on Interactive 3D Graphics, Computer Graphics*, 24, 2, Snowbird, 25-28 march 1990, pp 129-137.

[TF88] Terzopoulos, D., and Fleisher, K., “Modeling Inelastic Deformation : Viscoelasticity, Plasticity, Fracture” *SIGGRAPH’88 Conference Proceedings, Computer Graphics*, 22, 4, Atlanta, 1-5 august 1988, pp 269-278.

[TW88] Terzopoulos, D., and Witkin, A., “Physically Based Models with Rigid and Deformable Components” *IEEE Computer Graphics and Application*, november 1988, pp 41-51.

[ULC94] Uhl, C., Luciani, A., and Cadoz, C., “Le Simulateur Temps-réel TELLURIS pour le Langage CORDIS-ANIMA : Modèles physiques, Gestes, Sons, Images”(*in french*) *Troisième séminaire du groupe de travail “Animation et Simulation”* Lille, 20-21 october 1994, pp 59-73.

[VC95] E. Varlet, and C. Chaillou, “Etude d’un simulateur d’écho-endoscopie bilio-pancréatique” (*in french*) *Actes des 3èmes Journées de l’Afig*, Marseille, France, 22-24 november 1995.



(a) Flat-shaded (b) wireframe

Figure 5. Example of deformation: an icosahedron held by a tool
On (b), the deformable body is dark, the rigid body is light