

P³DMA: A Physical 3D Deformable Modelling and Animation System ^{*}

Miquel Mascaró Portells, Arnau Mir y Francisco Perales

Departament de Matemàtiques i Informàtica
Unitat de Gràfics i Visió
Universitat de les Illes Balears
Crta. de Valldemossa, Km. 7,5
07071 Palma de Mallorca
<http://dmi.uib.es/research/GV>

Abstract. Animation and Realistic Simulation of a 3D object's elastic deformation is actually an important and challenging feature in applications where three-dimensional object interaction and behaviour is considered or explored. Also, in interactive environments we need a rapid computation of deformations. In this paper we present a prototype of a system for the animation and simulation of elastic objects in an interactive system and under real-time conditions. The approach makes use of the finite elements method (F.E.M) and Elasticity Theory. The simulation is interactively visualized in an Open Inventor environment. Using picking node selection the user can interactively apply forces to objects causing their deformation. The deformations computed with our approach have a physical interpretation based on the mathematical model defined. Furthermore, our algorithms perform with either 2D or 3D problems. Finally, a set of results are presented which demonstrate performance of the proposed system. All programs are written in C++ using POO, VRML and Open Inventor tools. Real time videos can be visualized on web site: <http://dmi.uib.es/people/mascport/defweb/dd.html>

Key Words: Elastic Deformation, Finite Elements Method, Elasticity Theory, Computer Animation, Physical Models, VRML.

1 Introduction and Related Work

Obviously, flexible and deformable objects are inherently more difficult to model and animate in computer graphics than rigid objects. Until recent years, the computer graphic methods proposed were limited to modelling rigid objects. However, recent advances in algorithms and computer graphics hardware support the processing of flexible objects. Today, there is a great need in many engineering and medical applications to be able to simulate the material and geometrical behaviour of 3D objects under real forces. In general, different modelling techniques are usually classified into three categories: geometrical, physical and hybrid:

^{*} This work is partially subsidized by CICYT under grant TIC2001-0931 and by UE under grant Humodan-IST.

Geometrical techniques. Geometrical models do not consider the physical properties of objects. They focus on appearance and the deformation is represented by geometrical equations. This means that the user has a high degree of intervention and they are computationally faster than other approaches.

Physical techniques. In this group of techniques, the objects are modelled as a triangular or rectangular grid in 2D or voxelated volume in 3D. Each joint or node in the grid can be affected by forces and the global grid is governed by the interaction of physical forces on each node considered. This kind of methods are more realistic from the mathematical and physical viewpoint and the user only defines the initial conditions and the system can simulate a real physical simulation over time. Unfortunately, they are more computationally expensive than geometrical techniques.

Hybrid techniques. Finally, we can combine physical and geometrical methods to avoid problems and improve efficiency.

In particular, the growth in hardware graphics can overcome the time-consuming restrictions of physically based methods. So in this paper we present our system called P³DMA. It is based on Finite Element Methods (F.E.M) and uses the Elasticity Theory. As we know, the solid theory used guarantees the robustness of the system and is actually widely used by other researchers [NNP02]. Thus, we are principally interested in designing a system that can run in real or near real time systems. We believe that in Virtual Reality systems the time in interaction and feedback is very critical. In this case, the efficiency of implementation is very important and results must be checked to reach this condition. In some cases, an initial off-line process can be introduced to improve efficiency.

This paper is organized in several sections. The second section includes the mathematical model proposed. The third section is dedicated to presenting the F.E.M implemented. The fourth section includes the algorithm designed to resolve the dynamical system. Finally, we conclude with some considerations about parallelization, efficiency and computational cost. The paper also includes the conclusions, future work and related bibliography.

2 Mathematical Model Proposed.

Let Ω be an enclosed and connected solid in \mathbb{R}^3 . Let us assume that the boundary of Ω , Γ , is C^1 piecewise. We divide Γ into two parts, Γ_0 and Γ_1 , where Γ_1 is the part of the boundary which receives external forces and Γ_0 is the fixed part of the boundary whose size we assume to be strictly positive. Note that boundary Γ does not necessarily need to be connected, which will enable us to simulate deformations of objects with holes.

The aim of this work is to study and analyse the computational cost of the evolution of Ω under the action of external forces \mathbf{f} on the inside and external sources \mathbf{g} on the boundary Γ_1 .

The position of the object is defined by the function $\mathbf{u}(t, \mathbf{x})$. Our problem is, therefore, reduced, given the functions \mathbf{u}_0 (initial position of the object) and

\mathbf{u}_1 (initial speed), to finding the position $\mathbf{u}(t, \mathbf{x}) = (u_1, \dots, u_3)$ of these in the domain $Q_T = \Omega \times (0 \times T)$ which will verify the following evolution system in time:

$$\left\{ \begin{array}{l} \rho \frac{\delta^2 u_i}{\delta t^2} - \sum_{j=1}^3 \frac{\partial}{\partial x_j} \sigma_{ij} = f_i, \quad i = 1, 2, 3 \text{ en } Q_T, \\ u_i = 0, \quad i = 1, 2, 3 \text{ en } \Gamma_0 \times (0, T), \\ \sum_{j=1}^3 \sigma_{ij} n_j = g_i, \quad i = 1, 2, 3 \text{ en } \Gamma_1 \times (0, T), \\ u_i(\cdot, 0) = u_{0,i}, \quad i = 1, 2, 3 \text{ en } \Omega, \\ \frac{\partial u_i}{\partial t}(\cdot, 0) = u_{1,i}, \quad i = 1, 2, 3 \text{ en } \Omega. \end{array} \right. \quad (1)$$

where functions σ_{ij} are the components of the tension tensor, n_j are the components of the normal vector at a point on the surface of the domain $\Gamma_1 \times (0, T)$ and ρ is the density of the object.

The resolution of the above problem is carried out by variational formulation. The solution of a discrete approximation \mathbf{u}_h of the above formulation gives us the approximate solution to our problem.

Specifically, we consider a subspace V_h with a finite dimension $I = I(h)$ of Hilbert's space H defined by $H = \left\{ \mathbf{v} \in (H^1(\Omega))^3, \text{ tal, que } \mathbf{v} = 0 \text{ sobre } \Gamma_0 \right\}$.

Our problem is reduced to finding a function \mathbf{u}_h defined in Q_T solution to the following differential system:

$$\left\{ \begin{array}{l} \forall \mathbf{v}_h \in V_h, \quad \rho \frac{d^2}{dt^2}(\mathbf{u}_h(t), \mathbf{v}_h) + a(\mathbf{u}_h(t), \mathbf{v}_h) = L(\mathbf{v}_h), \\ \mathbf{u}_h(0) = \mathbf{u}_{0,h}, \\ \frac{d\mathbf{u}_h}{dt}(0) = \mathbf{u}_{1,h}, \end{array} \right. \quad (2)$$

where $a(\cdot, \cdot)$ is the bilinear continuous form defined by $a(\mathbf{u}, \mathbf{v}) = \sum_{i,j=1}^3 \int_{\Omega} \sigma_{i,j}(\mathbf{u}) \varepsilon_{ij}(\mathbf{v}) dx$, (\cdot, \cdot) is the following scale product defined for functions defined in Q_T : $(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^3 \int_{\Omega} u_i(\mathbf{x}) v_i(\mathbf{x}) d\mathbf{x}$ and $L(\mathbf{v})$ is the following continuous linear form on V_h : $L(\mathbf{v}) = \sum_{i=1}^3 \int_{\Omega} f_i v_i d\mathbf{x} + \int_{\partial\Omega} g_i v_i d\sigma$.

Let φ_i be a V_h base of functions. If we write the solution to look for \mathbf{u}_h as, $\mathbf{u}_h(t) = \sum_{i=1}^I \xi_i(t) \varphi_i$, the components ξ_i verify the next differential system:

$$\rho \sum_{i=1}^I \xi_i''(t) (\varphi_i, \varphi_j) + \gamma \sum_{i=1}^I \xi_i'(t) (\varphi_i, \varphi_j) + \sum_{i=1}^I a(\varphi_i, \varphi_j) \xi_i(t) = L(\varphi_j). \quad (3)$$

In the above system we have added a new term $(\gamma \sum_{i=1}^I \xi_i'(t) (\varphi_i, \varphi_j))$ to simulate a damping effect of the object. The above system, written in a matrix form, is:

$$\rho M \xi'' + \gamma M \xi' + K \xi = L, \quad (4)$$

with M and K as the mass and tension matrices respectively:

$$M = ((\varphi_i, \varphi_j)) \quad i, j = 1, \dots, I$$

$$K = (a(\varphi_i, \varphi_j)) \quad i, j = 1, \dots, I$$

By discretizing in time this last equation:

$$\begin{aligned} M \left(\frac{\rho}{\Delta t^2} + \frac{\gamma}{2\Delta t} \right) \xi(t + \Delta t) = \\ = L + \frac{\rho M}{\Delta t^2} (2\xi(t) - \xi(t - \Delta t)) + \frac{\gamma M}{2\Delta t} \xi(t - \Delta t) - K\xi(t). \end{aligned} \quad (5)$$

The simulation of different physical phenomena such as instantaneous blows, constant forces, waves, etc. are implicit in the expression of vector L .

3 F.E.M and K, M, L . Definition.

In order to choose the type of finite elements to use we will base our decision on two basic criteria: the type of finite element to be used must correctly transmit the propagation of the tensions in the direction perpendicular to each face of the finite element and the type of finite elements to be used must possibility a real time computational process.

This is why finite elements of a rectangular prism type will be chosen. This kind of finite element makes a right transmission of the propagation of the tensions in the direction perpendicular to each face of the finite element and gets a low computational cost.

Note that by means of the type of finite elements chosen it is possible to define uniform grids (grids which possess all the finite elements of an identical length), and non-uniform grids. This second type of grid let us make an approach of the boundary of Ω .

First of all, we will define the fundamental tool which will allow us to work with the finite elements: domain with a pair of points.

Let i and j be two arbitrary nodes of the grid of finite elements of the object. Let $\text{sup } \varphi_i$ be the set \mathbb{R}^3 where $\varphi_i \neq 0$.

$\Omega_{i,j}$ is defined as the integration domain of a pair of points (i, j) such as

$$\Omega_{i,j} = \text{sup } \varphi_i \cap \text{sup } \varphi_j .$$

3.1 Base functions

In the three-dimensional model, there are three types of base functions:

$$\varphi_i^{(1)} = (\varphi_i, 0, 0) \quad , \quad \varphi_i^{(2)} = (0, \varphi_i, 0) \quad , \quad \varphi_i^{(3)} = (0, 0, \varphi_i) . \quad (6)$$

The expression of φ_i is the same in each base function: it is that function which has a value of 1 in the i -th node and 0 in the other nodes.

3.2 Deformations tensor

The deformations tensor is defined by the following expression:

$$\varepsilon_{ij}(\mathbf{v}) = \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right), \quad 1 \leq i, j \leq n. \quad (7)$$

3.3 The tension matrix K

The internal bonds of the object can be seen in tension matrix K .

The components of matrix K are: $K_{ij} = K(\varphi_i^{(k)}, \varphi_j^{(k)})$, where $\varphi_i^{(k)}$ and $\varphi_j^{(k)}$ are the base functions defined in (6) and the expression of $K(\mathbf{u}, \mathbf{v})$ is the following where \mathbf{u} and \mathbf{v} are any H functions:

$$\begin{aligned} K(\mathbf{u}, \mathbf{v}) &= \lambda \int_{\Omega} \left(\sum_{k=1}^n \frac{\partial u_k}{\partial x_k} \right) \left(\sum_{k=1}^n \frac{\partial v_k}{\partial x_k} \right) dx_1 dx_2 dx_3 \\ &+ 2\mu \sum_{i,j=1}^n \int_{\Omega} \varepsilon_{ij}(\mathbf{u}) \varepsilon_{ij}(\mathbf{v}) dx_1 dx_2 dx_3, \quad \forall \mathbf{u}, \mathbf{v} \in H. \end{aligned} \quad (8)$$

The expression of matrix K is the following:

$$K = \begin{pmatrix} K(\varphi_i^{(1)}, \varphi_j^{(1)}) & K(\varphi_i^{(1)}, \varphi_j^{(2)}) & K(\varphi_i^{(1)}, \varphi_j^{(3)}) \\ K(\varphi_i^{(2)}, \varphi_j^{(1)}) & K(\varphi_i^{(2)}, \varphi_j^{(2)}) & K(\varphi_i^{(2)}, \varphi_j^{(3)}) \\ K(\varphi_i^{(3)}, \varphi_j^{(1)}) & K(\varphi_i^{(3)}, \varphi_j^{(2)}) & K(\varphi_i^{(3)}, \varphi_j^{(3)}) \end{pmatrix}. \quad (9)$$

The space of functions we consider is that generated by the polynomials $\mathbb{R}^3 < 1, x_1, x_2, x_3, x_1 \cdot x_2, x_1 \cdot x_3, x_2 \cdot x_3, x_1 \cdot x_2 \cdot x_3 >$. Therefore, the function φ_i will have a linear combination of these polynomials.

In order to find $K_{ij} := K(\varphi_i^{(1)}, \varphi_j^{(1)})$ for instance, we must carry out two integrals on the domain of the pair of points Ω_{ij} . Below, we write $\Omega_{ij} = \cup Q_k$, where Q_k is a square prism type finite element which forms part of Ω_{ij} .

Therefore, we can calculate K_{ij} in the following way: $K_{ij} = \sum_k K_{ij}^{(k)}$, where $K_{ij}^{(k)}$ corresponding to $K(\varphi_i^{(1)}, \varphi_j^{(1)})$, for instance, would have the expression:

$$K_{ij}^{(k)} = (\lambda + 2\mu) \int_{Q_k} \frac{\partial \varphi_i}{\partial x_1} \frac{\partial \varphi_j}{\partial x_1} dx_1 dx_2 dx_3 + \mu \int_{Q_k} \frac{\partial \varphi_i}{\partial x_2} \frac{\partial \varphi_j}{\partial x_2} + \frac{\partial \varphi_i}{\partial x_3} \frac{\partial \varphi_j}{\partial x_3} dx_1 dx_2 dx_3.$$

With k fixed, there will be $64 = 8 \times 8$ different $K_{ij}^{(k)}$ values since each square prism Q_k has a total of 8 vertices.

Let $Q_s = [0, 1] \times [0, 1] \times [0, 1]$. The calculation of each of the integrals that appear in the expression $K_{ij}^{(k)}$ can be carried out in the following way:

$$\int \int \int_{Q_k} \frac{\partial \varphi_i}{\partial x_i} \frac{\partial \varphi_j}{\partial x_j} dx_1 dx_2 dx_3 = V_k \int \int \int_{Q_s} \frac{\partial \varphi_i}{\partial x_i'} \frac{\partial \varphi_j}{\partial x_j'} dx_1' dx_2' dx_3',$$

where V_k is the volume of Q_k , $\frac{\partial \varphi_i}{\partial x_i} = \sum_j \frac{\partial \varphi_i}{\partial x_j'} \frac{\partial x_j'}{\partial x_i} = a_i \frac{\partial \varphi_i'}{\partial x_i'}$ and the variable change of Q_k a Q_s is $x_i' = b_i + a_i x_i$, $i = 1, 2, 3$.

The calculation of $\int \int \int_{Q_s} \frac{\partial \varphi_i}{\partial x_i} \frac{\partial \varphi_j}{\partial x_j} dx_1 dx_2 dx_3$ is quite simple as we are working with a standard cube.

In this way, the 64 possible values $K_{ij}^{(k)}$ can be obtained. It can be seen that there are only 8 different values.

3.4 The mass matrix M

The mass matrix M will be made up of nine sub-matrices whose expression is the following:

$$M = \begin{pmatrix} (\varphi_i^{(1)}, \varphi_j^{(1)}) & 0 & 0 \\ 0 & (\varphi_i^{(2)}, \varphi_j^{(2)}) & 0 \\ 0 & 0 & (\varphi_i^{(3)}, \varphi_j^{(3)}) \end{pmatrix}, \text{ where:}$$

$$(\varphi_i^{(1)}, \varphi_j^{(1)}) = (\varphi_i^{(2)}, \varphi_j^{(2)}) = (\varphi_i^{(3)}, \varphi_j^{(3)}) \neq 0 \text{ si } \Omega_{ij} \neq \emptyset.$$

In order to calculate (φ_i, φ_j) in an effective way, we will use a method of approximate integration using the vertices of the finite elements as nodes. That is, using: $(\varphi_i^{(1)}, \varphi_j^{(1)}) = \sum_{k|Q_k \subset \Omega_{ij}} \int_{Q_k} \varphi_i(\mathbf{x}) \varphi_j(\mathbf{x}) d\mathbf{x}$, we approximate the integration as:

$$\int_{Q_k} \varphi_i(\mathbf{x}) \varphi_j(\mathbf{x}) d\mathbf{x} \approx \sum_{l=1}^8 A_l \varphi_i(P_l) \varphi_j(P_l), \quad (10)$$

where P_l are the vertices of the finite element Q_k and A_l are the coefficients of the approximate integration formula.

In this way, we manage to make the mass matrix M diagonal since $\varphi_i(P_l) = \delta_{il}$. Furthermore, since the numerical integration error is less than the error we make in the variational approximation of the problem which is in the order of h^3 where h is the maximum length of the sides Q_k (see [Cia80]), the use of the integration method does not increase the overall error in the approximation.

In order to find $\int_{Q_k} \varphi_i(\mathbf{x}) \varphi_j(\mathbf{x}) d\mathbf{x}$, we will move onto a standard cube $Q_s = [0, 1] \times [0, 1] \times [0, 1]$ by the adequate change in variable and there we will use the expression (10). In this way coefficients A_l will not depend on the finite element Q_k chosen. In the standard cube, coefficients A_l equal: $A_l = \frac{1}{8}$.

By the way,
 $\int_{Q_k} \varphi_i(\mathbf{x}) \varphi_j(\mathbf{x}) d\mathbf{x} = \frac{\delta_{ij} V_k}{8}$.

3.5 The external force vector L

The external force vector L will be of the type:

$L = \left(L(\varphi_i^{(1)}), L(\varphi_i^{(2)}), L(\varphi_i^{(3)}) \right)^T$, where $L(\varphi_i^{(k)})$, $k = 1, 2, 3$, are dimension vectors N with N as the number of nodes of the grid of finite elements which does not belong to Γ_0 , that is, non fixed nodes.

The expressions of the vectors $L(\varphi_i)$, $L(\varphi'_i)$ and $L(\varphi''_i)$ are the following:

$$L\left(\varphi_i^{(k)}\right) = \int_{\text{sup } \varphi_i} f_k(\mathbf{x})\varphi_i(\mathbf{x})d\mathbf{x} + \int_{\partial\Omega \cap \text{sup } \varphi_i} g_k(\mathbf{x})\varphi_i(\mathbf{x})d\sigma,$$

where $k = 1, 2, 3$.

In all the experiments carried out, we have assumed that $f = 0$. Therefore, the first term in the above expressions will be null.

If the external forces \mathbf{g} applied on the boundary are constant, the above expressions are reduced to: $L\left(\varphi_i^{(k)}\right) = g_k \int_{\partial\Omega \cap \text{sup } \varphi_i} \varphi_i(\mathbf{x})d\sigma$.

Our problem is, therefore, reduced to finding $\int_{\partial\Omega \cap \text{sup } \varphi_i} \varphi_i(\mathbf{x})d\sigma$.

We will assume that the boundary of Ω is approximated by square prism type finite elements. Therefore, we have the case in which the integration domain $\partial\Omega \cap \text{sup } \varphi_i$ will be: $\partial\Omega \cap \text{sup } \varphi_i = \cup \Pi_{ik}$, where Π_{ik} are flat rectangles situated on a plane $x_i = \text{constant}$.

We have, therefore, the case in which the value of the above integral can be calculated as: $\int_{\partial\Omega \cap \text{sup } \varphi_i} \varphi_i(\mathbf{x})d\sigma = \sum_k \int_{\Pi_{ik}} \varphi_i(\mathbf{x})d\sigma$

The above integral $\int_{\Pi_{ik}} \varphi_i(\mathbf{x})d\sigma$ can be reduced by the variable change adapted to a double integral on the standard square $[0, 1] \times [0, 1]$.

4 Dynamic Solution System, Parallelization and Computational Cost

The matrix of the system (5) is diagonal, so the i -th component of the value $\xi(t + \Delta t)$ can be obtained as:

$$\xi(t + \Delta t)(i) = \frac{L(i) + \xi_1(i) + \xi_2(i) - \xi_3(i)}{M_1(i, i)}, \quad i \in \{0, \dots, 3N\},$$

where

$$\begin{aligned} M_1 &:= M \left(\frac{\rho}{\Delta t^2} + \frac{\gamma}{2\Delta t} \right), \quad \xi_1 := \frac{\rho M}{\Delta t^2} (2\xi(t) - \xi(t - \Delta t)), \\ \xi_2 &:= \frac{\gamma M}{2\Delta t} \xi(t - \Delta t), \quad \xi_3 := K \xi(t). \end{aligned}$$

4.1 Parallel computation of the tension matrix K

Using the fact that component ij of matrix K can be calculated as $K_{ij} = \sum_{Q_k \subset \Omega_{ij}} K_{ij}^{(k)}$, we can parallelize the computation of K_{ij} by assigning the calculation of $K_{ij}^{(k)}$ to each CPU.

4.2 Parallel computation of the solution of the problem

If the external forces are variable during the deformation process the calculation of vector L will be depending of the time parameter.

If these forces are constant or constant in parts, we can calculate the value of the L vector in a parallel process. We assign to each CPU the compute of $\int_{\partial\Omega \cap \text{sup } \varphi_i} \varphi_i(\mathbf{x}) d\sigma$.

In the figure 1 the calculation process of the deformation is outlined, where the system, from the conditions of the material to be deformed and the application of the external forces in time, is capable of launching a totally parallelizable calculation process.

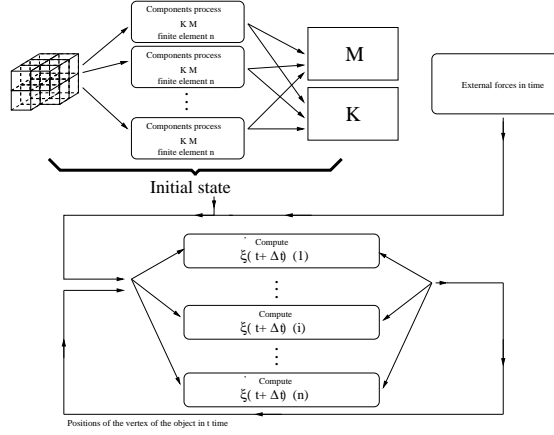


Fig. 1. Outline of the parallel calculation process of the dynamic solution system of deformations.

4.3 Computational cost of the calculation of the matrices K y M

The computational cost of the tension and mass matrices will be a function of the number of simultaneous processes that the system can bear, and a function of the quantity of finite elements of the object to be deformed.

Let ne the number of the finite elements of the object. Let np the number of simultaneous processes the computerized system can bear. Let Q_l an arbitrary finite element.

Let C_{MK_l} the computational cost associated to calculate $\sum_{i,j} (K_{ij}^{(l)}) + \frac{\delta_{ij} V_l}{8}$ and the additions to reach the final values of the matrices M and K associated to each independent process. Then we can define:

$$C_{MK} = \max_{1 \leq l \leq ne} C_{MK_l}.$$

Finally, we obtain the computational cost to calculate the matrices M, K :

$$Cost_{MK} \leq \frac{C_{MK} * ne}{np}.$$

So, the computational cost is $\frac{O(ne)}{np}$.

Computational cost of the calculation of an iteration of the solution

The computational cost of the solution of one iteration step in the dynamic system is linear $O(N)$ where N is the number of nodes in the grid of finite elements such that the matrix of the system is diagonal.

5 An Elastic Simulation Example

In figures 2, 3, and 4 we can see the simulation process of a 60 seconds elastic deformation. We apply some forces in a kind of a latex square tube of dimensions $0.028 \times 0.028 \times 0.224$ meters. We apply some F_x newton forces in the upper and lower marks during the first 24 seconds (left, figure 2). Between 12-24 seconds we apply some F_z newton forces in right and left marks (middle, figure 2). In two seconds time of the simulation process we can observe a little tube expansion and rotation (right, figure 2).

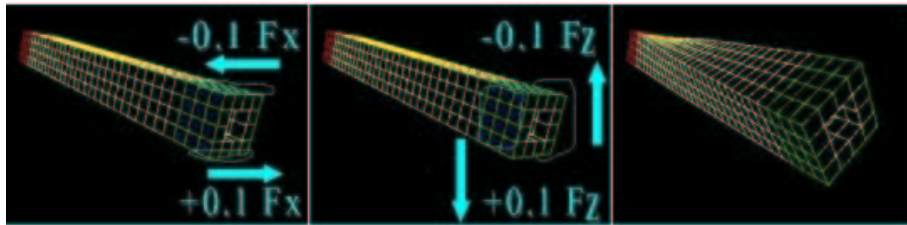


Fig. 2. *Initial forces conditions of the simulation and 2 sec. time of the deformation process.*

At 13 seconds time we can note the effect of the F_x forces result, a greater expansion and anti-clock wise rotation of the latex tube (left, figure 3). At 24 seconds time we can see the final state of the object after the F_x and F_z forces action (middle, figure 3). At 26 seconds the action forces have been stopped and the object is returning to its initial state (right, figure 3).

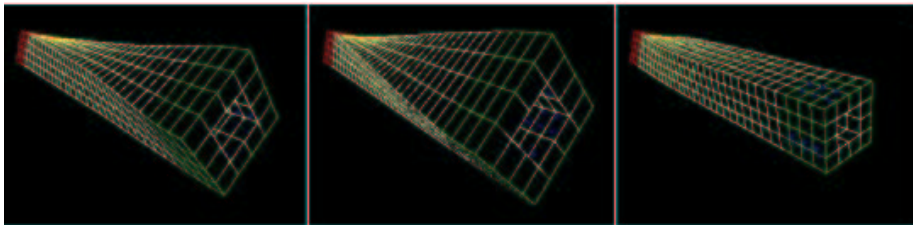


Fig. 3. *Left to right 13, 24 and 26 seconds time of the simulation process.*

At 27 seconds time the elastic energy of the tube has made a clock-wise rotation (left, figure 4). Here to the final state of the simulation process the object makes an armonic rotation balance to reach its initial state (middle-right, figure 4).

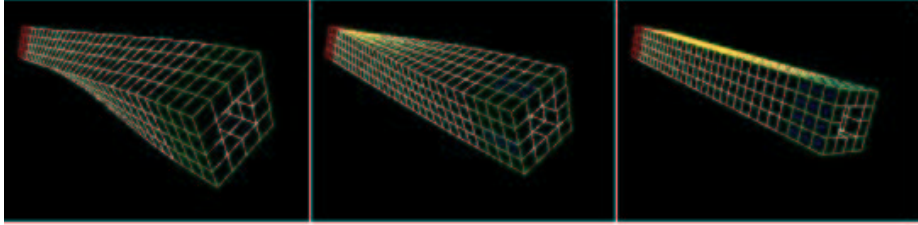


Fig. 4. *Left to right 27, 30 and 60 seconds time of the simulation process.*

6 Conclusions and Future Work

In this work we have presented a 3D deformation model based on the Theory of Elasticity. This model uses a rectangular parallepiped grid of finite elements that let us make a pre-computation values of the matrices M & K . That cause possibility a real time simulation of the deformation process. The main contributions can be summed up in the following points:

- The calculation of deformations is totally parallelizable. We assign the computation process for the finite elements grid matrix values to several CPU's. Values according to one voxel of the finite elements grid are assigned to one CPU. The final values for the tension and mass matrices are the additions of the partial results computed for each parallel process. The solution of the dynamic system can be parallelized too cause the system is diagonal.
- The chosen finite elements correctly transmit the tensions in a direction perpendicular to the boundary of the same. Therefore, the deformations presented are quite realistic. Whatever quadrangular parallelepiped voxel can be used in this deformation model to makes the finite elements grid of the object.
- The computational cost has been reduced so that the deformations can be represented in real time. Experiments have been realized using an ATHLON 900 CPU with 512 MB of RAM memory. The model supports real time process in an objetc with 12000 voxels.
- A study of the computational cost of the algorithm has been carried out. We reach a lineal computational cost depending on the number of voxels of the object and the quantity of simultaneous processes that the system can bear.

In the context of future work, we would like to highlight the following:

- A study of the deformations of two-dimensional varieties such as the study of clothes deformations or the deformation of the human skin.
- A study of deformations when internal forces are not null, $f \neq 0$.
- A study of deformations when the external force g depends on the speed of the object. Integrating inertial forces to the model and collision checking techniques.

References

- [AnsOLK00] Anshelevich, E., Owens, S., Lamiraux, F., Kavraki, L. E. Deformable Volumes in Path Planning Applications. ICRA 2000, Department of Computer Science, Rice University, Houston, 2000.
- [Aon90] Aono, M. A. Wrinkle Propagation Model for Cloth. Proc. CG. Int'l, Springer-Verlag, 95-115, 1990.
- [Bro94] Bro-Nielsen, M. Modelling Elasticity in Solids using Active Cubes - Application to Simulated Operations. Institute of Mathematical Modelling Technical University of Denmark. 1994.
- [Cia80] Ciarlet, P.G.. The Finite Element Method for Elliptic Problems. Université Pierre et Marie Curie, Paris. North-Holland Publishing Company, Amsterdam-New York, Oxford. 1980.
- [CohC91] Cohen, L. D., Cohen, I. Finite Element Methods for Active Contour Models and Ballons for 2D and 3D Images. CEREMADE, France. Cahiers de Mathématiques de la Décision, MD n° 9124 November 1991.
- [DebDBC99] Debunne, G., Desbrun, M., Barr, A., Cani, M. P. Interactive multi-resolution animation of deformable models. Computer Animation and Simulation '99. Eurographics Workshop, Milan, 1999.
- [Dec96] Decaudin, P. Geometric Deformation by Merging a 3D-Object with a Simple Shape. INRIA. Graphics Interface'96 proceedings, Toronto. 1996.
- [ErdDJ01] Erdan, G., Duaqing, X., Jingbin, W., Chen, C. Dynamic Cloth Motion Simulation Using a Novel Fast Collision Detection. Computer Science and Engineering Department, ZheJiang Univ. Conference Paper, 2001.
- [FalPT97] Faloutsos, P. and M. van de Panne. And Demetri Terzopoulos. IEEE Transaction on Visualization and Computer Graphics, vol 3, n. 3, 1997.
- [FanRR94] Fang, S. and Raghavan, R. and Richtsmeier, T. Volume Morphing Methods for Landmark Based 3D Image Deformation. National University of Singapore, - School of Medicine, Baltimore, 1994.
- [Far97] Farin, G. Curves and surfaces for computer-aided geometric design: a practical guide. San Diego, California. London: Academic Press, ISBN: 0122490541, 1997
- [FenK98] Fenster, D., Kender, J.R. Sectorized Snakes: Evaluating Layered-Energy Segmentations. Department of Computer Science, Columbia University, New York, NY 1027 USA, 1998.
- [GibM97] Gibson, S.F.F. and Mirtch, B. A Survey of Deformable Modelling in Computer Graphics. Technical Report, Mitsubishi Electrical Research, 1997.
- [HagF98] Hagenlocke, M., Fujimura, K. CFFD: a tool for designing flexible shapes. The Visual Computer (1998) 14:271-287, Springer-Verlag 1998.

- [HinG96] Hing, N. Ng and Grimsdale, R. L. Computer Graphics Techniques for Modelling Cloth. IEEE Computer Graphics and Applications, 28-41, 1996.
- [KavLH98] Kavradi, L.E., Lamiroux, F., Holleman, C. Towards Planning for Elastic Objects. Department of Computer Science, Rice University (Houston). Workshop of Algorithmic Foundations of Robotics, pages 313-325, 1998.
- [MasMP00] Mascaró, M., Mir, A., Perales, F. Elastic Deformations Using Finite Element Methods in Computer Graphic Applications. AMDO 2000, pp. 38-47, 2000. Springer-Verlag Berlin Heidelberg 2000.
- [McIT97] McInerney, T., Terzopoulos, D. Dept. of Computer Science, University of Toronto, Canada. Published in the Proc. CVRMed'97, Grenoble, France, march, 1997.
- [McQW00] McDonnell, K.T., Quin, H., Wlodarczyk, R.A. . Dept. of Computer Science, University of New York at Stony Brook.
- [Met97] Metaxas, D. N. Physics-Based Deformable Models. University of Pennsylvania, USA. KluwerAcademic Publishers, 1997.
- [MonDSY00] Montagnat, J., Delingette, H., Scapel, N., Ayache, N. Representation, shape, topology and evolution of deformable surfaces. Application to 3D medical image segmentation. INRIA, 2000.
- [NNP02] I. Nikitin, L. Nikitina, P. Frolov. Real Time simulation of elastic objects in Virtual Environments using finite element method and pre-computed Green's functions. Eight EG Workshop on VE, 2002, 47-52, S. Muller, W. Sturzlinger (editors), 2002.
- [NurRR01] Nürnberger, A., Radetzky, A., Kruse R. Using recurrent neuro-fuzzy techniques for the identification and simulation of dynamic systems. Neurocomputing 36, 123-147, Elsevier Science B. V., 2001.
- [RavT92] Raviart, P. A., Thomas, J. M. Introduction à l'analyse numérique des équations aux dérivées partielles. Ed. Masson, 3^e edición, 1992.
- [Rud90] Rudomin, I.J. Simulating Cloth using a Mixed Geometry-Physical Method. Doctoral Dissertation, Dept. of Computer and Information Science, Univ. Of Pennsylvania, 1990.
- [ShiHCJ01] Shi-Min, H., Hui, Z., Chiew-Lang, T., Jia-Guang, S. Direct manipulation of FFD: efficient explicit solutions and decomposable multiple point constraints. The Visual Computer (2001) 17:370-379. Springer-Verlag 2001.
- [TerQ94] Terzopoulos, D., Qin, H. Dynamic NURBS with geometric constraints for interactive sculpting. ACM Transactions on Graphics, 13(3):103-136, 1994.
- [ThalCCVW98] Thalmann, N. M., Carion, S., Courchesne, M., Volino, P., Wu, Y. Virtual Clothes, Hair and Skin for Beautiful Top Models. MIRALab, University of Geneva, 1998.
- [ZabS99] Zabararas, N. Srikanth, A. An Object-Oriented Programming Approach to the Lagrangian FEM Analysis of Large Inelastic Deformations and Metal-Forming Processes. International Journal for Numerical Methods in Engineering. Int. J. Numer. Meth. Engng. 45, 399-445 (1999).