Interactive Physically-based Simulation of Virtual Objects Torsion

Benoît Le Gouis

Master research Internship

Internship report

Interactive Physically-based Simulation of Virtual Objects Torsion

---

Author:
Benoît Le Gouis

Supervisor:
Maud Marchal
Hybrid Team

Abstract

In the fields of virtual reality, physically-based simulation focuses on reproducing the motion and properties of real objects in virtual environments. Until now, various physically-based models of mechanical phenomena have been proposed in the literature, simulating several physical properties such as rigid, fluid or deformable behaviors of the multiple real-life objects. In this master thesis, we propose to simulate the torsion of any deformable object and its haptic rendering. We intend to create haptic and visual rendering of a virtual environment, in which objects can be twisted, either by the user or by the action of other objects of the environment. Our two contributions are first the introduction of a novel constraint for torsion modeling in a physically-based simulation, and second a novel coupling scheme for haptic rendering of 3D virtual object torsion.

Keywords : Virtual reality, Physically-based Simulation, Torsion, Continuum Mechanics, Haptic Rendering

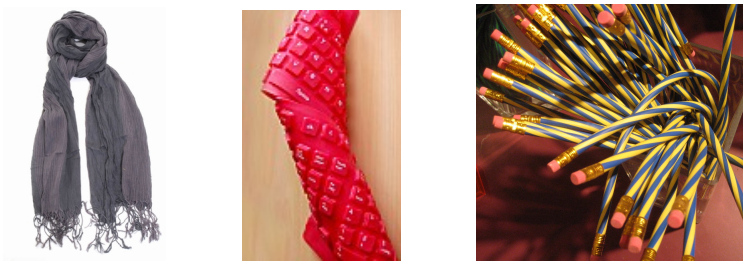
Contents

Chapter 1

Introduction

In the field of virtual reality, the interaction between a user and the virtual environment is performed through multi-sensory interfaces with different sensorial modalities, such as visual or haptic feedback. The quality of the interaction is partly due to the realism and perception of the virtual world. Among the different sensorial feedback, haptic means related to the sense of touch. For instance, a haptic interface provides a direct interaction, and allows to use a sense which provides a lot of information about objects.

As virtual reality is the interaction of a human and a virtual environment [11], physically-based models are developed in order to improve the quality of the interaction. They are focused on reproducing the motion and properties of real objects in virtual environments. Among the various physically-based models of mechanical phenomena proposed in the literature, several physical properties such as rigid, fluid or deformable behaviors of the multiple real-life objects have been simulated. For deformable objects, torsion is one of the key phenomenon, along with compression and stretching. As illustrated in Figure 1.1, torsion is involved when manipulating any kind of deformable object of our everyday life, including clothes, rubber, or even human organs.



(a) (b) (c)

Figure 1.1: Examples of deformable object object: (a) clothes, (b) rubber and (c) plastic

However, physically-based models of torsion have not been proposed in the context of virtual reality systems, especially with haptic rendering. For example, a virtual learning application could implement the interaction with flexible wires, and thus need accurate simulation and a good haptic feedback.

Figure 1.2 shows the entire interaction process in physically-based simulation of torsion. It shows the important steps in the interaction process. Thus, the user interacts through a haptic device with the virtual world. The forces for the haptic feedback are computed using the physical simulation. The properties of the virtual object allows the simulation of complex mechanical phenomena.

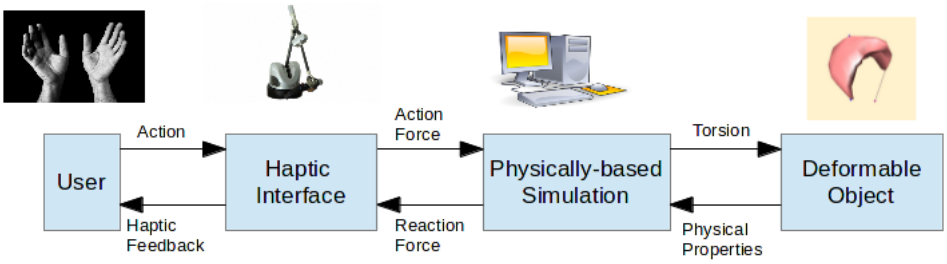


Figure 1.2: The interaction chain involved in the torsion of a deformable object

The three last blocks have two main common challenges. First of all, interaction needs real time methods. Lag introduces a hindrance for the user and should be avoided as much as possible. Real time simulations require efficient methods at each step of the process. Another challenge is realism, which is also a great advantage for interaction, as it makes feel the virtual world closer to real one. It involves equally the haptic device, the physically-based simulation and the deformable object, from the design of sensitive devices to the elaboration of appropriate models.

Our contributions are twofold. We first created a novel interaction method simulating the action of the hand on an object. We then propose a novel haptic rendering method for the force feedback of the torsion of an object.

The master thesis is organized as follows. Chapter 2 summarizes the related work for both for physically-based simulation of deformable objects and haptic rendering. Chapter 3 gives an overview of our work, including the framework designed for our contributions. Chapter 4 presents our first contribution, a novel constraint model for the physically-based torsion of deformable models. Chapter 5 details our second contribution, a novel coupling scheme for the haptic feedback of virtual object torsion. Chapter 6 describes our results, followed by a conclusion in the last chapter.

Chapter 2

Related Work

2.1 Physical Simulation of Torsion

In this section, we present the main principles relying to the physical simulation of torsion. This section is composed of two parts : first a summary of the main principles of the physics theory, and then the physical simulation of deformable objects under torsion in a virtual environment.

2.1.1 Mechanical Torsion

In order to have a good simulation of the mechanical torsion, we need some physics basis to work with. In this part, we will explain about the mechanical torsion, as considered in the continuum mechanics.

Introduction to Continuum Mechanics

Continuum mechanics is the study of all deformable materials, including deformable solids, as well as liquids and gases [12]. Our study will mainly focus on deformable solids [7]. A body will be considered as a continuous object. Its objective is to characterize the macroscopic behavior of a solid with all possible external constraints. The constraints can be for example compression, dilatation, flexion or torsion [4]. We focus on torsion, seen as an isotropic and reversible transformation, ie if no more torque is applied on the object, it will deform itself back to its initial state.

A few principles are essential in continuum mechanics, such as the mass conservation : a body will keep the same mass over a transformation. The body must also respect the Newton laws of motion. The first law is the following :

$$m\ddot{x} = \mathbf{f}_{ext} \tag{2.1}$$

where $\ddot{x}$ is the acceleration of a material point, with a position x, m its mass, and $\mathbf{f}_{ext}$ the sum of all external forces applied on this material point. This must be true for all material points.

In order to study the deformation of a deformable object, we have to study the displacement of all its material points. This displacement is the difference between the position of the material point at a time t and its original position, as shown in Figure 2.1.

With this deformation field, we can characterize the deformation of the

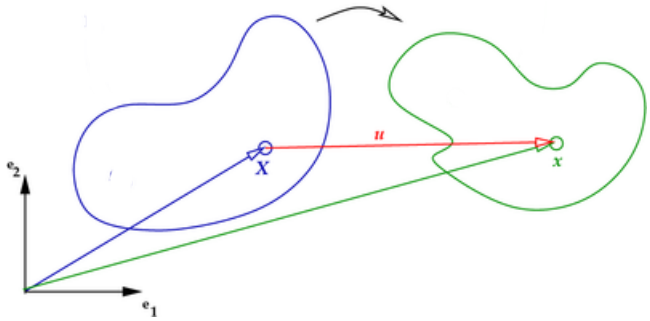


Figure 2.1: The displacement field $\mathbf{u}(\mathbf{x}, t)$

body, called the strain in mechanics. The strain measures the relative elongation of the body, for all dimensions. In 3D, the strain is related to the displacement field using a tensor of order 2 :

$$\epsilon_G(\mathbf{x}, t) = \frac{1}{2}(\nabla\mathbf{u}(\mathbf{x}, t) + [\nabla\mathbf{u}(\mathbf{x}, t)]^T + [\nabla\mathbf{u}(\mathbf{x}, t)]^T\nabla\mathbf{u}(\mathbf{x}, t)) \tag{2.2}$$

$$\epsilon_C(\mathbf{x}, t) = \frac{1}{2}(\nabla\mathbf{u}(\mathbf{x}, t) + [\nabla\mathbf{u}(\mathbf{x}, t)]^T) \tag{2.3}$$

ϵ_G is called Green-Lagrange non-linear tensor, and ϵ_C is its linearized version.

When a body is deformed, internal forces are created, aiming at bringing the body back to its initial state. These forces are the internal stresses of the body. They depend on the actual strain, and on the stiffness of the body. For a Hookean body, the relation between the stress σ and the strain is linear :

$$\sigma(\mathbf{x}, t) = E\epsilon_C(\mathbf{x}, t) \tag{2.4}$$

where E is the so-called Young's modulus, representing the stiffness of the body.

These equations can give a model for the deformation of a body, but there exist in general no analytical solution for such equations, requiring some approximations to have an estimation.

Definition of Torsion

The mechanical torsion is a rotation torque applied to two opposite parallel planes of an object. There is pure torsion when no other force is applied on the side face of the object, and the weight force is ignored. If other forces are applied, or if the torque is not only a rotation, the torsion must be combined with the other transformations, such as flexion. It is mainly used in theory of beams, as the corresponding simplifications make the equations solvable.

2.1.2 Physically-based Simulation

In the previous part, we have seen the theory about torsion. We will present numerical approaches in order to be able to calculate the deformation of our objects. Since a continuous model is not adapted to our finite representation of objects, a few approximation are made. In order to evaluate the different methods, we will consider three criteria which are essential for our problem. First we will consider the computational efficiency and more precisely the real-time possibility of the method. We will then consider the physical realism of the method, since we need a physically plausible method. Since many methods use 1D objects, we will also consider the 3D adaptation possibility.

Numerical Approaches

First, we are dealing with objects with a fixed mass, and therefore we need a representation that keeps the information of the mass, and allow methods for the mass conservation constraint. Among the representations presented in [20], we will focus on the Lagrangian mesh based methods, since the mesh free methods are more suitable for the representation of fluids, such as water or smoke [23].

A first possible representation is a mass-spring system [6]. The vertices of the mesh contains all the mass information, and the link between the vertices is made by springs. Any displacement of a vertex will lead to a modification of the adjacent vertices, until an equilibrium position is reached. This method provides fast simulation, but with poor realism, and compression will lead to significant global volume change, which is incompatible with the mass conservation constraint.

Another representation is the Boundary Element Method (BEM), which represents an object by its surface mesh, and therefore achieves all com-

putation on a two-dimensional representation of the object, providing fast simulation [14]. The drawbacks are that the object must be homogeneous, and topology modifications such as fractures lead to more important handling.

Finally, the most used representation is the Finite Element Method (FEM). It relies on a discretization of the object into a set of disjoint elements, and most commonly in computer science into a set of disjoint tetrahedra. Using directly the equations from continuum mechanics on finite elements would give perfectly realistic deformations, but with the complete loss of real-time. Some approximations, or optimization, must be done in order to gain computation time.

Some work rely on a hierarchy in the representation of the object itself. A rough mesh is used in [9] for the deformable object, requiring less computation time, and refine where the object is deformed, in order to keep a good accuracy in the deformation.
An underlying skeleton for the deformations is used in [5]. The main deformation is applied to the skeleton, and then refined in each segment. These two techniques use approximation, so are more useful when only good appearance is needed.
Other techniques study the physical constraints, in order to find a way to solve them a better way. It's the approach of [13] and [19]. They extract for each element the rotational part of the deformation gradient, then apply the corotation. With this transformation, the Cauchy's tensor is made linear. The strain system, made linear, can be solved more efficiently, with better accuracy. In order to have the proper deformation, the rotation must be re-applied after. The method of Müller et al. is one of the most used in physically-based simulation as of today.
[21] use a QR decomposition, Q being an orthogonal matrix, and R being an upper triangular matrix. This decomposition also extracts a rotation matrix (Q) but significantly faster than the previous method. The drawbacks are an anisotropy introduced by the choice of axes for the rotation, and the strain calculated is a bit higher. These drawbacks can be compensated by a refinement of the mesh, allowed by the computational efficiency.

Time Integration

Most of the motion equations rely on differential equation, that need to be solved numerically. With $\mathbf{x}$ the position vector, the Newton's second law of motion will be

$$\ddot{\mathbf{x}} = F(\dot{\mathbf{x}}, x, t) \tag{2.5}$$

In order to have only first order equations, the equation is rewritten.

$$\dot{\mathbf{x}} = \mathbf{v} \tag{2.6}$$
$$\dot{\mathbf{v}} = F(\mathbf{v}, \mathbf{x}, t) \tag{2.7}$$

A few solution exist to integrate this equation. The explicit (of forward) Euler integration simply transforms the times derivatives by finite difference.

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \mathbf{v}(t) \tag{2.8}$$
$$\mathbf{v}(t + \Delta t) = \mathbf{V}(t) + \Delta t F(\mathbf{v}(t), \mathbf{x}(t), t) \tag{2.9}$$

This model gives explicit values for $\mathbf{x}$ and $\mathbf{v}$, but is stable only for small Δt. Another method is the implicit (or backward) Euler integration, keeping $t + \Delta t$ on the right side of the equation.

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \mathbf{v}(t + \Delta t) \tag{2.10}$$
$$\mathbf{v}(t + \Delta t) = \mathbf{V}(t) + \Delta t F(\mathbf{v}(t + \Delta t), \mathbf{x}(t + \Delta t), t) \tag{2.11}$$

This requires more computation, since the system must be solved, but provides a stable solution to the problem.

1D Models

Many 1D models have been dedicated to torsion. First of all, the beam theory in continuum mechanics has led to a consequent number of beam-based models for simulation. Beam are a model for 1D objects, with a radius small compared to the length. Hair is a good example for a beam, since the radius can be ignored, compared to the length. In a first approach, called the Absolute Nodal Coordinate (ANC), point have coordinates referring to their material frame. A material frame is centered on the centerline of the beam, have two axes in order to orientate the frame, and one collinear to the centerline, as shown in Figure 2.2. This allows a better representation for continuum mechanics based methods. This representation has been used in several articles, including [10], [15] and [26]. It provides good results, with good simulation of flexion, compression, dilatation and torsion for beams, with sufficient computation time.

A comparison of ANC and FEM is done in [24]. It shows that FEM provide better results in linearized cases, and ANC provides better results for an elastic line.
The Lagrange multipliers are used in [25], with the hypothesis of an inextensible deformable rod. They achieve linear complexity. They are thus able to simulate any deformation of the rod, including friction phenomena. For instance Figure 2.3 shows a rod deformed with the Lagrangian method,

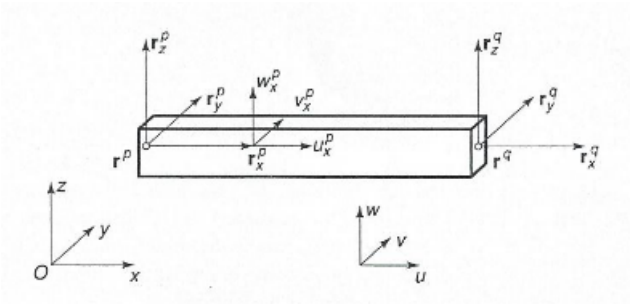


Figure 2.2: Nodal Coordinates, center being at $\mathbf{r}_x^p$, and the three corresponding axes. Image taken from [24]

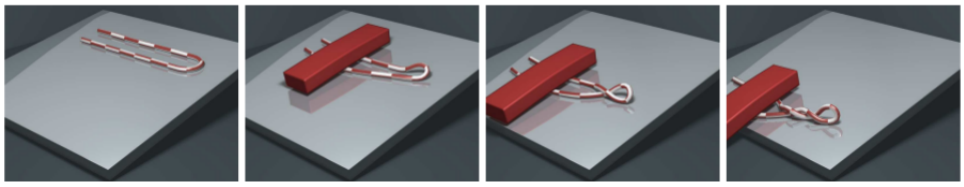


Figure 2.3: A rod deformed through the friction with a heavy object [25]

with good realism.

Strictly inextensible rods using Super-Helix are also simulated in [3]. The method uses recursively the Super-Helix in order to reach a linear complexity. Figure 2.4 shows the torsion of a rod for different resolutions using the Super-Helices method. This method is highly relevant for linear objects such as hair, but strongly depends on this linear aspect, and is thus difficult to adapt for 3D problems.

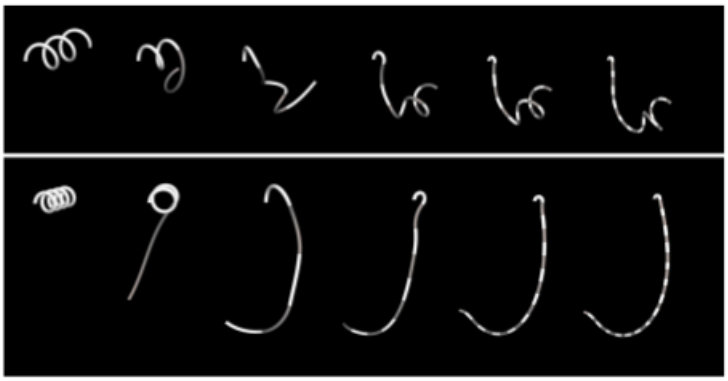


Figure 2.4: Multiple resolution rods deformed with the Super-Helices method [3]

Evaluation

	Real-Time	Realism	3D Torsion
Mass-Spring	+	-	++
BEM	++	+	+
FEM	- -	++	++
Beam	++	+	+
Super-Helices	++	+	-

Table 2.1: Comparison of the different numerical approaches : Three 3D representations (Mass-Spring, BEM and FEM) and two 1D representations (Beam and Super-Helices) are compared according to three criteria, the real-time, the realism and the 3D adaptation for torsion

In Table 2.1, the previously presented methods are compared, according to three criteria. First, the ability to create real-time models, then the realism of the methods, and finally the possible adaptation of the method for 3D torsion. These three criteria are essential for our problems, since simulation requires real-time methods for the interaction, the realism is also crucial, as explained in the introduction, and the possible 3D adaptation is also important, since we want to twist 3D objects.
First the mass-spring, with good real-time and 3D adaptation properties. Then the BEM, with really good real-time properties and good realism and 3D adaptation properties, the main drawback being the homogeneity constraint. Then the FEM, considered as if no calculus simplification was performed, and exact solving of equations. It has really good results for realism and 3D adaptation, but is not suitable for real-time simulation. Then the beam model for 1D torsion. This model has really good real-time properties, and good realism and 3D adaptation properties. Finally, the Super-Helices model, with really good real-time properties, good realism, but poor 3D adaptation.

2.2 Interaction Through Haptics

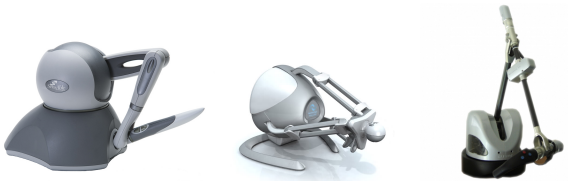
In the previous section, we have presented the physically-based simulation for the deformation of objects. The goal of the internship being the interactive torsion of the objects, we have now to present the interactive part of the torsion. This interaction is performed through haptic devices. We will hence present the State-of-the-Art concerning haptic interaction for deformable objects.

2.2.1 Haptic Feedback

Haptic feedback is the information provided through the sense of touch to the user. There are two kinds of haptic feedback. First the tactile feedback, providing information about the surface of the object, and then the kinesthetic feedback, providing information about the force applied to an object. We have chosen to only focus on the kinesthetic feedback in the context of our internship .

The force that can be applied on a deformable object will have different components, called degree-of-freedom (DOF). Each translation or rotation that can be applied on the object is a degree-of-freedom. Without any constraints, an object has six DOF, three translations and three rotations. For the torsion of deformable objects, we need these six DOF.

In order to interact with the object, we need an interface. For haptic feedback, we need a material device. The number of DOF available for the interaction strongly depends on the chosen device. For example, the devices in Figure 2.5 have all six DOF for positioning the object, but only the Virtuose (Haption, Soulgé sur Ouette, France) has six DOF of force feedback. The Phantom (Sensable, Wilmington, USA) and the Falcon (Novint, Washington, USA) have respectively three and up to five DOF for feedback. As



(a) (b) (c)

Figure 2.5: Different haptic devices : (a) Phantom, (b) Falcon and (c) Virtuose

we need six DOF for force feedback, we will use the Virtuose during the internship.

2.2.2 Haptic Rendering

Background

The forces applied by the virtual object to the user through the haptic interface is referred as haptic rendering. As explained in [16], there are two main methods for haptic rendering. First the *admittance rendering*, which calculates the forces involved for the object, and outputs a position of the controlled point. Then, the *impedance rendering* calculates the desired position, and outputs a device force on the user. Since our goal is to have

feedback on the torsion of the object, we will use the impedance rendering.

A few elements are mandatory in order to have a good haptic rendering. First, it is mandatory to have a good frequency. A major problem for interacting objects is the collision detection. For deformable objects this problem is even worse, since the shape can change, and any representation of the object has to be updated at each deformation. A good frequency is 1000 iterations per second for solid objects, hence a frequency of 1KHz. It is much more important than the required display rate for visual interface (about 30Hz). Since most applications with a haptic interface also have a visual interface (for example [17]), much work has been done in order not to need an entire calculus of the scene and of the resulting forces for each haptic time step.

Haptic Rendering for Simulation

Simulation requires important computation time, which is most of the time incompatible with the required frequency for haptic rendering. Models have been proposed to solve this problem. For instance, Mendoza et al. propose a method for deformable objects [18]. They manipulate a probe, and interact with a deformable object. Their idea is to continue the probe until they reach the object, and they only calculate the collision within a short range of the impact point. Davanne et al. propose a subdivision of the scene, in order to be able to estimate quickly the possible interpenetration of the deformable bodies, hence collision [8]. This method also allows a different time rate between haptic rendering and environment simulation.

Other techniques exist in order to reduce the computation time caused by collision detection. Barbic reduces the object's complexity with a model reduction, keeping only the necessary information, and thus allowing a simulation with a restricted number of degree-of-freedom for the object [2]. The object being really simpler, it requires less computation for collision. For multi-collision, he also proposes a method for rendering only a part of the force if more computation is needed, and with a sufficient quality.

Lin et al. propose a multi-resolution hierarchy in order to compute the collision [22]. They have different levels of details, in order to know which part of the object is to be calculated more precisely because of probable collision. It allows to keep a good quality while having better performance.

To conclude, much work has been done for collision detection, and for interaction with deformable objects. But, to the best of our knowledge, no method exists for the torsion of deformable objects.

Chapter 3

Overview of our work

In the previous chapter, we have summarize the State-of-the-Art concerning first the physically-based simulation of deformable objects for torsion, and then haptic rendering. We stated that FEM-based models implementing the corotational method have really good results for the simulation of deformable objects, especially for large displacements. However, the modeling of object torsion has never been addressed for 3D virtual objects. We also stated that there exist no haptic feedback techniques for torsion. The closer existing techniques for haptic rendering concern deformable objects.

In the upcoming chapter, we present the main contributions of this master thesis, as well as the framework we have designed for the haptic rendering of 3D virtual object torsion. The contributions are twofold. The first contribution is a **novel physically-based model to simulate the hand grabbing and deforming the objects**. The second contribution concerns a **novel coupling scheme for haptic rendering of object torsion**. These two contributions are detailed respectively in chapter 4 for the torsion model and chapter 5 for the novel haptic coupling scheme.

3.1 General pipeline of our work

Our global pipeline is summarized in Figure 3.1. It consists of a haptic device manipulated by the user. The user can then interaction with the 3D virtual environment composed of physically-based deformable objects. During this master, we used a 6 DoF haptic device , the Virtuose (Haption, Soulgé sur Ouette, France). We used SOFA as framework for the physically-based simulation [1]. The main properties of the framework are detailed in the next paragraph.

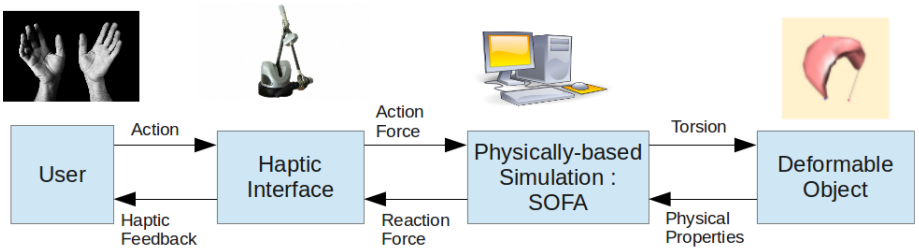


Figure 3.1: The global pipeline of our interaction. The user applies a force on the haptic interface. This force is then sent to the physically-based simulation, that applies a torsion, or any other deformation on the deformable object. The deformable object has internal forces, which are then treated by the simulation. The simulation then sends a reaction force to the haptic interface, that provides a haptic feedback to the user.

3.2 The framework for the physically-based simulation

Our contributions were performed within a framework called SOFA [1]. SOFA, Simulation Open Framework Architecture, is a framework dedicated to physically-based simulation. It contains most of the known representation of the objects, with most of the corresponding methods, giving the user a large choice for simulation. It relies on a tree-based representation of a scene, with at the root the common aspects of the scene (gravity, general information about the object). Each branch contains specific elements. For example, a branch with an object can contain its mesh loader, its physical representation, its graphical representation, and might also contain a constraint for some of the mesh nodes, as shown in Figure 3.2. The corresponding simulation interface is shown in Figure 3.2.

This tree-based representation can be converted into a XML version, describing all the scene. The simulation itself is performed using the SOFA Modeler, in which several parameters can be chosen. First of all, the integration timestep can be modified at any moment of the simulation. All parameters from the objects from the scene tree can also be changed during the simulation. Some representations of objects also allow direct interaction through the use of the mouse. The user can hence move any node of the mesh, provided that the node is not constrained, in which case the constraint will be applied on priority.
From a software point-of-view, all the elements in the scene are implementations of global elements, for example a Corotational FEM Forcefield and Mass-Spring force field will be two different implementations of a global

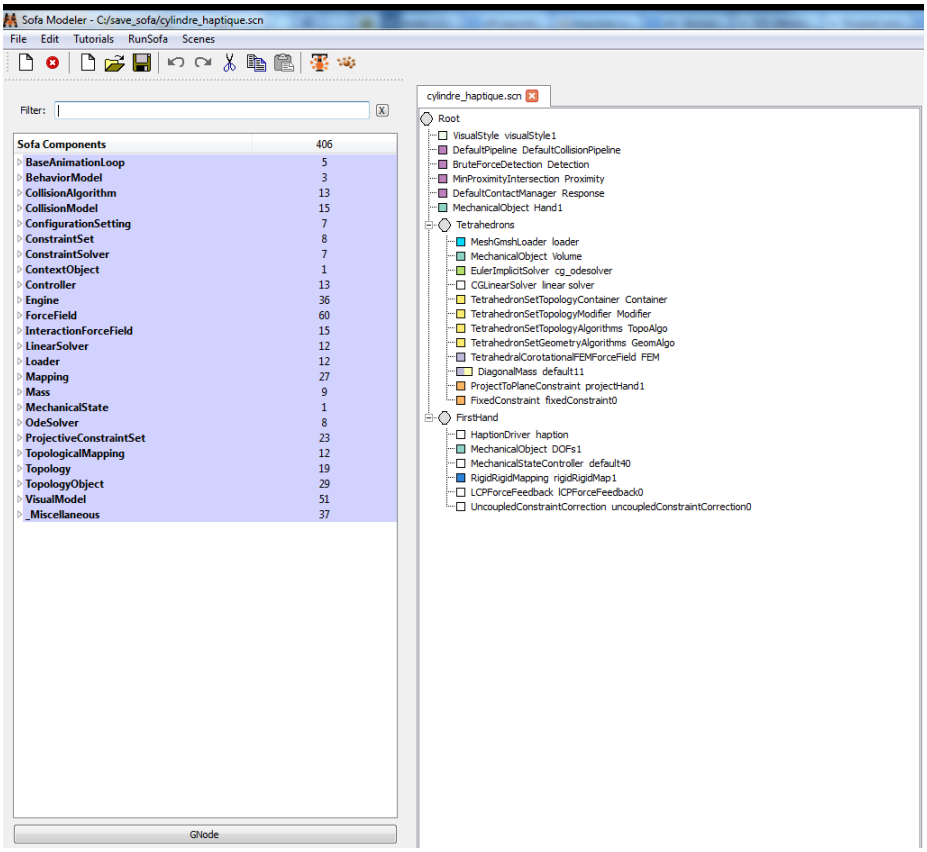


Figure 3.2: Example of a scene tree using SOFA: the deformation of a cylinder is performed with the corotational FEM method

Forcefield element. These global elements generally do not depend on the implementation of the others elements, which means that it is quite easy to make a specific implementation of an element, since it does not require the modification of the others elements. For instance, for our interaction, we have decided to implement it as a constraint, which does not affect the representation of the object. This constraint only needs the inherited classes from the general constraint element.
SOFA also includes drivers for several haptic devices, including the ones presented in Figure 2.5. They handle the displacement of the haptic device. If the haptic device is linked with an object, a Mapping is required in order to link the informations from the haptic device and from the mesh. Another element is required if force feedback is needed. This element only takes into account the collision between objects. All the code is implemented in C++, for better performances.

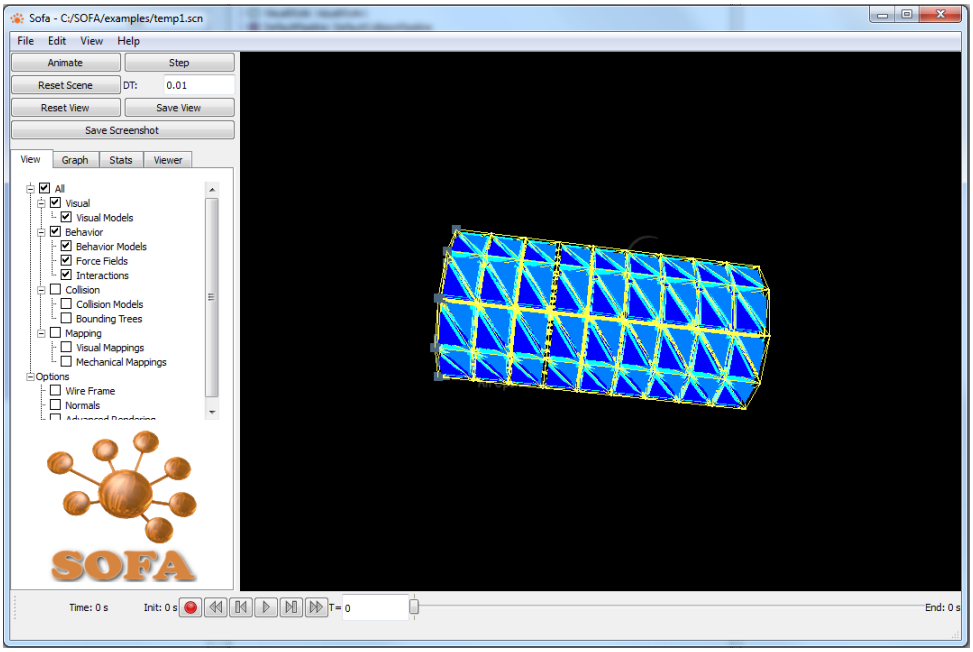


Figure 3.3: The corresponding simulation interface

Chapter 4

Physically-based Simulation of Torsion

In this chapter, we describe our first contribution on the physically-based simulation of 3D virtual object torsion. This contribution consists in the formalization of a novel physically-based constraint for modeling the torsion of an object through a hand. Section 4.1 details our hypotheses for the simulation of 3D deformable objects. Section 4.2 introduces our novel model for torsion while Section 4.3 illustrates the implementation within our framework.

4.1 Physically-based Simulation of 3D Deformable Objects

Hypotheses

In order to simulate the torsion of a deformable object, we need first to chose a representation for this object. Since we need a good realism while having small computation time, we chose the Finite Element Method (FEM), implementing the corotational method [19]. This method allows to have great deformation of our object, which is important, since the resolution of the equation system solving the deformation of the object is a highly non-linear problem. As it solves the equations from the continuum mechanics equations, it allows to simulate the deformation in a realistic way. The corotational method also allows the resolution of these equations in interactive real time. This last property is required for the haptic rendering of the 3D object deformations.

Corresponding Equations

As presented in the Related Work, the corotational FEM method relies on the extraction of a rotation for each tetrahedron, in order to extract a linear Cauchy tensor. Considering the Equation 2.4 applied to a single tetrahedron, we have

$$\sigma = E(\mathbf{x} - \mathbf{x}_0) \tag{4.1}$$

$E \in \mathcal{R}^{12*12}$being the element's stiffness matrix, and $\mathbf{x} \in \mathcal{R}^{12}$ the coordinates of the 4 points in the element. With $R_{\mathbf{X}}$ the rotation part of the rigid body transformation, let $R_e \in \mathcal{R}^{12*12}$ be the matrix containing four times R_x on its diagonal, and zeros everywhere else. $R_e^{-1}\mathbf{x}$ is the tetrahedron rotated back to the orientation of $\mathbf{x}_0$, and allows a to calculate the strain. A new calculus of

$$\sigma = R_e E(R_e^{-1}\mathbf{x} - \mathbf{x}_0) \tag{4.2}$$

gives the calculus of the strain in the tetrahedron as if $R_{\mathbf{X}}$ did not exist, but then rotated back to its proper orientation with R_e. R_x can be obtained by the polar decomposition of the $\nabla\mathbf{u}$ matrix. This can be further extrapolated to the entire mesh.

4.2 Model Description

Motivation : the Hand Seen as a Plane

In order to interact through a haptic device, we need to define the potential interaction. In real world, people would use their two hands to take the object. There are very few haptic devices in the literature able to render haptic feedack in the fingers directly. For this reason, our objective is to use a more classical 6DoF haptic device. For this type of device, we can not exactly reproduce the action of the hand with haptic feedback. The simulation of the hand itself with its precise interaction with the object requires too much computation, and the corresponding haptic rendering would be too complex if there could be any device allowing feedback on all the palm of the hand. We have therefore to simulate otherwise the behaviour of the hand in the virtual environment.

In the State-of-the-Art, most of the interaction rely on a direct coupling with an object, with collision detection, or with a single node of the mesh. The simulation with a single node would not allow the interaction for the torsion phenomenon. It could deform the object, and even make a small rotation of a part of the object, but it is impossible to make a good torsion of our object with only a node, and it is not realistic to consider that the action of the hand on an object can be simulated through the interaction with a single node of the mesh. In our case, we also do not want to have one hand directly coupled to an object. The object to be deformed would not

be deformed if coupled to the haptic device, and an external object would only apply external forces, and not the required torque for torsion.

Another property to notice is that the hand acts on the surface of the object: if we want to simulate an interaction, we need to make it on surface elements of the object. In order to have a simple, model for interaction, but still complex enough to have a good interaction with the object, we decided to consider that the hand acts on a plane, as illustrated in Figure 4.2. The interaction between the hand and the object will therefore be on surface elements of an object in a determined plane.

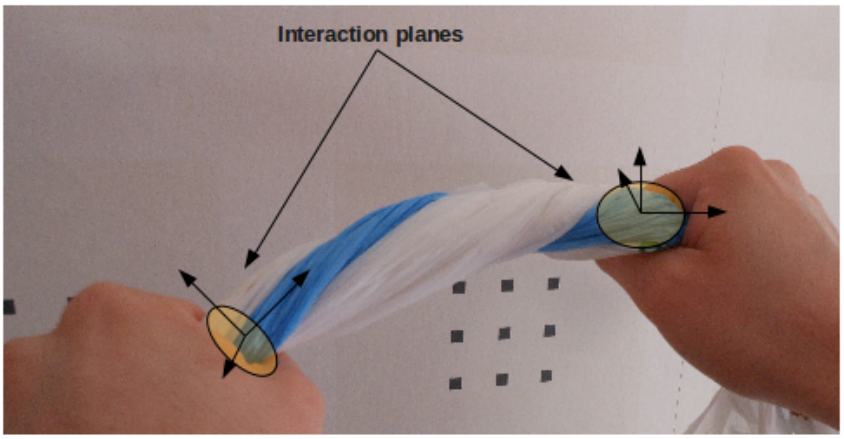


Figure 4.1: The two interaction planes representing the action of the hand on the object.

The Interaction Frame

As our haptic devices allow 6 Degrees-of-Freedom, we have to make an interaction relying on a position (3 DoF in translation) and an orientation (3 DoF in rotation). These virtual position and orientation of the haptic device in the virtual environment define the plane in which we are performing our interaction, and what point is the center of it. We need the information about the point, as every rotation needs a center, and the virtual position of the haptic device is this center. We hence call it the center of interaction in the following. These center and orientation define a local frame for the plane of interaction. More formally, our frame is defined with the following notation:

$$(\mathcal{O}(t), \mathbf{i}(t), \mathbf{j}(t), \mathbf{k}(t)) \tag{4.3}$$

with $\mathcal{O}(t)$ the coordinate of the rotation center, $\mathbf{i}(t)$ and $\mathbf{j}(t)$ two orthogonal normed vectors in the plane, and $\mathbf{k}(t)$ the normed normal vector of the plane, at the instant t of the simulation. We note $R(t) = (\mathbf{i}(t), \mathbf{j}(t), \mathbf{k}(t))$ the rotation matrix formed by the three base vectors.

The Action on the Object Mesh : a Constraint on Surface Nodes

As the hand performs an action on the surface of the object, we make the interaction only on the surface FEM elements of our object, ie the surface nodes of the mesh of our object. We consider that there is no friction inside the hand, which means that the nodes have fixed coordinates in the local frame of the interaction plane. Therefore, a constraint will be applied on our nodes. There are two methods for deforming an object. It is possible to apply forces on it, and solve the resulting equations, and it is also possible to create a constraint on certain nodes. These nodes will have defined coordinates, and the equations of continuum mechanics are solved on the other nodes taking into account the position of the constrained nodes. For our interaction, we chose to create a constraint on the nodes, since our torsion model needs to be applied on specific node positions. With this constraint, we also ensure that the node will have fixed coordinates in the local frame. A representation of constrained nodes is shown in Figure 4.2. For each node l in our object, the coordinates $p_l(t)$ in the global frame are hence :

$$p_l(t) = R(t) * \hat{p_l} + \mathcal{O}(t) \tag{4.4}$$

$$\hat{p_l} = R^{-1}(0)(p_l(0) - \mathcal{O}(0)) \tag{4.5}$$

$\hat{p_l}$ is the invariant local coordinates of the node, determined at the initialization of the interaction. Any movement of the haptic device moves the plane, and thus all the constrained nodes. This approach constitutes a novel interaction model, with a novel simulation of the hand, requiring only 6 parameters. It allows to have two-handed interaction with the object, as well as only one-handed interaction with part of the object fixed in the scene. The simplicity of the model also allows to interact with bigger meshes, since it does not require a lot of computation time.

The Two Interaction Phases

There are two steps in order to define the constraint. The first step corresponds to the initial placement of the frame, allowing the user to choose around which node he wants to interact, as shown in Algorithm 1. When the center and orientation are chosen, all the nodes inside a surface triangle intersecting the interaction plane are considered an interacting nodes, and thus constrained in the local frame. We also consider that the hand must surround the constrained nodes, and thus the constrained nodes will form a connected part of the surface. In order to achieve this, we perform a ray tracing algorithm, tracing one ray leaving from $\mathcal{O}(t)$ with direction $\mathbf{i}(t)$. It gives us a triangle from the surface mesh. We then compute recursively the neighbours that are in the plane. This ensures that the obtained triangles are in the plane, and that they represent a connected part of the surface

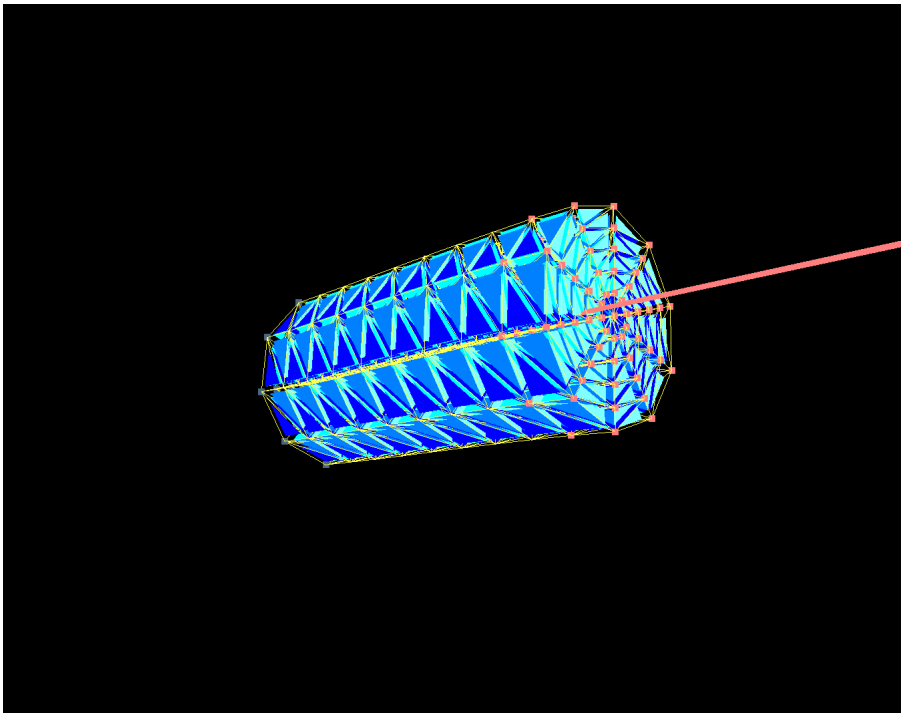


Figure 4.2: The orange nodes are constrained. The virtual frame of interaction is represented by a node (inside the object), and the orange normal.

mesh. The Initialize method must be called once for each hand.

In the second phase, the interaction planes moves, and hence all the constrained nodes, as seen in Algorithm 2. The Update function in Algorithm 2 is called at each timestep. It uses the local coordinates initialized with the Initialize method, constraining the nodes to their initial local coordinates. $MeshTriangles$ contains all the surface triangles from the mesh of the object. The simulation of the non-constrained nodes is achieved by forces. We assumed that it would be enough for the simulation of the object. Future work could implement non-linear FEM.

Algorithm 1: Initialize(Center $\mathcal{O}$, Orientation R)

```
Array<Triangle> TriangleArray ;          // the list of all
triangles intersecting the plane
for all TriangleT ∈ MeshTriangles do
  if Intersects(T, (R.column(0), R.column(1))) then
    TriangleArray.push_back(T) ; // the triangle intersects the
    interaction plane
  end if
end for
Triangle T = RayTracing(R.column(0)) ; // perform Raytracing in
TriangleArray for better performances
Array<Triangle> TriangleBuffer ;         // the list of all
triangles treated for connexity
Array<Triangle> TriangleConnected ;  // the list of all the
connected triangles
TriangleBuffer.push_back(T);
while TriangleBuffer.size()! = 0 do
  Array<Triangle> Neighbours =
  TriangleBuffer.pop().Neighbours;
  for all N ∈ Neighbours do
    if !N.isTreated() then
      TriangleConnected.push_back(N);
    end if
  end for
end while
Array<Points> InteractionNodes
for all Tr ∈ TriangleConnected do
  if !Tr.Vertex1 ∈ InteractionNode then
    InteractionNodes.push_back(tr.Vertex1);
  end if
  if !Tr.Vertex2 ∈ InteractionNode then
    InteractionNodes.push_back(tr.Vertex2);
  end if
  if !Tr.Vertex3 ∈ InteractionNode then
    InteractionNodes.push_back(tr.Vertex3);
  end if
end for
InteractionNodes.sort();
for all l ∈ InteractionNodes do
  LocalCoordinates[i] = R^{-1}(Coordinates[l] − O);
end for
```

Algorithm 2: Update (Center $\mathcal{O}$, Orientation R)

for all $l \in InteractionNodes$ **do**
  $Coordinates[l] = R * LocalCoordinates[l] + \mathcal{O}$;
end for

4.3 Implementation

The constraint has been implemented in SOFA as a projection. The nodes are indeed projected to their initial position in the local frame of interaction. The projections in SOFA have several methods of projection, including for position and velocity. The recording of the constrained position is performed during the initialization method, and each time that the ProjectPosition method is called, we get the center and the orientation of the haptic device. We then calculate the global position from this frame, and project the position.

Chapter 5

The Haptic Interaction

In this chapter, we describe the second contribution on the haptic rendering for the torsion of deformable objects. This contribution consists in a haptic rendering based on the difference between the deformed mesh and the corresponding rest state of the object. Section 5.1 details the motivations for such a haptic rendering. Section 5.2 describes the two phases of haptic interaction with the corresponding algorithms. Section 5.3 explains the coupling scheme for this rendering, while Section 5.4 describes the implementation within SOFA.

5.1 Motivations

Our goal is to simulate the reaction of the deformed object on the hands that deformed it. This reaction is provoked by internal forces leading the object to deform back to the rest state. These internal forces are induced by the elastic deformation of the object, ie the deformation of the object without considering global translation or rotation in the scene. We do not consider plastic deformation in our model. We need to have really fast methods for haptic rendering, since the timestep for haptic rendering is 30 times smaller than the timestep for visual rendering. We thus need to create a rendering which is not exactly the one that could be calculated with all the constrained nodes. We therefore decided to have a rendering depending on the global position and orientation of the plane, without considering the local topology of the mesh.

For our rendering, we need to create a method relying only on the elastic deformation of the object, and only on the position and orientation of the interaction center. The elastic deformation applied to the interaction frames is given by the difference between the interaction frames in the deformed object and the interaction frames in the corresponding rest state of the deformable body, as shown in Figure 5.1.

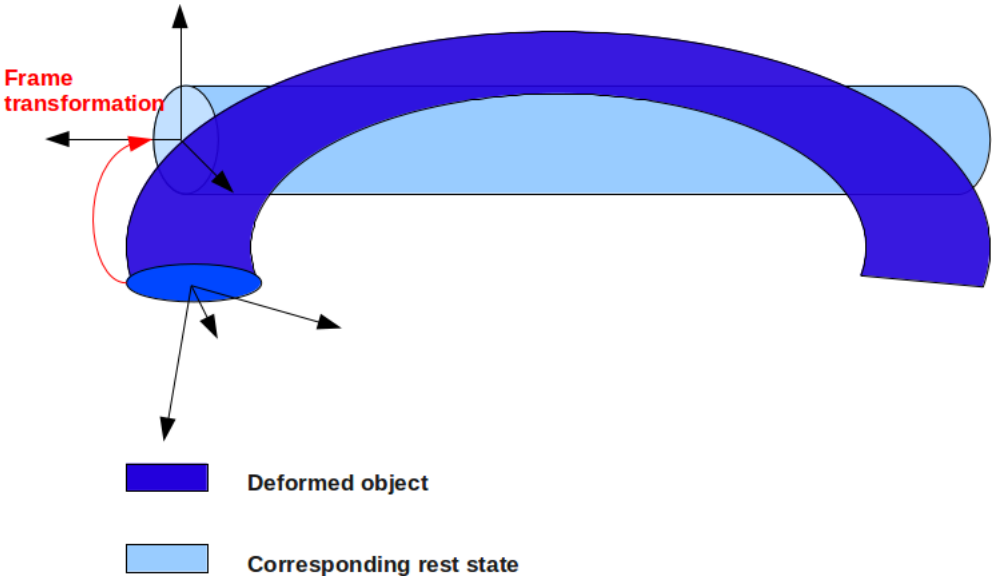


Figure 5.1: The difference with the rest state of the object gives the direction and torque for force feedback

This difference gives us a translation and a rotation between the two frames. Our hypothesis is that the force and torque in the haptic feedback have to be proportional to this translation and rotation. Part of this coefficient will be the rigidity coefficient. The beam theory states, that the torque T of the torsion force for an angle per length unit of θ, with a rigidity coefficient (the Young' modulus) $\mathcal{E}$ is $T = \theta * \mathcal{E}$.
This means that the force required for the torsion of the beam, the torque T, which is also the opposite of the reaction force, only depends on the rigidity $\mathcal{E}$ of the object and on the angle per length unit θ of the rotation.

5.2 The two interaction phases

In our model, there is no object coupled to our interface, contrary to most applications in the related work. Our device is coupled to a virtual frame, and follows the two interaction phases seen in the previous chapter. In the first phase, the user moves a virtual frame, without constraining any node in the object. He can chose at any time, by pushing a button on the haptic device, to begin the interaction. At this moment, the Initialize function is called, with the position and orientation of the virtual frame as parameters, as seen in Algorithm 3. It is interesting to note that for SOFA, a haptic device must be linked with an object. In our case, we need to create an object without mesh, hence a virtual object, which only contains the position and

orientation of the haptic device. This position and orientation are accessible from the constraint.

Algorithm 3: BeginInteraction

$Point\ Center = GetRestPosition();$
$Point\ HapticCenter = this.position;$
$Matrix\ Orientation = GetRestOrientation();$
$Matrix\ HapticOrientation = this.orientation;$
$Point\ LocalRestPosition = HapticCenter - Center;$
$Matrix\ LocalRestOrientation = HapticOrientation * Orientation^{-1};$
$Initialize(HapticCenter, HapticOrientation);$

During each timestep, the Update function is called with the current position and orientation of the virtual frame, constraining the nodes in the interaction frame. Haptic rendering is performed during this second phase, as seen in Algorithm 4.

Algorithm 4: HapticRendering

$Point\ Center = GetRestPosition();$
$Point\ HapticCenter = this.position;$
$Matrix\ Orientation = GetRestOrientation();$
$Matrix\ HapticOrientation = this.orientation;$
$Real\ YoungModulus = GetYoungModulus();$
$Real\ DampingCoefficient = GetDampingCoefficient()$;
`// internal parameter, will be independent from the objects`
$TranslationRendering = YoungModulus * DampingCoefficient *$
$(HapticCenter - Center - LocalRestPosition);$
$TorqueRendering = YoungModulus * DampingCoefficient *$
$(HapticOrientation * Orientation^{-1} * LocalRestOrientation^{-1});$

5.3 Our Novel Coupling Scheme

We stated that for our coupling scheme, we would rather use an impedance rendering, ie calculation over the position of objects in the scene, and sending forces as feedback to the user. Here the position that we calculate are the position and orientation of the interaction frame, which is coupled to a constraint on the object mesh. With these position and orientation, we calculate the internal forces that should apply on the constrained nodes, using the Young's Modulus as a coefficient for rigidity. This coupling scheme, depicted on Figure 5.3, along with the rendering, constitute one of the contributions of this master thesis.

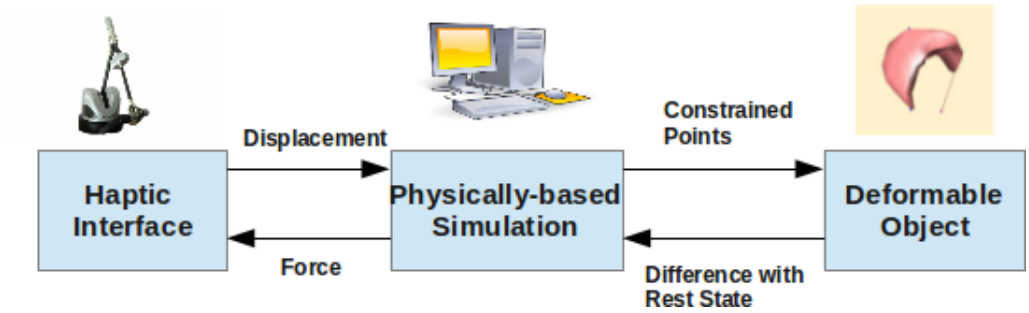


Figure 5.2: The coupling scheme for our haptic feedback

It is really important to notice that the haptic devices will be coupled to virtual frames, and not to actual objects. During the interaction phase, the haptic devices will constrain nodes in the same object, and the resulting interaction is a major goal of our internship. The novelty of our haptic rendering also relies on the fact that it not a collision-based rendering. As far as we know, it is the only haptic rendering based on internal forces for deformable objects. It allows to interact with the coupled object, and not only with the other objects through the collision with the coupled object. This method is thus a great improvement for the interaction with deformable objects.

5.4 Implementation in Our Framework

The haptic rendering has been implemented in the haption driver for the Virtuose. The driver contains a pointer to the Constraint element. During the first phase, the force feedback is deactivated, in order to enable the free displacement in the environment to chose the interaction plane. The Virtuose has two black buttons on it, simulating the two buttons of the mouse. When the first button is pressed, the Initialize method from the constraint is called, and the force feedback is activated. When the second button is pressed, the constraint is cleared, and the force feedback deactivated, allowing the user to be able to chose several different planes of interaction.

Chapter 6

Results

We managed to implement the constraint, as well as the corresponding haptic rendering. For now it only involves one haptic device. It means that a part of the nodes in the object are fixed, in order to be able to have an interaction with it. We will first show some results about the simulation of torsion, first with an evaluation of the computation time performance of the simulation on several meshes, and then with a comparison between the torsion of a cylinder and the theoretical solution, given by the theory of beams. We will later present the results of the haptic rendering.

6.1 The Physically-based Simulation

6.1.1 Computation Time Performance

The deformation has been performed on several meshes. We compare for each simulation the number of Frame-per-Second (FPS) given by the SOFA interface of simulation. The integration timestep (see Related Work) was 0.01 second. The first mesh that we deformed was a simple cylinder, composed of 2430 tetrahedra. The result can be seen in Figure 6.1.1 This interaction has been performed with 60 FPS, which is much more than the required 30 FPS for visual feedback. Despite being performed on a quite inaccurate mesh, the deformation is visually plausible.
The second mesh on which we performed the interaction is a rather bigger cylinder, composed of 8332 tetrahedra. The deformation can be seen in Figure 6.1.1. With this mesh, the FPS is only 17, which is just insufficient for our use. The constrained points however behaved surprisingly good, without any noticeable lag, which is a rather good result.

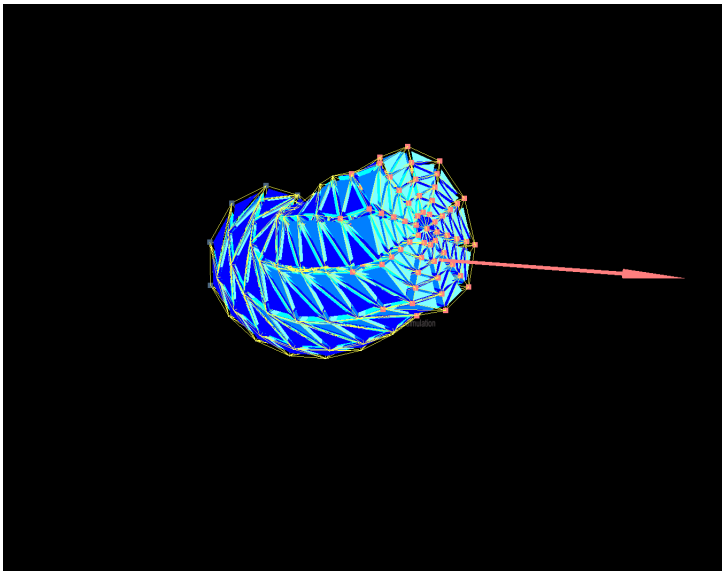


Figure 6.1: A simple cylinder deformed with a rotation of its external face

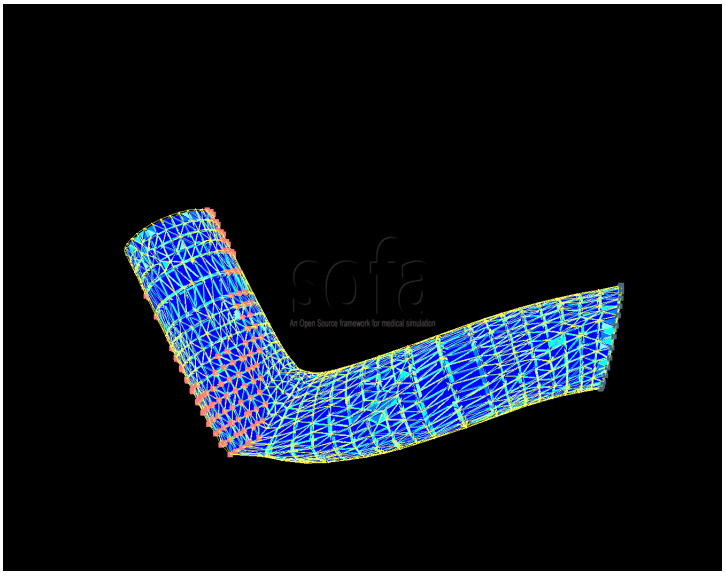


Figure 6.2: A bigger cylinder deformed with a non axis-aligned plane

In order to know how the meshes behave with a size between the two previous ones, and in order to know what the maximum size is for interaction, we deformed a dragon mesh, composed of 4323 tetrahedra. This also shows how more complex object behave with our interaction. The result can be seen in Figure 6.1.1.

The FPS for this simulation is 30, which is our limit for interaction.

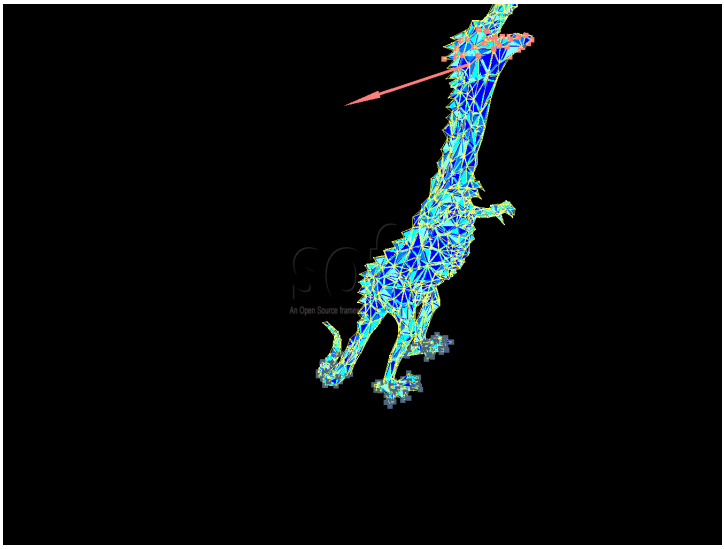


Figure 6.3: A dragon mesh with fixed feet. The interaction plane is located in the neck

The computation time (the inverse of the FPS) seems to be rather linear in the number of tetrahedra, which would be a reasonable hypothesis. To confirm it, we performed the deformation on a really important mesh, a cylinder composed of 31558 tetrahedra. The FPS was 4.3, which confirms the linear hypothesis, as shown in Figure 6.1.1. It also strongly affected

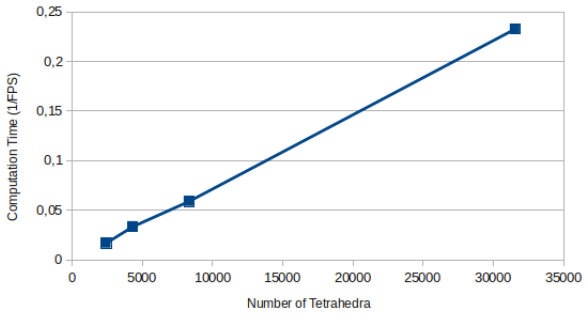


Figure 6.4: The computation time as a function of the number of tetrahedra

the haptic rendering. Table 6.1 sums up the different FPS for the studied meshes.

Moreover, we also noticed that the tetrahedra composed of both constrained and non constrained nodes are deformed too much during fast movements of the haptic device. It needs then a few milliseconds to sta-

Mesh	Tetrahedra	FPS
Small Cylinder	2430	60
Dragon	4323	30
Medium Cylinder	8332	17
Large Cylinder	31558	4.3

Table 6.1: Comparison of the FPS during the deformation of different meshes

bilize to a more normal configuration.

6.1.2 Comparison with Theoretical Torsion for a Beam

In order to validate our deformation, we need to compare our deformation with the theoretical deformation of an object. A simple model to study for the torsion is the beam model. As explained in the Related Work, a beam is an object for which two dimensions are negligible compared to the third one. The object we used for the study is the middle cylinder from the previous section. We will approximate it as a beam, with a total length of 10, compared to a radius of 1. The St-Venant hypothesis states that, for a torsion along the z-axis, for a total rotation of angle θ_{tot} on a beam of total length $\mathbf{z}_{tot}$, the action on a plane $z = a$, $0 \leq a \leq \mathbf{z}_{tot}$ is a rotation, of angle $\theta_a = \theta_{tot} * \frac{a}{\mathbf{z}_tot}$. We deformed the cylinder with the two methods, the explicit rotation of every node in the mesh for the analytical solution, and a rotation on the two opposite planes with our constraint, with an angular speed of 0.1 rad per second. For our comparison, $\mathbf{z}_{tot} = 10$, and $\theta_{tot} = 2\Pi$. Figure 6.1.2 shows on top the analytical deformation, and on the bottom the object deformed using the corotational FEM. We can observe that the global deformation is very similar, but the tetrahedra with both constrained and non-constrained nodes undergo some large deformation, which is what had been noticed in the previous results.

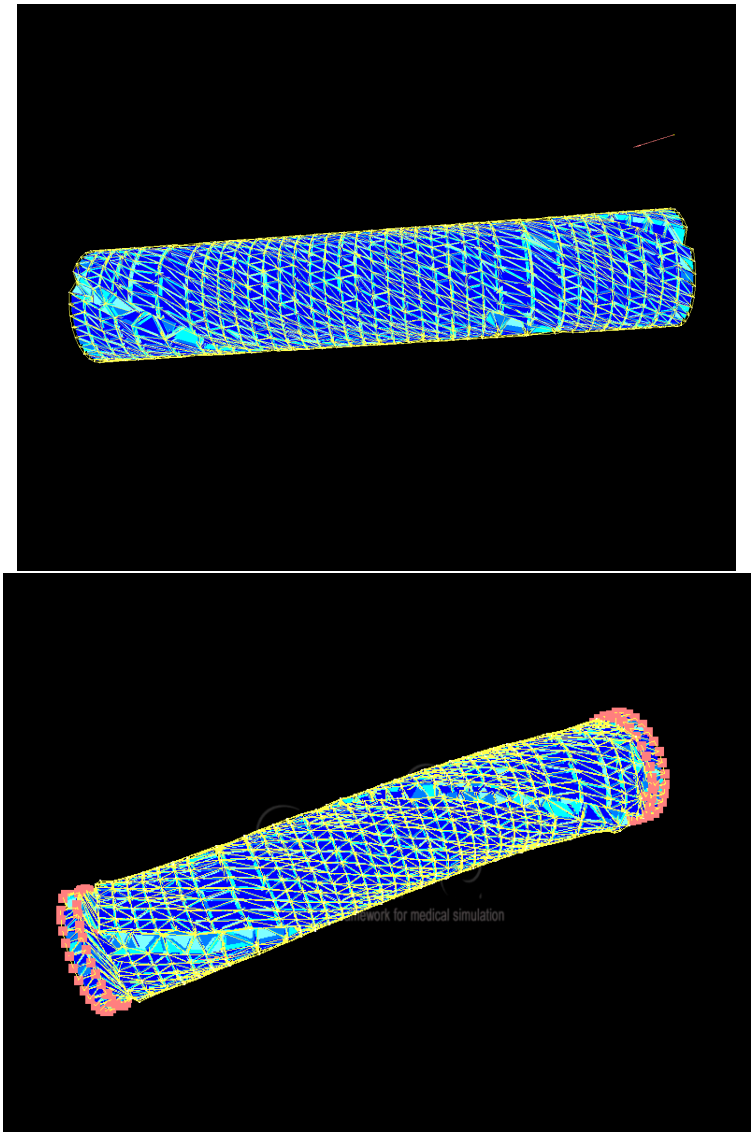


Figure 6.5: The two deformed cylinders. On the top, the deformation is obtained through analytical solution, and on the bottom, the deformation is performed by the physically-based simulation.

6.2 Haptic Rendering

Haptic rendering provides good results. Figure 6.2 shows the direct interaction with a deformable object, with coherent haptic feedback. For most of the simulations performed, the haptic rendering was performed in real time, the bigger cylinder being the only exception, inducing a severe lag and a drastic decline of performances. Figure 6.2 shows the force feedback

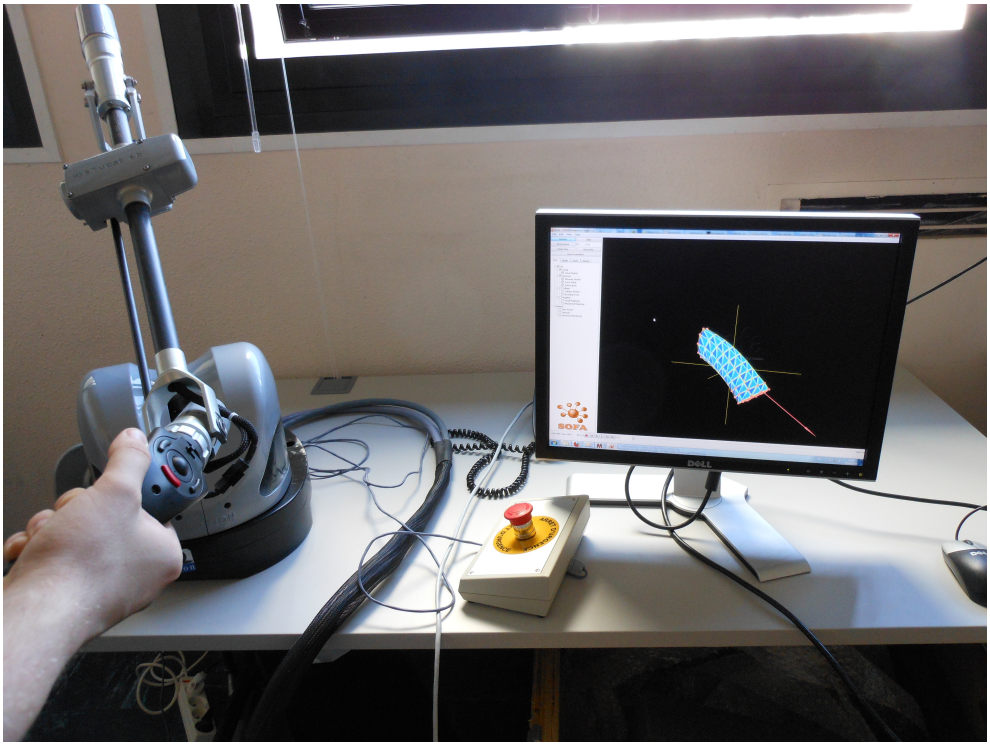


Figure 6.6: Interaction through a haptic device

for every dimension, in translation and in rotation obtained on a deformation of the cylinder. It shows independent actions on each force feedback for each direction. The force feedback in rotation has significantly smaller values than the translation. This is a normal phenomenon, considering that any value close to 1 in rotation force feedback leads to important instability. The rotation force feedback is nevertheless strongly noticeable.
The much higher values in translation give a good feedback on the force induced by the elongation or compression of the mesh.
As expected, the force feedback opposes the displacement of the haptic device, as a normal reaction of the objects to get back to its rest state.

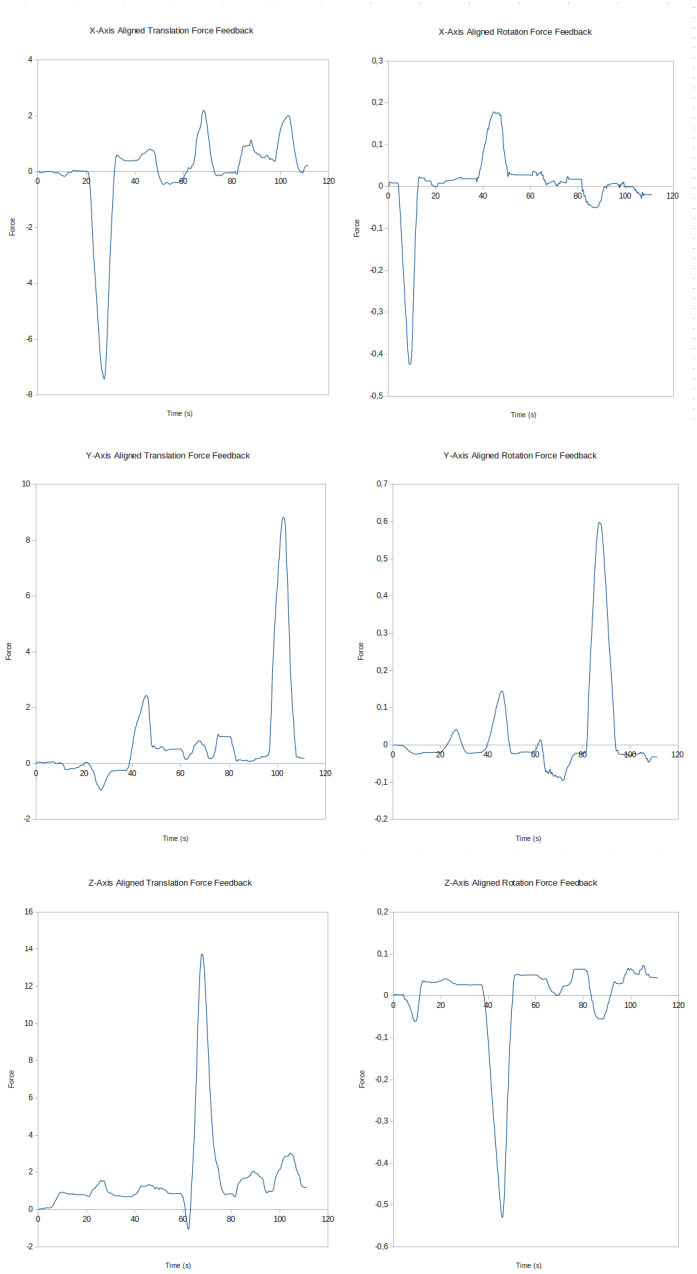


Figure 6.7: Force feedback for each dimension, both in translation and in rotation. For $0 \leq t \leq 20$ we have a rotation around the x-axis, followed by a translation along the x-axis for $20 \leq t \leq 40$. Then for $40 \leq t \leq 60$ we have a translation along the z-axis, followed by a rotation around the z-axis for $60 \leq t \leq 80$. We finally have for $80 \leq t \leq 100$ we have a rotation around the y-axis, followed by a translation along the y-axis for $100 \leq t \leq 120$.

Chapter 7

Conclusion and Perspectives

The torsion is a major element concerning the simulation of deformable objects. We wanted an interaction capable of simulating the torsion, and to provide haptic feedback. In this report we have presented the Related Work in the domains of physically-based simulation of deformable objects, the theoretical grounds for torsion, and haptic feedback for deformable objects. We have then presented the two contributions of this master thesis, namely a novel interaction model for deformable objects simulating the action of the hand on the object, and a novel haptic rendering for this interaction, providing force feedback for the internal forces opposing the torsion movement. For smaller meshes, we have obtained good results for the simulation of arbitrary deformations of objects, including torsion, elongation and compression, and flexion. We have shown that the simulation is too slow for bigger meshes, but the haptic rendering does not suffer from this problem. For the rest of the internship, the perspectives are manifolds.
We first want to integrate a second haptic device, for a better interaction with the deformable object. We also intend to fix the problem of adjacent elements during fast displacement of the constraint. Part of the Future Work would be to try other representations of meshes in order to perform a better simulation. Many ideas for this were evoked in the Related Work, with 1D models and multi-resolution models. It would also be interesting to study the interaction in a more complex environment, and more precisely interaction with other objects while performing the deformation of the object. Finally, a user study could validate the haptic rendering as providing good sensations on deformation of objects.
Possible applications for our model could include surgery simulation tools, with better deformation and haptic feedback of organs, or for virtual tools for learning, with a better haptic interaction with deformable objects.

Chapter 8

Acknowledgement

I would like first to thank Maud, for her constant help, her precious advices and for bringing back motivation when it was needed. I would like then to thank Anthony, for his inestimable technical support with my first steps in SOFA. I would finally like to thank all the other interns from the office, for their help, and for the motivational mood they brought.

Bibliography

[1] Jérémie Allard, Stéphane Cotin, François Faure, Pierre-Jean Bensoussan, François Poyer, Christian Duriez, Hervé Delingette, and Laurent Grisoni. Sofa an open source framework for medical simulation. In *Medicine Meets Virtual Reality (MMVR'15)*, Long Beach, USA, February 2007.

[2] J. Barbic. *Real-time Reduced Large-Deformation Models and Distributed Contact for Computer Graphics and Haptics*. PhD thesis, Carnegie Mellon University, 2007.

[3] F. Bertails. Linear time super-helices. In *Proceedings of Computer graphics forum*, volume 28, pages 417–426, 2009.

[4] D. Calecki. *Physique des milieux continus 2 : Traction, Torsion et Flexion*. Hermann, 2007.

[5] S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popovic. Interactive skeleton-driven dynamic deformations. In *Proceedings of ACM Transactions on Graphics (TOG)*, volume 21, pages 586–593, 2002.

[6] Y. Chen, Q. Zhu, A. Kaufman, and S. Muraki. Physically-based animation of volumetric objects. In *Proceedings Computer Animation 98*, pages 154–160, 1998.

[7] A. Curnier. *Mécanique des milieux déformables*. Presses polytechniques et Universitaires Normandes, 2005.

[8] J. Davanne, P. Meseure, and C. Chaillou. Stable haptic interaction in a dynamic virtual environment. In *Proceedings of Intelligent Robots and Systems, 2002.*, volume 3, pages 2881–2886, 2002.

[9] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic real-time deformations using space & time adaptative sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 31–36, 2001.

[10] O. Dmitrochenko. Finite elements using absolute nodal coordinates for large-deformation flexible multibody dynamics. *Journal Of Computational and Appplied Mathematics*, 215:368–377, 2008.

[11] P. Fuchs, G. Moreau, A. Berthoz, and J.-L. Vercher. *Le traité de la réalité virtuelle volume 1 - L'Homme et l'environnement virtuel*. École des Mines de Paris, 2006.

[12] J. Garrigues. *Fondements de la mécanique des mileux continus*. Lavoisier, 2007.

[13] M. Hauth and W. Strasser. *Corotational Simulation of Deformable Solids*. Citeseer, 2003.

[14] P. Hunter and A. Pullan. Fem/bem notes. *Department of Engineering Science. The University of Auckland, New Zealand*, 2001.

[15] K. Larsson, G. Wallgren, and M. G. Larson. Interactive simulation of a continuum mechanics based torsional thread. In *Proceedings of the 7th workshop on virtual reality interaction and physical simulation*, pages 49–58, 2010.

[16] M. C. Lin and M. A. Otaduy. *Haptic Rendering : Foundations, Algorithms, and Applications*. A K Peters, Ltd., 2008.

[17] C. Luciano, P. Banerjee, L. Florea, and G. Dawe. Design of the ImmersiveTouch: a high-performance haptic augmented virtual reality system. *Proceedings of Human Computer International*, 2005.

[18] C. A. Mendoza and C. Laugier. A solution for the difference rate sampling between haptic devices and deformable virtual objects. In *Proceedings of International Symposium on Robotics and Automation*, 2000.

[19] M. Müller and M. Gross. Interactive virtual materials. In *Proceedings of Graphics Interface 2004*, pages 239–246. Canadian Human-Computer Communications Society, 2004.

[20] A. Nealen, M. Muller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. In *Proceedings of Computer Graphics Forum*, volume 25, pages 809–836. Wiley Online Library, 2006.

[21] M. Nesme, Y. Payan, and F. Faure. Efficient, physically plausible finite elements. In *Proceedings of Eurographics*, 2005.

[22] M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. In *Proceedings of ACM SIGGRAPH 2005 Courses*, page 72, 2005.

[23] W.T. Reeves. Particle systems–a technique for modeling a class of fuzzy objects. In *Proceedings of ACM SIGGRAPH Computer Graphics*, volume 17, pages 359–375. ACM, 1983.

[24] A. L. Schwab and J. P. Meijaard. Comparison of three-dimensional flexible beam elements for dynamic analysis: Finite element method and absolute nodal coordinate formulation. In *Proceedings of IDETC/CIE*, pages 24–28, 2005.

[25] J. Spillmann and M. Harders. Inextensible elastic rods with torsional friction based on lagrange multipliers. *Computer Animation and Virtual Worlds*, 21(6):561–572, 2010.

[26] H. Sugiyama, J. Gerstmayr, and A. A. Shabana. Deformation modes in the finite element absolute nodal coordinate formulation. *Journal Of Sound and Vibration*, 298(4):1129–1149, 2006.