Generalized God-Objects: a Paradigm for Interacting with Physically-Based Virtual World

Philippe Meseure, Julien Lenoir, Sylvère Fonteneau, Christophe Chaillou

Generalized God-Objects: a Paradigm for Interacting with Physically-Based Virtual Worlds

Ph. Meseure
SIC, CNRS FRE 2731,
Univ. Poitiers, France
meseure@sic.univ-poitiers.fr
http://www.lifl.fr/∼meseure

J. Lenoir, S. Fonteneau, C. Chaillou
Alcove, INRIA Futurs, IRCICA-LIFL,
UMR 8022, Univ. Lille 1, France
lenoir@lifl.fr
http://www.lifl.fr/∼lenoir

Abstract

In this paper, we show a method to interact with physically-based environments in a way which guarantee their integrity whatever the mechanical properties of the virtual interaction tool and the control device. It consists in an extension of the god-object concept. The interaction tools are modeled as physical bodies which tend to reach, if possible, the position maintained by the user. Their behavior is computed via the dynamic laws of motion by the simulation engine, as the other bodies in the scene. The cases of articulated rigid bodies and deformable bodies are studied. This mechanism also provides a unified framework which allows the control of virtual objects via devices providing force feedback or not. Finally, some applications including virtual surgery are shown to illustrate the effectiveness of the approach.

Keywords: Physically-based Virtual Environments, Real-time Simulation, Computer Human Interface, Medical Simulation, Force Feedback

Introduction

With the increasing computing power of modern workstations, more and more applications resort to physical simulation to animate virtual worlds. Such applications include CAD, surgery simulation and games. However, the interaction of the user with a virtual environment still arises many issues. Indeed, the user usually controls a virtual tool (a virtual hand for instance) whose position is often imposed whatever the geometry of the virtual environment. In purely passive environments, it only leads to visually-annoying interpenetrations, which can be avoided by keeping the virtual tool in a former non-colliding position. In physically-based environments, the interpenetration can lead to undesirable behaviors of the simulated objects since the overlap is not physically valid. What is more, the lack of physical behavior of the tool sometimes makes a task hard to complete (see Fig. 5).

Such a problem can be addressed by using haptic devices, which allow the control of the position maintained by the user via a force feedback. However, this approach brings numerous issues such as the frequency adaptation of the simulation to the haptic devices... and, above all, the cost of such devices. Another approach to address the interaction problem is to use relative displacement devices such as a mouse. The movement of the mouse is translated to a 3D movement which is applied, if possible, to the virtual tool. The interesting property of relative-displacement devices is that their movement can be ignored, if necessary.

At first look, the use of non-haptic absolute-position devices should be avoided for physically-based environments. However, in some specific applications, the resort to such devices is inescapable. For instance, the simulation of a laparoscopy procedure requires the use of tools (forceps, graspers, etc.) whose

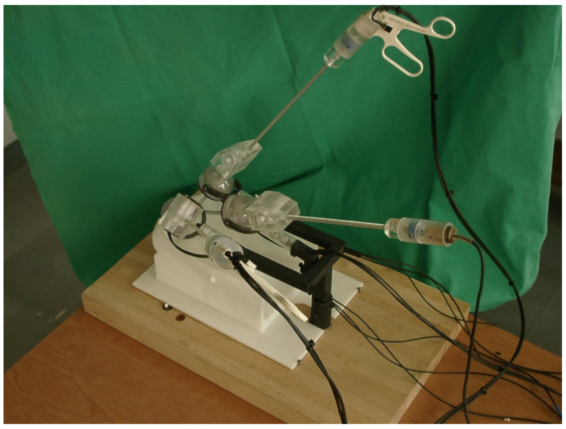

Figure 1: An non-haptic interface for laparoscopy simulation, built at the Medical Technologies Institute, Lille.

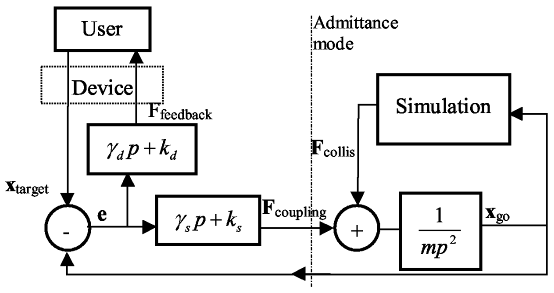

Figure 2: Use of a dynamic rigid god-object to interface the simulation and the device. Only the translation part is represented. The interface can eventually be haptic (in impedance mode).

absolute positions are measured and sent to the simulation workstation (see Figure 1). In this context, we also notice that the interaction tools cannot be considered as a purely rigid objects. Laparoscopy graspers obey the physics of articulated bodies. Endobags, endoscopes or certain sensing tools exhibit deformations. The need of a general method to handle complex interaction tools in a physically-based environment has therefore arisen.

In this paper, we show a device-independent and unified framework to guarantee consistent interactions with virtual worlds, no matter which type of devices and which type of interaction tools are used. This framework allows us to interact not only by the way of rigid bodies but also articulated or deformable structures. In practice, we are mostly concerned with the use of non-haptic devices. Nevertheless, along the paper, we will always pay attention to the compatibility issues of haptic devices.

Related work

Our solution relies on the *god-object* concept which has originally been imagined by Dworkin and Zelter [1]. They point out that, due to the time-discrete simulation and acquisition of the device state, an object handled by a user exhibits non-continuous and uncorrelated positions, so appears to move as if it was controlled by the "hand of God". So, they choose to consider this position as a target that a virtual object wants to reach but with a limited velocity.

This concept was later applied to the control of haptic devices by Zilles and Salisbury [2] and named as the *virtual proxy* method by Ruspini et al. [3]. The idea is to constrain the position of a virtual point (the *proxy*) which targets the position of the tip of a haptic device. The position of the proxy is computed to avoid interpenetration with the virtual environment. Moreover, the feedback force is computed by comparing the current positions of the proxy and the interface tip. The virtual proxy method has been extensively used in haptics application. Meyer et al. [4] showed the versatility of the god-object method for the design of an architecture dedicated to haptic devices.

We must point out that the proxy position is usually determined in a *static* way. That is, for a given position of the device, the closer position which is compatible with the environment obstacle is computed (by means of optimization methods) and applied as the proxy position. This method has been adapted by Mendoza et al. to allow the interaction with a deformable body [5]. A static approach can however lead to discontinuous positions over time. For more complex manipulated objects or interactions, specific methods must be taken to compute the proxy position. In a way similar to computer animation [6], dynamics is also a convenient approach to control the position of a colliding body. Therefore, McNeely et al. [7] use the laws of motion to compute the position of a rigid virtual proxy with a complex geometry. Zhuang and Canny [8] show that dynamics allows the use of finer collision models, which is necessary to interact with a deformable body. A dynamic approach is also used to handle articulated structures [9, 10], but in these work, the interface only control the tip of the articulated body. We therefore propose to generalize the dynamic approach and extend it to allow the control of any complex physical bodies. We consider that the device controls all or only a subset of the degrees of freedom of the virtual tools, and let the dynamics deal with the others.

Finally, let us mention that the god-object method enables the *virtual coupling* of the haptic device to the simulation [11]. The forces returned to the user are different from the simulation forces but are adapted to the device dynamics and mechanical properties. We also rely on this method to allow the use of non-haptic devices.

Dynamic god-object

In this section, we describe how the dynamic god-object method can fulfill our requirements in the case of a rigid body interaction tool. This dynamic approach can be presented as a feedback control (Fig. 2). So a second order control is used, which corresponds to the dynamic equation:

$$\begin{cases} m\frac{d^2\mathbf{x}_{go}}{dt^2} = \sum \mathbf{F} \\ \frac{d\mathcal{I}\mathbf{\Omega}_{go}}{dt} = \sum \mathbf{T} \\ \frac{d\mathbf{q}_{go}}{dt} = \frac{1}{2}\mathbf{\Omega}_{go}\mathbf{q}_{go} \end{cases} \tag{1}$$

where m is the mass and $\mathcal{I}$ the inertia matrix of the virtual tool, $\mathbf{x}_{go}$ the position of the god-object, $\mathbf{q}_{go}$ a quaternion representing its orientation, $\mathbf{\Omega}_{go}$ its rotational velocity. $\mathbf{F}$ and $\mathbf{T}$ represent all the external forces and torques applied to the god-object, including in particular an attraction toward the device position, defined with $\mathbf{x}_{target}$ and $\mathbf{q}_{target}$. This attraction is based on the following error measures $\mathbf{e}^{(t)}$ and $\mathbf{e}^{(r)}$:

$$\begin{cases} \mathbf{e}^{(t)} = \mathbf{x}_{target} - \mathbf{x}_{go} \\ \mathbf{e}^{(r)} = \mathbf{q}_{target}{\mathbf{q}_{go}}^{-1} \end{cases}$$

$\mathbf{e}^{(r)}$ represents a rotation whose angle is named α and axis $\mathbf{u}$. We want the god-object to follow the target position of the device, that is, we want to minimize the previously defined errors. A PD control is therefore applied [3]:

$$\begin{cases} \mathbf{F}_{coupling} = \gamma_s^{(t)}\frac{d\mathbf{e}^{(t)}}{dt} + k_s^{(t)}\mathbf{e}^{(t)} \\ \mathbf{T}_{coupling} = \gamma_s^{(r)}(\mathbf{\Omega}_{target} - \mathbf{\Omega}_{go}) \\ \qquad +k_s^{(r)}\alpha\mathbf{u} \end{cases} \tag{2}$$

which corresponds to a zero-length damped spring of stiffness k_s and damping γ_s which links the position and orientation of the body to the target. The overall system to solve becomes:

$$\begin{cases} m\frac{d^2\mathbf{x}_{go}}{dt^2} + \gamma_s^{(t)}\frac{d\mathbf{x}_{go}}{dt} + k_s^{(t)}\mathbf{x}_{go} = \\ \quad \gamma_s^{(t)}\frac{d\mathbf{x}_{target}}{dt} + k_s^{(t)}\mathbf{x}_{target} + \sum \mathbf{F}_{collis} \\ \mathcal{I}\frac{d\mathbf{\Omega}_{go}}{dt} + \gamma_s^{(r)}\mathbf{\Omega}_{go} = \mathcal{I}\mathbf{\Omega}_{go} \times \mathbf{\Omega}_{go} \\ \quad +\gamma_s^{(r)}\mathbf{\Omega}_{target} + k_s^{(r)}\alpha\mathbf{u} + \sum \mathbf{T}_{collis} \\ \frac{d\mathbf{q}_{go}}{dt} = \frac{1}{2}\mathbf{\Omega}_{go}\mathbf{q}_{go} \end{cases} \tag{3}$$

The solving of these equations is out of the scope of this paper, but fast and stable techniques exist [12]. These equations make the position and orientation of the god-object $\mathbf{x}_{go}$ and $\mathbf{q}_{go}$ follow the state of the target $\mathbf{x}_{target}$ and $\mathbf{q}_{target}$. When it collides, the god-object is submitted to forces preventing interpenetration. These forces disturb this feedback control and make the god-object and its target dissociate. Depending on the collision, the collision forces makes the god-object stay blocked at its position or turn around the obstacle if possible. The collision responses are obtained from any techniques such as penalty methods for instance [13]. From the simulation point of view, the virtual tool behaves as any other dynamic body in the scene. Note however that a critical damping must be chosen for γ_s to provide a fast and stable feedback. In practice, we choose γ_s such that $1.4 \leq \frac{\gamma_s}{\sqrt{mk_s}} \leq 2$.

The Fig. 2 shows the complete mechanism[1]. We can see that the device only provides forces to the god-object, and not positions. In other words, we apply an admittance control (send forces/get positions) to the virtual tools, even if the device is an impedance one (sends positions/gets forces) [14]. Naturally, an admittance

[1]Note that a preliminary version of the scheme used a first order control loop corresponding to a kinematic equation of motion ($\frac{1}{mp}$ instead of $\frac{1}{mp^2}$), but we found such feedback control not stable enough.

haptic device can also be used. In that case, the haptic interface handles the god-object directly. That is, the control loop of the device sends $(\mathbf{F}_{target}, \mathbf{T}_{target})$ and gets $(\mathbf{x}_{go}, \mathbf{q}_{go})$.

Finally, we see on the figure that a force is provided to the device. In practice, this force is computed by an equation similar to Eq. (2), but with different constants (virtual coupling). Since these values can be 0 as well, no force is thus sent to the device, and the user does not feel any haptic feedback. However, the god-object control loop remains exactly the same, so its behavior is not altered and the virtual tools still takes the obstacles of the environment into account while it moves. Thus, we keep the same mechanism even if the device is not haptic. To summarize, we have shown that this mechanism is compatible with admittance or impedance, haptic or non-haptic, device.

Generalized dynamic god-object

As the mechanism relies on the dynamic equations of motion, we generalize the rigid body approach to allow the handling of more complex tools. Our main idea is to couple the degrees of freedom of the device to the ones of the god-object. If the god-object owns more degrees of freedom, they are not controlled by the device and their values are only determined by the simulation.

Articulated rigid body

In some applications, the use of devices which are made of rigid body articulations is needed (tools for laparoscopy simulation, or the complete PHANToM[2] articulation, for instance). This means that the device sends the position of all its degrees of freedom (angle for a rotational link, translation for a sliding link, etc...) to the workstation. In the virtual environment, we use a virtual body which owns the same kinematics and we want to make it follow the movement of the interface.

The previously presented mechanism remains compatible, but imposes to rely on a dynamic model of the articulated virtual body. We suppose that the articulated interface does not exhibit loops and the articulation can be represented as a tree. We thus choose an efficient yet robust method based on Lagrange multipliers [15]. The Eq. (1) is then written:

$$\begin{pmatrix} \mathcal{M} & -\mathcal{L}^T \\ \mathcal{L} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{A} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{B} \\ \mathbf{E} \end{pmatrix} \tag{4}$$

λ are the lagrange multipliers. $\mathcal{M}$ is a matrix constituted with all the generalized matrix $\mathcal{M}_i$ of each body i of the articulation:

$$\mathcal{M}_i = \begin{pmatrix} m_i & 0 & 0 & 0 & 0 & 0 \\ 0 & m_i & 0 & 0 & 0 & 0 \\ 0 & 0 & m_i & 0 & 0 & 0 \\ 0 & 0 & 0 & & & \\ 0 & 0 & 0 & & \mathcal{I}_i & \\ 0 & 0 & 0 & & & \end{pmatrix}$$

with m_i the mass and $\mathcal{I}_i$ the inertia matrix of each body i. $\mathbf{A}$ is a vector collecting the acceleration $\mathbf{A}_i = \left(\frac{d^2\mathbf{x}_i}{dt^2}, \frac{d\mathbf{\Omega}_i}{dt}\right)$ of all the bodies, where $\mathbf{x}_i$ is the position of the inertia center and $\mathbf{\Omega}_i$ the instantaneous rotational velocity. In the same way, $\mathbf{B}$ collects the forces and torques applied to each body i, $\mathbf{B}_i = (\mathbf{F}_i, \mathbf{T}_i)$.

In Eq. (4), $\mathcal{L}$ is the constraint matrix. To compute $\mathcal{L}$ and $\mathbf{E}$, every link constraint $g(\mathbf{x}, \dot{\mathbf{x}}, t) = 0$ is rewritten using the Baumgarte stabilization scheme [16]:

$$\frac{d^2\mathbf{g}}{dt^2} + \frac{2}{\Delta t}\frac{d\mathbf{g}}{dt} + \frac{1}{\Delta t^2}\mathbf{g} = 0 \tag{5}$$

where Δt is the simulation time step. We separate all the terms including the acceleration variables to compute the $\mathcal{L}$ matrix while the remaining terms form the $\mathbf{E}$ vector.

To virtually couple the interface and its corresponding god-object, we link each corresponding bodies of the two articulations with positional and rotational springs (see Fig. 3). Thus, coupling terms appear in the $\mathbf{B}_i$ vector of each body:

$$\left\{ \begin{array}{l} \mathbf{F}_i = \gamma_s^{(t)} \left(\frac{d\mathbf{x}_{target_i}}{dt} - \frac{d\mathbf{x}_i}{dt}\right) \\ \quad +k_s^{(t)} \left(\mathbf{x}_{target_i} - \mathbf{x}_i\right) + \sum \mathbf{F}_{collis} \\ \mathbf{T}_i = \gamma_s^{(r)} \left(\mathbf{\Omega}_{target_i} - \mathbf{\Omega}_i\right) + k_s^{(r)} \alpha \mathbf{u} \\ \quad + \left(\mathcal{I}_i \mathbf{\Omega}_i\right) \times \mathbf{\Omega}_i + \sum \mathbf{T}_{collis} \end{array} \right. \tag{6}$$

[2] *http://www.sensable.com*

The resolution of Eq. (4) is made by decomposing the acceleration in two different parts: the tendency acceleration ($\mathbf{A_t}$) and the correction acceleration ($\mathbf{A_c}$) [17]. $\mathbf{A_t}$ aims at determining the acceleration without considering constraints and $\mathbf{A_c}$ represents the correction needed on the acceleration to take into account the constraints. We have $\mathbf{A} = \mathbf{A_t} + \mathbf{A_c}$.

This way, the system is decomposed into three equations:

$$\begin{cases} \mathcal{M}\mathbf{A}_t = \mathbf{B} \\ \mathcal{L}\mathcal{M}^{-1}\mathcal{L}^T\lambda = \mathbf{E} - \mathcal{L}\mathbf{A}_t \\ \mathbf{A}_c = \mathcal{M}^{-1}\mathcal{L}^T\lambda \end{cases} \tag{7}$$

which are successively solved for $\mathbf{A}_t$, λ and $\mathbf{A}_c$. Since each part of the virtual articulated body tends to reach the position of its corresponding part on the interface, while respecting the linking constraints, we obtain an adequate overall movement.

If the interface is haptic, a force-feedback can be computed. Each rigid body i of the articulated interface undergoes the following forces and torques which are supplied from the coupling:

$$\begin{cases} \mathbf{F}_{feedback_i} = \gamma_d^{(t)}(\frac{d\mathbf{x}_i}{dt} - \frac{d\mathbf{x}_{target_i}}{dt}) \\ \qquad\qquad +k_d^{(t)}(\mathbf{x}_i - \mathbf{x}_{target_i}) \\ \mathbf{T}_{feedback_i} = \gamma_d^{(r)}(\mathbf{\Omega}_i - \mathbf{\Omega}_{target_i}) \\ \qquad\qquad -k_d^{(r)}\alpha\mathbf{u} \end{cases} \tag{8}$$

When necessary (depending on the interface API), these forces and torques can be applied to the parent body in the articulated body tree structure:

$$\begin{cases} \mathbf{F}_{parent} + = \mathbf{F}_{child} \\ \mathbf{T}_{parent} + = \mathbf{T}_{child} \\ \qquad +(\mathbf{x}_{child} - \mathbf{x}_{parent}) \times \mathbf{F}_{child} \end{cases} \tag{9}$$

Deformable body

We also propose a way to handle deformable tools. Since devices usually provide (at most) three rotations and three translations, we decide to control the rigid motion of the structure and let the simulation handle deformations. For that

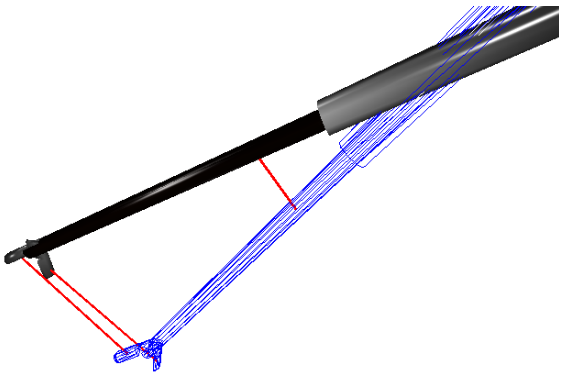

Figure 3: Articulated Graspers. The graspers are modeled as an articulated rigid body whose parts are controled by the interface (in wireframe). All the lines represent translational or positional damped springs.

purpose, it seems convenient to rely on models based on a rigid component such as [18]. This model consists in a mass/spring surface net which is connected via zero-length springs to a rigid body representing the rest shape of the net.

We can first consider that the position of this rigid component is directly controlled by the device (the rigid component is the target of the deformable shell). However, this can lead to instabilities of the mass net in case of fast movements. Instead, we consider the rigid component of the deformable model as a simulated virtual rigid body which aims at reaching the target position of the device (see Fig. 4). This rigid component is similar to a rigid god-object but is not displayed: Instead, the deformable surface shell is shown and interacts with the environment.

Results

All the tests presented in this section have been designed with our dynamic simulator [19]. It allows the simulation of different physical models, including rigid and various deformable bodies, as well as articulated bodies (that is rigid or deformable bodies linked by constraints). The collisions are handled by penalty methods using a sphere-based approach. The simulations are run on a bi-processor Xeon Pentium IV 2.4GHz 512Mb and use an implicit Euler integration. Videos can be found at the following url: *http://www.lifl.fr/∼meseure/GODOBJECT*.

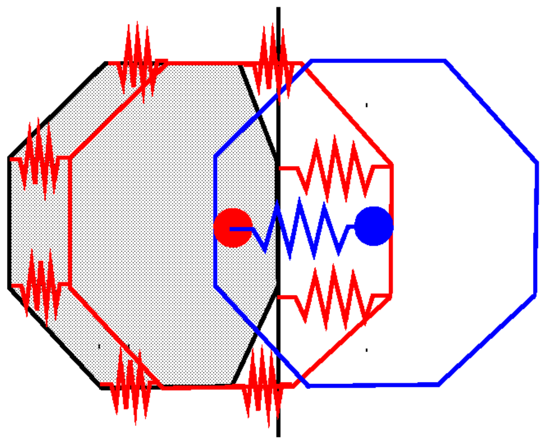

Figure 4: A deformable god-object. From right to left, we find the device position, the rigid component of the deformable god-object and the deformable body (filled with grey). This last is built with a mass/spring net which is linked to the rigid body.

In our first example, the virtual environment is a construction game. In Fig. 5, we show that it is not convenient that the interface controls directly the position of a block. This leads to collisions with other blocks or the board, which make the solving of the dynamics not stable. In Fig. 6, we use our generic interface. We have used the mouse as a non-haptic device, but the same simulation could have been done with a haptic device. We notice that the handled block movement always respects the integrity of the simulation environment, even if the user has followed a trajectory which passes through obstacles. In that case, the god-object is blocked at its place or tries to turn around the obstacle. Another advantage of the god-object can be used in such applications. By changing the virtual body that is used as a god-object, we provide a convenient way to select which tools the user wants to handle. This implies that, at certain moments, the interface does not control any virtual objects. The computation time of a simulation step takes from 4 to 8ms (depending on the number of collisions).

For the design of our laparoscopy simulator, we have represented the surgical graspers as a hierarchical articulated rigid body (see Fig. 3). The trochar moves around a fixed point (the insertion point inside the cavity). Its translations and the rotation around its main axis are blocked. Its movement is reduced to the combination of two rotations. The tube of the instrument slides through the trochar, so only translates along and turns around its main axis. The graspers or forceps can only open/close at the tip of the tube. However, the articulation does not exhibit enough rigidity. When in contact with an organ, the graspers tend to (easily) open wider, whereas a backward movement of the tube is expected instead. To get a high rigidity of the graspers position, we choose not to simulate them as rigid bodies but instead impose their overture. In Fig. 7, only the trochar and the tube

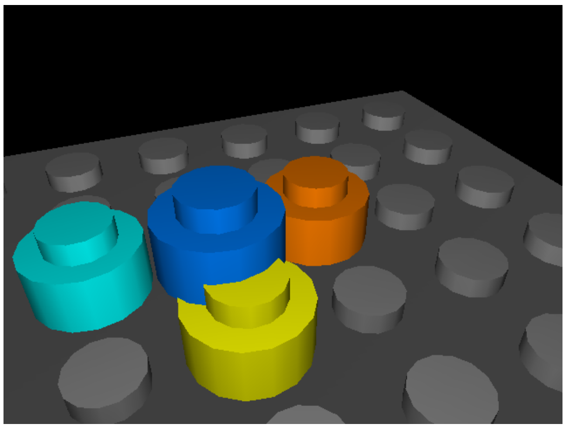

Figure 5: A virtual lego using a position-based approach. The position of the dark block is imposed by the user and deeply penetrates the light block whose simulation becomes instable.

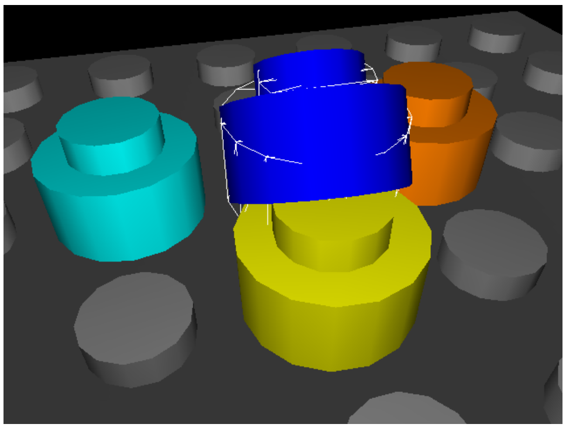

Figure 6: A virtual lego using god-object. The same position of the device as Fig. 5 is maintained but the dark block is simulated as a god-object and respects the environment.

are actually simulated. The resulting tool moves as expected to reach the position of the interface, turning around organs or obstacles if necessary and possible.

Due to the complexity of articulated body resolution, the computation time of a simulation step is from 10 to 20ms according to the arisen collisions. These timings imply that our model of laparoscopic tools exhibits a high latency which can disturb the user. Thus, a high computing power is required to reduce this latency. It is even more needed if a haptic device is used. Indeed, if the virtual object moves slowly because of the simulation high computation time, the user feels as if the virtual tool is "heavy" (where as it is only "slow"). Some specific optimizations and the use of faster workstations are likely to address this problem, but we want to point out that the simulation speed is a critical parameter.

Finally, some tests have been done to validate the approach for deformable bodies (see Fig 8). Compared to the rigid body interaction, deformations are first computed before the god-object actually turns around an obstacle. Naturally, a more realistic model of deformable body should be used, but we implemented this approach more as a proof of concept than for a real application.

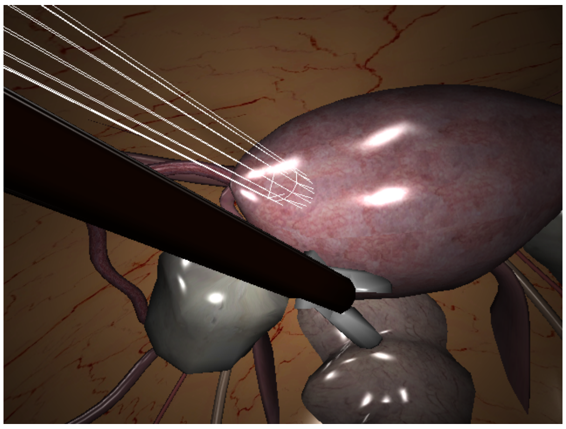
Figure 7: God-object graspers. The position of the interface is shown in wireframe and heavily collides the uterus while the god-object stays in contact.

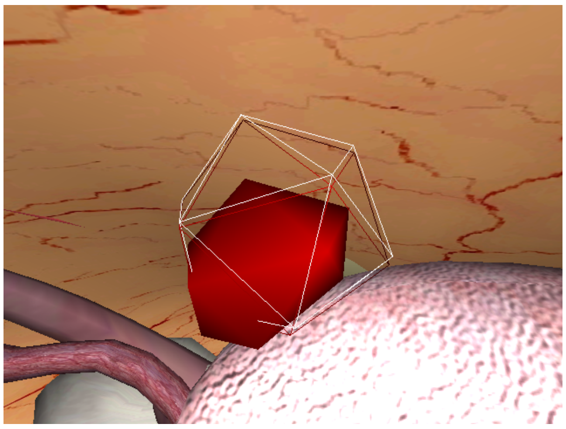
Figure 8: Deformable god-object. The position of the interface is shown in wireframe.

Perspectives

In this paper, we have shown that the god-object method is a versatile tool to drive both haptic and non-haptic devices. First, it provides a generic architecture: The position provided by a device is transformed to forces sent to a virtual object, whose behavior is computed by the laws of dynamics. When the virtual object can not reach the desired position, feedback forces are computed, which the device is free to exploit or not. Second, virtual objects are controlled via forces and no position is imposed to them. The virtual manipulated objects are forced to remain consistent with regards to the environment. This dynamic approach allows the extension of the god-object method to more complex interaction tools such as articulated and deformable bodies. We have implemented this system successfully for the control of surgical tools in a virtual environment and use it in our laparoscopy simulator. However, since no constraints are imposed to the motion of the device, the user can keep an impossible position of the virtual tool. It can be a drawback in certain applications: The learning of a certain gesture in a pedagogic simulator, for instance. So we plan to compensate the lack of haptic feedback by another sensorial way. Naturally, the use of visual and/or audio effects can be used. An illusion of haptic feedback [20] would maybe be more appropriate but new approaches need to be found for absolute-position devices.

References

[1] P. Dworkin and D. Zelter. A new model for efficient dynamic simulation. In *EURO-*

GRAPHICS Workshop on Computer Animation and Simulation, pages 135–147, Barcelone, September 1993.

[2] C.B. Zilles and J.K. Salisbury. A constraint-based god-object method for haptic display. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 146–151, Pittsburgh, 1995.

[3] D.C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In *SIGGRAPH 97 Computer Graphics Proceedings*, pages 345–352, Los Angeles, August 1997.

[4] T. Meyer, G. Andrade-Barroso, and B. et Arnaldi. Une architecture pour le retour d'efforts. In *Actes des 15èmes journées de l'AFIG*, December 2002.

[5] C. Mendoza and C. Laugier. Realistic haptic rendering for highly deformable virtual objects. In *IEEE Virtual Reality Conference*, Yokohama, March 2001.

[6] A. Lamouret, M.P. Cani-Gascuel, and J.D. Gascuel. Combining physically-based simulation of colliding objects with trajectory control. *Journal of Visualization and Computer Animation*, 1995.

[7] W.A. McNeely, K.D. Puterbaugh, and J.J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *SIGGRAPH 99 Computer Graphics Proceedings*, pages 401–408, August 1999.

[8] Y. Zhuang and J. Canny. Haptic interaction with global deformations. In *IEEE Int. Conf. on Robotics and Automation*, San Franciso, April 2000.

[9] D. Ruspini and O. Khatib. Dynamic models for haptic rendering. In *Advances in Robot Kinematics (ARK)*, pages 523–532, Salzburg, June 1998.

[10] D. Constantinescu, S. Salcudean, and E. Croft. Haptic feedback using local models of interaction. In *11th Symp. on Haptic Interface for Virtual Environment and Teleoperator Systems*, 2003.

[11] J. Colgate, M. Stanley, and J. Brown. Issues in the haptic display of tool use. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1995.

[12] L. Hilde, P. Meseure, and C. Chaillou. A fast implicit integration method for solving dynamic equations of movement. In *ACM Virtual Reality Software and Technology*, pages 71–76, November 2001.

[13] M. Moore and J. Wilhelms. Collision detection and response for computer animation. *Computer Graphics (SIGGRAPH 88)*, 22(4):289–298, August 1988.

[14] R. Adams and B. Hannaford. A two-port framework for the design of unconditionnally stable haptic interfaces. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1254–1259, 1998.

[15] D. Baraff. Linear-time dynamics using lagrange multipliers. In *SIGGRAPH 96 Computer Graphics Proceedings*, pages 137–146, New Orleans, August 1996.

[16] J. Baumgarte. Stabilization of constraints and integrals of motion. *Computer Meth. Appl. Mech. Eng.*, 1:1–16, 1972.

[17] J. Lenoir and S. Fonteneau. Mixing deformable and rigid-body mechanics simulation. In *Computer Graphics International*, Crete, June 2004.

[18] P. Meseure and C. Chaillou. A deformable body model for surgical simulation. *Journal of Visualization and Computer Animation*, 11(4):197–208, September 2000.

[19] P. Meseure, J. Davanne, L. Hilde, J. Lenoir, L. France, F. Triquet, and C. Chaillou. A physically-based virtual environment dedicated to surgical simulation. In *Int. Symp. on Surgery Simulation and Soft Tissue Modeling (IS4TM)*, pages 38–47, Juan–les–Pins, June 2003.

[20] A. Lécuyer, S. Coquillart, and P. Coiffet. Simulating haptic information with haptic illusions in virtual environments. In *NATO-RTA Human Factors and Medecine Panel Workshop*, 2000.