Realistic Computer Simulations Based on Visual and Force Feedback

Juan Manuel IBARRA ZANNATHA[1], Claudia MARMOLEJO RIVAS[1,2]
[1]Departamento de Control Automático, CINVESTAV
[2]Escuela de Informática de la Universidad Autónoma de Sinaloa
{jibarra ,cmarmolejo }@ctrl.cinvestav.mx

Abstract

In addition to surgical applications and Robotics, the haptic interfaces are used in scientific visualization, in training, to assist visually impaired people, and to built realistic simulation systems. The computer simulation based on virtual reality should calculate the inputs to the physical haptic interface, which depends on both simulation modeling and control algorithms. Moreover, they rely entirely on real-time computer input. Then, the haptic rendering to be used in control of the haptic interface needs the real time solution of the models representing the complete dynamical behavior of the simulated system. The main issues in the haptics domain are control of mechanical devices using force feedback; creation of realistic virtual worlds, production of realistic kinesthetic and tactile feels; teleoperation and telepresence together with data compression; and the construction of haptic devices The contents of this document have a general interest in the wide world of realistic simulations, but is more detailed in the aspects touching robotic application and the case with point contact obtained, by example, with the Phantom system, with emphasis in the haptic rendering aspects.

1. Introduction

Computer simulation is a powerful tool, very useful to understand or to control all kinds of natural phenomena or human designed systems. The objectives of computer simulation include the understanding of the simulated systems, the design of their components, the control of their behavior or the performance optimization of those systems. Dynamic behavior of natural or man made systems, kinematics of moving mechanisms, geometrical and topological properties of objects and the relationships between them, are some of the phenomena that we are interested in to simulate.

At its early stages, computer simulation gave only a listing with values of the variables of interest. The graphical representation of the simulation results was a big progress. But nowadays, Computer Science provides a wide variety of methods to display the results of simulations, including sophisticated 3D interactive virtual worlds, together with sound feedback. But there are some applications where this kind of feedback is not enough to evaluate the simulation results. In surgical training systems or in programming robot's tasks concerning contact with its environment, we need to involve more than the sense of vision, to have more realistic immersion systems. It is our sense of touch that provides us with much of the information necessary to interact with our environment. It involves information about physical properties such as inertia, friction, compliance, roughness, and temperature. We use the term *haptic* to evoke the sense of touch or something related to this sense [8]. Then, a physically accurate simulation joining all kinds of sensorial information, like vision, sound, and touch, let us obtain insight into the real-world behavior of the dynamic system under study, and enhance the level of immersion in a virtual world.

Haptics can be divided into two categories. First, the kinesthetic sense, based on force feedback, through which we sense movement or forces in muscles and joints corresponding to the weight of the grasped virtual objects, their mechanical compliance, inertia, as well as motion constraints. Secondly, the tactile sense that allows human operator to feel the roughness of virtual surfaces (textures), their edges (shapes), and their temperature.

In this frame, we need a touch interface, known as haptic interface, to reproduce kinesthetic and tactile senses corresponding to the interaction of the user with the objects *living* in virtual worlds. Haptic interface is a force-reflecting mechanical device used to apply forces to a human operator, typically through his finger or hand, creating the illusion of physical contact with a real physical environment. The main goal of the haptic interface is to command the movements of the objects in a virtual environment, displaying to the user the corresponding interaction forces, together with the 3D visualization of the corresponding virtual environment. Then, the use of haptic interfaces allows very realistic virtual reality (VR) simulations.

This document intends to be a survey on the primary issues in Haptics, starting with the presentation of the main kind of haptic devices used in the area of force feedback simulations or teleoperation, and some interesting points about the control of mechanical devices to produce force feedback. The aspects about the haptic rendering problem, together with the dedicated to haptic devices, are the subject of one of the most detailed chapters, including: physical modeling (kinematical and dynamical models), collision detection, grasping, object deformations and generation of the interaction forces.

2. Haptic Devices

Haptic mechanical interfaces and its supporting software are alternative or supplementary input devices to the mouse, keyboard, or joystick, that allow us to feel and manipulate objects laying in virtual worlds in a very realistic way considering the shape, texture, weight, stiffness, and temperature of that objects. Haptic devices are capable, through its actuators, to deliver some forces to the user proportionally to his interaction with the virtual world. In this sense, haptic interfaces have a bi-directional energy exchange between the user and the computer (see figure 1), unlike devices used exclusively as input (mouse, keyboard) or output (visual and sound feedback).

When the user changes the position of the haptic interface, or applies forces, data are transmitted to the computer at very high rates. Once captured, the data are processed running specialized haptic programs (graphic rendering, collision detection, haptic rendering, etc.) wich refresh the manipulated virtual world. In response to the changes in the virtual world, the computer sends, to the haptic interface, the corresponding positions and forces to be felt by the human operator.

In Computer Graphics and image processing applications, low scene refresh rates of 20 to 30 frames/sec are enough to satisfy the human sensorial requirements. In contrast, the response of our sense of touch is better to vibrations of 200 to 300 Hz or higher. The difference between haptics and vision bandwidths is one order-of-magnitude, motivating the use of a dedicated controller for the haptic interface to guarantee not only high fidelity force feedback, but also dynamic stability.

Taxonomy of haptic devices is not an easy task. We can classify them depending on the type of feedback used, their grounding arrangement, the type of actuators used, or the application area. Here, we are interested in general-purpose haptic devices, as opposed to those developed for a specialized application, such as surgical training. Most of the actual haptic interfaces use electrical actuators, especially DC motors, because of their ease of installation, control and cleanliness, with pneumatic and hydraulic actuators being less common. Depending on the type of feedback used, the haptic interfaces can be tactile or kinesthetic. Haptic devices producing force feedback (kinesthetic type), as opposed to devices with tactile feedback, can actively prevent a user from moving into restricted simulation space. However, this kind of haptic devices lacks the rich contact surface information produced by tactile feedback. High bandwidth haptic interfaces combining tactile and force feedback, such as the PHANToM arm, are best suited to fulfill the requirements of realistic VR simulations.

To produce some forces to the user, the haptic device needs a fixed mechanical reference. The haptic interfaces must be attached (*grounded*) to an immovable support to be able to apply forces to the human operator and to resist the actions of the user, even stopping him through large feedback forces. The immovable support provides also equilibrium and mechanical stability of the haptic device. Then, there are two kinds of haptic devices, depending on the grounding arrangement: those with reference in the user's body and devices with reference in a desk, in a wall, etc. Exoskeletons and gloves are devices of the first type, while most of the haptic devices from the second group are little desk-grounded articulated mechanisms, even if, in this group, there are also 6 dof (degree of freedom) robots, haptic joysticks, and a special kind of force feedback mice.

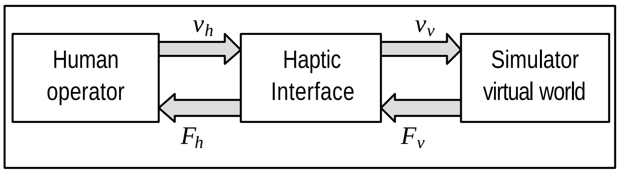

Figure 1. Haptic Interface viewed as a two-port System.

2.1. Desktop haptic devices

The taxonomy adopted in this document takes into account the type of attachment used by the haptic mechanical interfaces, but there are some other criteria to classify these devices. We can consider, for example, the kind of energy used as criteria, to distinguish between pneumatic, hydraulic or electric force feedback devices. Haptic devices can be classified also following the type of control or communication loop; we can have thus position control based or force control based haptic devices, and hybrid systems based on both position and force control.

Desk-grounded haptic devices are, in fact, small robot arms with at least three joints; DC motors actuate on each of them. A computer controls the haptic device to emulate the sense of touch producing a force proportional to the reaction exerted by the virtual environment, sending voltages to the motors when a collision between the tip of the haptic device and virtual objects is detected, to materialize the reaction force.

Joysticks. For many years, joysticks have been used in entertainment, computer graphics and industrial control applications. New versions of this input device include DC actuators to produce active forces to be felt by user, but they need more than the classical 2 DOF to exert a realistic feedback forces on the user's hand. The force feedback joysticks can have the classical spherical configuration with an extra rotational DOF or a Cartesian configuration with a third rotational DOF. The joystick can have also the Stewart Platform configuration with 6 DOF. The parallel cinematic structure of the Stewart Platform has better position accuracy, higher load capacity, and can exert big efforts than serial configurations like 3 DOF arms or joysticks. Its smaller work volume, the difficult force control and its complex direct kinematics are some of the Stewart Platform drawbacks.

The PHANToM. Produced by SensAble Technologies in Cambridge, MA, USA, the PHANToM™ is a small pen-based desk-grounded robot that permits VR simulation of single fingertip contact with virtual objects through a thimble or stylus. This is a weight counterbalanced and back drivable arm whose workspace is those of the user's wrist. It tracks the position and orientation of the virtual probe as it moves about the 3D workspace, and its actuators produces forces back to the user's fingertip as it detects collisions with objects in the virtual world. The haptic feedback of the PHANToM™ arm is ext remely crisp, due to its low inertia and static friction combined with its very high control bandwidth (1,000 Hz). This kind of desktop haptic device can reproduce both the kinesthetic sense (force feedback) and some features of the tactile sense. [11].

The FEELIt™ Mouse. This device, produced by Immersion Corporation in San Jose, CA, USA, is an example of a desktop tactile feedback interface. It is a 2 DOF mouse used to feel the roughness of the virtual object when the associated arrow is traversing one of its virtual surfaces in the screen. It is possible also to push with the arrow into virtual objects to determine their elasticity.

2.2. Wearable haptic devices

CyberTouch™. Wearable interfaces are haptic devices grounded to the user's body that has the workspace of the user's arm, giving him a larger work volume than desktop haptic devices do. These devices are, in fact, gloves that allow users to interact with virtual worlds in a dexterous way through natural hand gestures. An example of a tactile feedback glove is the CyberGlove™, used by the CyberTouch™ and the CyberGrasp™ devices, where 18 sensors measure the hand position. In the CyberTouch™ produced by Virtual Technologies, Palo Alto, CA, USA, the feedback is produced by six small vibro-tactile actuators, placed on the back of each finger and in the palm, vibrating at frequencies of up to 125 Hz.

The CyberGrasp™. This device from Immersion Corporation, is a 22 DOF exoskeleton that fits over a CyberGlove™, providing force feedback. In this device, the vibro-tactile actuators are replaced by a complex exoskeleton on the back of the user's hand formed by tendons transmitting forces produced by DC motors placed in a control box. These tendons can produce a continuous force of 12 N on each finger when the user is closing his hand. In some models an external device, like Polhemus Fastrak, measures the position and the orientation of the glove in the 3D space.

Desktop interfaces have a very limited workspace but can exert more important forces than wearable devices. Desktop devices are more ease to use and the relationship between compactness and force capabilities is better than in wearable devices. Gloves and exoskeletons have a large weight, and have important difficulties to generate forces corresponding to the virtual objects weight. Nevertheless, each kind of haptic device has its own application area.

Haptic Interfaces are articulated arms with a certain number of DOF formed by linkages and motion transmissions, where each joint is equipped with an encoder measuring its angular position, and a DC motor with its servo amplifier and D/A converter moving that joint. Different kinds of haptic device have the same electromechanical structure; in figure 2 is depicted a scheme of a general haptic interface.

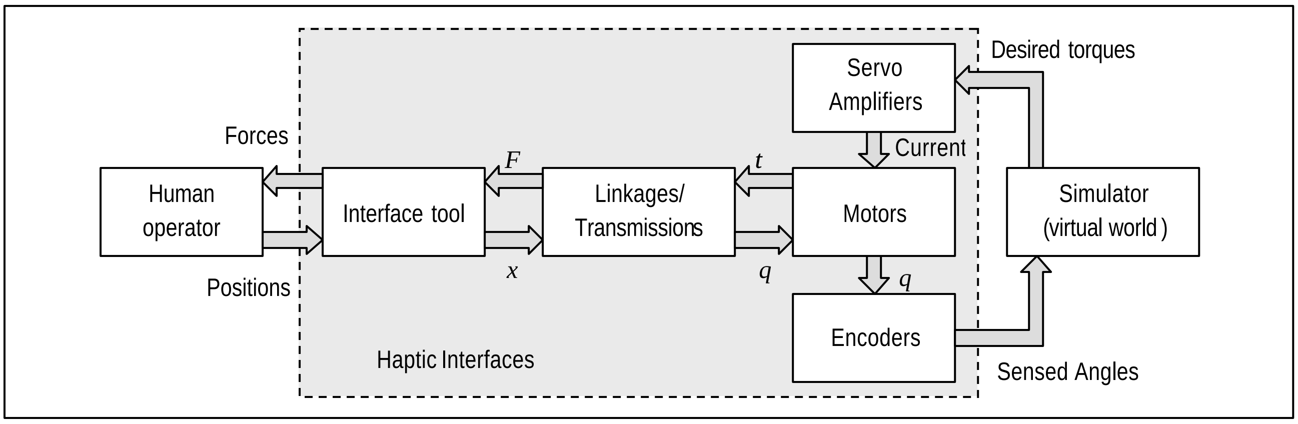

Figure 2. Scheme of a Haptic Interfaces from control viewpoint.

3. Force Feedback

The objective of the haptic interfaces is to produce the force feedback materializing the sense of touch, it allows a person to feel the weigh of virtual objects, or the resistance to motion that they create. Force and velocity are the key variables that define the nature of haptic contact. The goal of control law design for haptic displays is to provide a safe and stable user interface while maximizing the operator's sense of kinesthetic immersion in a virtual environment, it must provide realistic sense of touch taking into account two main requirements: high fidelity rendering and stability. But these two objectives are opposite in the sense that high fidelity haptic rendering need high force feedback gains generally producing self-induced oscillations and instability.

There are two classes of control schemes available for force reflection: impedance control and admittance control. Impedance controlled systems detect the motion commanded by the operator and control the force applied by the haptic device as in figure 1. Admittance controlled systems detect the force commanded by the operator and control the velocity or displacement of the haptic device [3]. Impedance controllers were generally used when the environment being simulated was highly compliant, such as human tissue in surgical simulators. Admittance control was generally used when the environment was unyielding such as flight simulator p lataforms [4].

The haptic interface can be considered like a two-port system terminated on one side by the user and by the virtual world on the other side. In that scheme, inspired on electrical networks [1], a force F_h and a velocity v_h characterize the energy exchange between the user and the haptic interface, whereas a force F_v and a velocity v_v represent the exchange between the interface and the virtual world. To have an ideal behavior, from the haptic rendering point of view, the haptic interface should be *transparent* (F_h=F_v and v_h=v_v, in figure 1); but haptic system requirements, from the stability point of view, demands to introduce some haptic distortion.

The interaction between the user and the virtual world is a bi-directional transfer of energy, since force multiplied by position represents mechanical work. According with Hannaford, B. et al. [7], the two port network with initial energy storage at t=0 of $E(0)$, is passive if, and only if:

$$\int_0^t (f_h(t)v_h(t) + f_v(t)v_v(t))dt + E(0) \geq 0 \;\; \forall t \geq 0$$

the energy applied to a passive network must exceed $-E(0)$ at all times, which means that the system dissipates energy, otherwise it is active because it generates energy. Active ports may contribute to inestability. The behavior of the user interacting with haptic interfaces is passive, because he does not introduce energy into the system [9]. Moreover, most mechanical virtual worlds are also passive, then, the stability of the overall system is guaranteed when the haptic interface is designed to be also passive. However, the sampling process perturbs the natural passivity of the virtual environment. In fact, it was shown that the smaller the sampling rate, the more energy can be generated by a virtual wall [5].

Adams *et al* [1] use a Passivity Observer (PO) and a Passivity Controller (PC) for reducing the performance compromise required for stable contact applied to haptics and bilateral teleoperations. When there are multiple interconnected elements, they observe each one separately in order to determine which ones are active and which are passive. The PO may or may not be negative for any one port element in the system at a particular time, but if it is negative, then the port may be contributing to instability, as they know the exact amount of energy generated, a time varying element, called a Passivity Controller, is designed in order to dissipate only the required amount of energy.

The basic haptic interface simulation consists of the human operator (HO), the haptic interface (HI), the passivity controller (PC) and the virtual environment (VE) as shown in figure 3 [1].

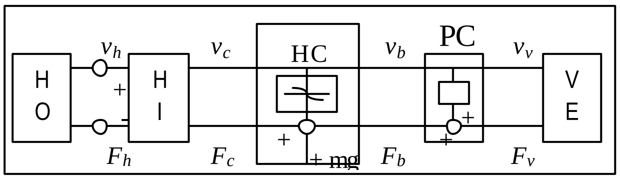

Figure 3. Haptic Interface viewed as a several port System.

3.1. State of the art in control of haptics

In any kind of application, virtual worlds of interest are always nonlinear and the dynamic properties of a human operator are always involved. These factors make it difficult to analyze haptic systems in terms of known parameters and linear control theory. The rapid growth of haptics makes control engineering more important. The objective of the control is to ensure the stability of the haptic system (including the haptic device, the corresponding software and even the human operator) creating a compelling sense of haptic presence.

4. Haptic Rendering

In the context of a dynamical system simulated in a computer like an interactive virtual environment, a human operator should be able to manipulate objects living in a corresponding virtual world sensing both a 3-D visualization and the interaction forces between his hand

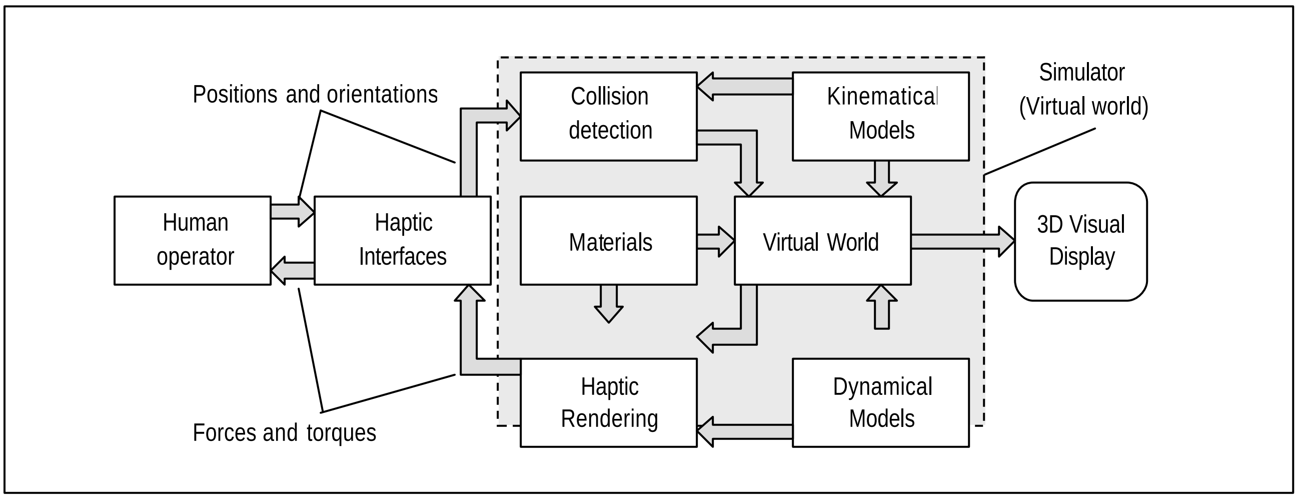

Figure 4. Virtual simulator should produce both visual display and haptic rendering.

and the manipulated virtual objects. Here, we call Haptic Rendering the display of the acting forces between user and the virtual environment through a motorized haptic device, used to manipulate the virtual objects, permitting the user to feel those forces.

A computer simulator capable to produce visual and force feedback is a complex system based on the real time evaluation of geometrical, kinematical and dynamical models (See figure 4). Geometrical and kinematical models are used not only to display the corresponding 3D virtual environments but also to detect the interaction between user and virtual objects living in this environment (collisions) and to calculate the geometric restrictions of virtual objects, including the user's probe. In the simulator, dynamical models are used to calculate the movement behavior of all the objects in the virtual world, taking into account the forces producing that movement, and to produce the haptic rendering corresponding to their interaction with the user.

Inside the virtual environment the human operator can move the virtual probe (manipulator, finger tip, specialized tool, etc.) using the haptic interfaces, generating physical contact between the virtual probe and the virtual objects. At this time, the simulator should detect the corresponding interaction and generate the graphic actualization of the virtual world using geometrical and kinematical models. Moreover, the simulator must calculate and generate the corresponding reaction forces like a repulsive force proportional to the amount of probe penetration into a virtual object, and to calculate and display the effects of these forces, like elastic and plastic deformations. The calculation of these forces and their effects over the virtual objects are based on the dynamical models and depend on the kind of materials supposed to be used to build that objects.

4.1. Geometric and Kinematical Models

First of all, a virtual world is a very realistic simulation capable to produce some kind of sensorial feedback to the human operator, where the visual feedback is an interactive 3D graphical environment. To have a visually realistic simulation the virtual objects must be characterized by its geometry (shape, size), and its surface (texture, color), while the position and orientation of all the virtual objects present in the virtual environment and their movement are ruled by its kinematics. Polygonal objects, spline or algebraic surfaces, implicit surfaces, CGS models, oct-trees, *k-d* trees, and deformable bodies are some of the 3D models that can be used to represent the geometry of the virtual objects. The 3D geometry and, even, some surface properties of the objects living in the virtual world can be created using languages specially conceived to built rigid bodies in a virtual reality context, as Java 3D, Visual Studio, OpenGL, and VRML (Virtual Reality Modeling Language). In these languages some primitives allows to construct complex interactive virtual worlds.

Often the objects to be represented in a given virtual environment are very complex or the application needs a very precise geometric representation, introducing a supplementary difficulty to generate their representation. In those cases, it is possible to acquire the real object by digitalization using some stereovision techniques, including the medical imaging systems (computer tomography, nuclear-magnetic resonance). Some commercial haptic devices have utilities to acquire geometric 3D information about the objects to be digitalized. By example, the GHOST SDK system, which is the development toolkit for the PHANToM^{TM} haptic device, permits to construct virtual environments

In despite of the method used to build the virtual environment (construction or acquisition), it is always possible to use photographic images as visual rendering primitives. For the visual feedback purposes, the surface representation of the virtual objects is quite enough, but the haptic rendering objectives demand a volumetric representation of the virtual objects. Nevertheless, in some simple applications realistic haptic interaction can be obtained between virtual objects modeled only by surface representation. Independently of the geometric model (surface or volumetric representation), an Internet browser equipped with the specific plug-in is used to display virtual environments created with a virtual reality languages or acquired by digitalization.

The movements of the objects in the virtual environment are commanded by their kinematics, equations relating the actuator's output variables with the position and velocity of each virtual object. Given a mobile object in the virtual environment, its attitude and its position is calculated by an external program based on the kinematics of this object. By example, the kinematics of a given mobile robot permits the calculation of the position and orientation of this object from the values of the position and velocity of the robot's actuators moving their wheels.

In order to have a virtual environment with a realistic visual feedback, we need to solve the geometrical as well as the kinematical model which represent, respectively, the aspect and movement of the virtual objects under consideration. Before calculating the reaction forces trough dynamical models that match with the interaction between the virtual object and the user's probe, we need to detect and measure the corresponding contact.

4.2. Collision detection

The collision detection problem is found in computer-simulated environments, computer aided design and machining (CAD/CAM), manufacturing, computer graphics, animation and Robotics. It is a bottleneck in all kinds of applications involving contact analysis and special reasoning among static or moving objects, such as motion planning, animation of articulated objects, assembly and disassembly, tolerance verification, or obstacle avoidance. In the haptic loop, collision detection is the first step, whose goal is to automatically report any geometric contact between the user's probe and the geometrical model. In computer graphics, collision detection is used to detect if there is overlapping of two given objects, while in haptic rendering its goal is not only to verify collisions between objects, but also to calculate the appropriate interaction forces to convey to the user the corresponding tactile feel.

In computer graphics, collision detection of static objects has many solutions. When the virtual objects are in movement, the trajectory of the analyzed object must be known a priori and parameterized as a function of time. At fixed time intervals, the consecutive positions of the object along of the known trajectory are calculated to check the interferences with other objects in the virtual world. When a human operator moves the objects, their position, velocity and acceleration are unconstrained variables with abrupt changes in direction and speed, making collision detection a highly consuming computational resources activity. When the considered objects have a complex geometry, to detect collisions may be a hard task [6], [2].

In a simple approximate approach a contact is detected when interference between the bounding boxes associated to virtual objects appear. In this method, the bounding box is a rectangular prism, whose edges are aligned with the axes of the world system of coordinates, constructed to enclose the entire object. In the Axis Aligned Bounding Boxes method (AABB), interference between two bounding boxes (rather than the corresponding objects) is detected whenever they overlap in all three orthogonal projections of both prisms.

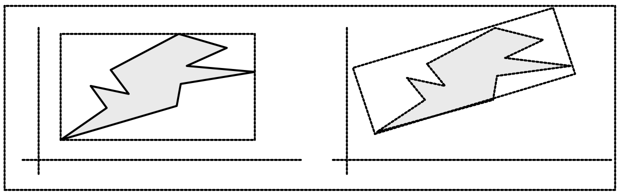

Figure 5. Bounding boxes produced by AABB and OBB algorithms respectively.

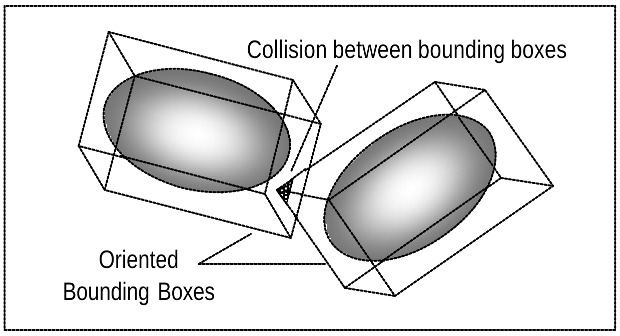

Figure 6. Collision between bounding boxes are not necessarily collision between the corresponding objects.

There is static and dynamic bounding, static ones have constant dimensions, and they need to be large enough to accommodate any possible orientation of that object in space; while a dynamic bounding box changes its dimensions as a function of the object's orientation even tough it remains aligned with the world's coordinate system. Dynamic bounding boxes reduce the volume wasted by static ones at expense of increased computational load.

The AABB method works fast but its low accuracy may pose some problems if dexterous manipulation of virtual objects is needed (see figure 6). Nevertheless, it is possible to increase the accuracy of the collision detection if the bounding box is oriented with the direction minimizing the non-occupied volume in the bounding box. In the oriented bounding boxes (OBB) algorithm, based on the statistics of the object, the bounding boxes are oriented with the basis formed by the normalized eigenvectors of the covariance matrix (see figure 5). The iterative version of this algorithm produces a hierarchical model, the OBB-Tree, formed by a sequence of overlapped bounding boxes locally oriented, producing a tight fitting representation. The OBB-Tree is a highly structured model reducing computational cost in collision detection when there are many objects or when they are very complex.

In the AABB or OBB algorithms we can use also bounding spheres, ellipsoids, cylinders or cones. The choice of the bounding volume is driven by two conflicting constraints: the selected volume should fit the original object as tightly as possible, and testing two such volumes for overlapping should be as fast as possible. In the AABB algorithm, the use of spheres permits the rapid test of the interference, but it has some problems to accommodate tightly long and thin oriented objects. Using ellipsoids in the OBB algorithm produce tight fits but checking interferences is an expensive task from computational viewpoint.

In both AABB and OBB algorithms, not all overlapping boxes will correspond to a collision, due to their unoccupied volume (see figure 6). To actualize the 3D virtual word for visual display this kind of approximate collision detection algorithms may be enough, but to calculate the force feedback needed in the haptic device, an exact method is required, at least in the approach and touching-grasping phases. Exact collision detection requires, for static polyhedral objects, to test every vertices of one object versus those of the second object, to investigate the possible interference between them.

In despite of the complexity of the exact collision detection methods, in the case of a virtual probe trying to touch simple virtual objects, it is possible to generate actualizations of the virtual world at a rate enough to ensure stability of the force feedback loop and to produce realistic haptic rendering. Rather than collision detection between two arbitrary objects, in this case, we should calculate the interference between a virtual probe, corresponding to the haptic device, and a given object. Two types of haptic interaction paradigms can be implemented here: point-based and ray-based. For the point-based haptic paradigm, the probe is simply modeled as a point. For the ray-based collision detection algorithms, the probe is modeled as a line segment. Both techniques have advantages and disadvantages. For example, it is computationally less expensive to render 3D objects using point-based techniques, allowing higher haptic servo rates. On the other hand, the ray-based haptic interaction technique handles side collisions and can provide additional haptic cues for conveying to the user the shape of objects.

4.3. Dynamical Models

To reproduce the kinesthetic and tactile senses, we need to determine the dynamical behavior of the interaction of the different objects living in the (simulated) virtual world, based (primarily) on Newtonian physical laws. Dynamical models, allowing to calculate and to generate the feedback forces corresponding to those interactions, should include elastic and plastic surface deformations, some physical constraints and models to simulate hard contacts (between rigid objects). When compliant virtual objects are grasped there are some surface deformations in the region of contact between the robot gripper and the grasped object. The geometric properties of those deformations, that can be elastic or plastic, are calculated and displayed using different kind of models as vertex-based models or spline-based models. While the forces associated with these deformations will be calculated by dynamical models taking into account the physical properties of the considered objects (mechanical compliance, mechanical impedance and elasticity). To produce more realistic simulations, physical constraints as gravity or friction may also be considered in the physical modeling of the virtual world.

The mechanical compliance of a given object represents its softness or hardness feeling during static interactions. A given object deformation corresponds to small forces if this object is soft, or to large forces if it is hard. To take into account dynamic effects we can use a more general variable, the mechanical impedance, which grows with the object mass, velocity, and acceleration. The elasticity of the virtual object is a physical property with an important influence on the forces during surface deformation. Elastic objects regain their original shape once the deforming force ends, while the no elastic objects remain deformed. The rigid (very stiff) objects generate large interaction forces without surface deformation.

4.4.1. Surface mechanical compliance

In the haptic loop context, one's interest is to calculate elastic deformation of virtual objects for their visual displaying, when they are manipulated by a human user through an haptic device; and, simultaneously, the generation of the forces resulting of this interaction to produce the corresponding kinesthetic feedback. In this kind of realistic simulations, the immersion of the user in the corresponding virtual world is made through a haptic device representing the movements of one of the user's fingers simulated in the virtual space as a probe that can be modeled using one of two different paradigms: a point-based or a line-based paradigm.

Point-based paradigm greatly simplifies the development of both haptic device and haptic rendering algorithms. Moreover, it allows a bandwidth and force fidelity that enable the simulation of a wide range of interactions. Here, the problem of computing the haptic rendering is reduced to one of tracing the motion of a point (the virtual probe tip) among the virtual objects and producing the forces representing the interaction between the virtual probe with that objects. Zilles and Salisbury call *haptic interface point* (HIP) to the endpoint location of the physical haptic device, as sensed by the encoders.

Due to the inherent mechanical compliance of the haptic devices, the maximum stiffness of any virtual object is limited. Then, the HIP often penetrates into a virtual object a greater distance than that possible in real life. In early haptic rendering systems, this penetration used to calculate directly the corresponding feedback force, this method is well suited to model simple virtual objects (planes, polyhedral or spheres). But, the use of simple mechanical impedance to model surface contacts has some drawbacks: small and thin objects do no have the internal volume required to generate realistic interaction forces; traversing volume boundaries generates force discontinuities; it is often unclear which piece of internal volume should be associated with which surface [14].

Hooke's law, the most used model of elastic deformation in virtual reality haptic rendering systems, is expressed as:

$$\mathbf{F} = k\,\Delta x$$

where the constant k is the stiffness of the virtual object and Δx is the surface deformation along a specified direction. This linear equation modeling the object stiffness is well suited for real-time force simulation. When the HIP is approaching a virtual object and goes inside it, the haptic rendering algorithm must calculate the feedback reaction force based on Hooke's law. This force is proportional to the penetration and should be normal to the touched surface. But, once inside, it is no possible to know what was the penetrated surface and it will be choose the nearest one.

This solution can produce some mistakes, which can be avoided dividing the polyhedral objects in sub volumes whose base is the considered facet and its apex is the centroid of the considered object. Thus, the direction of the reaction force will be normal to the facet of the sub volume having the HIP (see figure 7). For small and thin objects, the result can be worst. Once the HIP is inside the virtual object, a reaction force proportional to the penetration is calculated, but the user will continue to penetrate more deep into the object, due to the limitation on the haptic device stiffness. Then, it is easy to put the HIP in the next sub volume forcing the algorithm to generate a reaction force producing the HIP pass through the virtual object, as depicted in figure 8 [14].

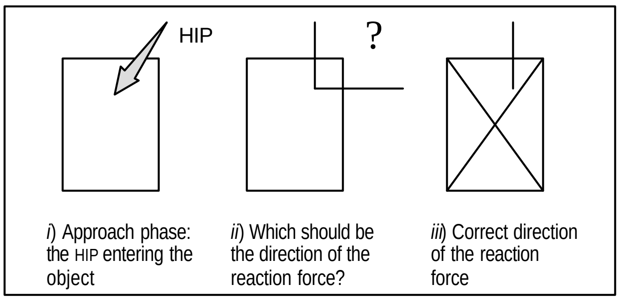
HIP
?
i) Approach phase: the HIP entering the object
ii) Which should be the direction of the reaction force?
iii) Correct direction of the reaction force

Figure 7. Using subvolumes to compute the forces (adapted from Zilles and Salisbury, [14])

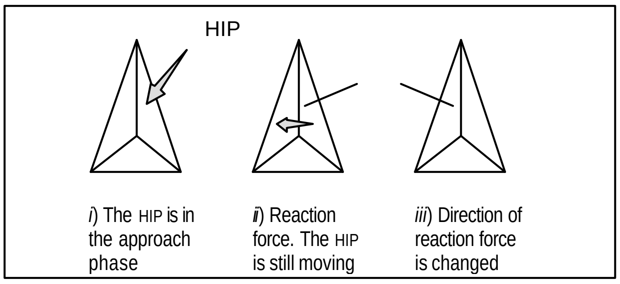
HIP
i) The HIP is in the approach phase
ii) Reaction force. The HIP is still moving
iii) Direction of reaction force is changed

Figure 8. The hip pass through a thin object (adapted from Zilles and Salisbury, [14])

An alternative method to calculate the reaction force is to consider, not the penetration of the probe tip into the virtual object, but instead to constrain the motion of a substitute virtual object. In the method proposed by Ruspini, Kolarov, and Khatib, [13] a representative object, a *proxy*, substitutes the probe tip (HIP), in the virtual environment. The method considers the virtual *proxy* connected to the HIP by a stiff spring. As the user moves the probe tip in the virtual workspace of the haptic interface (figure 9.a), it may pass through one or more virtual objects. However, these objects stop the *proxy* (figure 9.b), but quickly it moves, keeping always the contact with the object surface, to a position minimizing its distance to the HIP position commanded manually by the user through the haptic interface (figure 9.c). Thus, the force feedback will be produced by the virtual spring connecting the *proxy* with the HIP.

The method using the *god-object*, first proposed by Zilles and Salisbury [14], represents an equivalent approach to the *proxy* object, but here a priori knowledge of the surface topology is required. The *god-object* represents the virtual position of the HIP on the object's surface, constrained by the planes forming this surface, while the HIP is really commanded by the user toward the interior of that object. Lagrange multipliers are used to determine the actual location of the *god-object* during contact with a virtual object. This location is chosen to be the point that locally minimizes the distance between the *god-object* and the HIP, subject to the constraints that the *god-object* is on a particular surface. Once the *god-object* location is determined, simple impedance control techniques may be used to calculate the force to be displayed. A stiffness and damping can be applied between the HIP and the *god-object*, representing local material properties [14].

4.4.2. Surface deformation

In most cases, grasping assumes that the virtual hand or gripper and grasped object are undeformable, because of real-time computation constrains. Usually, the grasped object should change its shape in response to the grasping force applied by the user, and regains it once released if the deformation is elastic, or remains deformed when the deformation is plastic. Therefore the need of interactive surface deformation methods that satisfy the real time requirements of realistic simulation, classified as vertex based and spline-based, depending on whether the object surface is represented by polygonal meshes or parametric equations.

Vertex-based Methods. In the 3D models based on polygon meshes, vertices and edges define each polygon. Both vertices and edges are shared by adjacent polygons, then, storing all the polygons forming the virtual object implies to store some vertices several times. To avoid the problem, some graphics languages save the mesh as a look-up table with pointers to edges and, subsequently to vertices. In systems based on this kind of 3D geometrical model the user can interactively change the location of a vertex and its neighbors, according to some application-dependent deformation propagation law, in the corresponding look-up table, redefining the shape of polygons sharing it during the image rendering. When a virtual probe touch the surface of a 3D object the touched point goes inside the object pulling-in its neighbors following a given deformation propagation law. The surface of objects with a complex geometry, as body organs (liver, kidney, gall bladder), can be considered as an active surface. An active surface is an energy-minimized polygonal mesh which, when deformed, will seek to return to a low-energy state. The energy minimization process is modeled with ideal springs attaching each mesh vertex with its neighbors and between the current and rest vertex positions. The object mesh look-up table will require supplementary information about the position of the considered vertex (home, current) and the external applied force.

Spline-based Methods. We can use functions of higher degree than linear functions describing a polygonal plane, to reduce the storage needs, and to provide increased surface smoothness. Parametric bicubic surfaces are described by three point coordinates $x(s,t)$, $y(s,t)$, $z(s,t)$ being a function of two parameters s and t. A particular point location on this surface depends on the particular parameters s and t values, where $s, t \in [0,1]$. Thus, the points (0,0), (0,1), (1,0) and (1,1) correspond to the end points of the parametric surface. Several such patches can be joined or splined smoothly at knot points by assuring continuity of the first and second derivatives. Depending on the control points that determine the values of the constant coefficients, we can have different kinds of splines: hermite splines, Bezier splines, and *b*-splines.

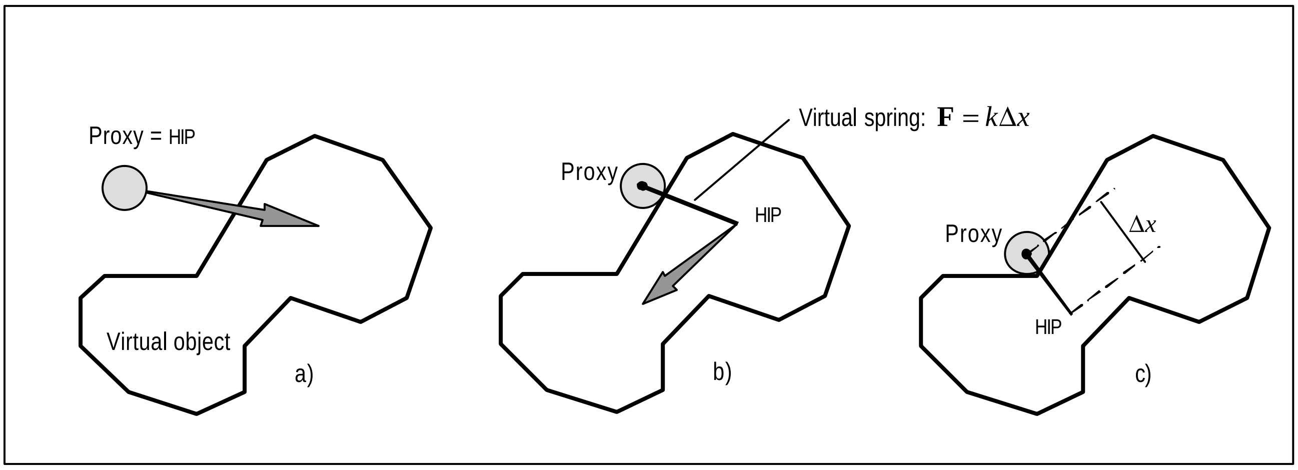

Figure 9. Haptic rendering using a proxy object. (Adapted from Ruspini and Kolarov, [13])

5. Conclusions

In order to create the illusion of touching a virtual object in a user of a haptic device, one has to solve several issues: a geometrical model, a dynamical model associated to the geometrical one, the use of a collision detection technique, a real time reaction force calculation in response to the user's manipulation, the stable and fast enough realistic visual and force rendering so that the user keeps immersed in the virtual world.

As we count on limited human and financial resources, our project is limited to the study of the geometrical and dynamical models of deformable objects that conforms the heart of the haptic system, which in the future may lead to get resources to buy haptic devices that contribute to close the haptic loop that provides a surgical application in the laparoscopic or endoscopic modes or an industrial one for training or prototyping production.

References

[1] Adams, R.; Hannaford, B. "*Control Law Design for Haptic Interfaces to Virtual Reality*" IEEE Trans. Control Systems Technology, vol. 10, pp. 3-13, Jan 2002.

[2] Burdea, Grigore C. "*Haptics Issues in virtual Environments*". IEEE. 2000.

[3] Bauman, R.; Clavel, R. '*Haptic interface for virtual reality based minimally invasive surgery simulation*'. Proc. IEEE Int. Conf. On Robotics and Automation. 1998. Pages 381 - 386

[4] Carignan, C.R.; Cleary, K.R. "*Closed Loop Force Control for Haptic Simulation of Virtual Environments*". Haptics-e. **1** [2] 2000.

[5] Colgate, E., Grafing, P.; Stanley, M.; Schenkel, G. "*Implementation of Stiff Virtual Walls in Force Reflecting Interfaces*". Proc IEEE Virtual Reality Annual International Symposium (VARAIS), IEEE, New York. 22pp. 1993.

[6] Cohen, J.; Lin, N.; Manocha D.; Ponamgi, N. "*I-COLLIDE: An Interactive an Exact Collision Detection System for Large Scale Environments*". Proc. ACM Interactive 3D Graphics Conference, ACM, New York, pp 189-196. 1995.

[7] Hannaford, B; Ryu J.H.; Kim, Y.S. '*Stable Contro Haptics*' in 'Touch in Virtual Environments. Haptics and the design of interactive Systems'. Ed. McLaughlin, M.L.; Hespanha, J.P.; Sukhatme, G.S. New Jersey: IMCS. 2001 p. 47 – 70.

[8] Hogan, N. "*Impedance Control: An Approach to Manipulation: Part I-II-III*". Journal of Dynamic Syst., Measure, and Cont., 107(1):1-24. 1985.

[9] Hogan, N. "*Controlling Impedance at the Man/Machine Interface*". Proc. IEEE International Conference on Robotics and Automation, IEEE, New York, pp 1626-1631. 1989.

[10] Immersion Corporation. *FeelItTM Mouse*. Technical Document, San Jose, CA, USA. 12 pp., October 1, 1997. Electronic version: www.immerse.com.

[11] Massie, T.; Salisbury K. '*The Phantom Haptic Interface: A device for Probing Virtual Objects*'. Proceedings of ASME WAM, DSC-Vol. 55-1, 1994, pp295-300.

[12] McLaughlin, M.L.; Hespanha, J.P.; Sukhatme G.S. (ed.). "*Touch in Virtual Environments: Haptics and the design of Interactive Systems*". IMSC Press Multimedia Series, Andrew Tescher, Series Editor. Prentice Hall PTR, NJ, USA. 2002.

[13] Ruspini, D., Kolarov, K., "*Robust Haptic Display of Graphical Environments,*" Proc. of The First Phantom User's Group Workshop, September 1996.

[14] Zilles, C.B.; Salisbury, J.K.. '*A Constranint-based God-object Method for Haptic Display*'. Department of Mechanical Engineering. Artificial Inteligence Laboratory. MIT. Cambridge, MA. *en* http://citeseer.nj.nec.com/cache/papers/cs/2516/http:zSzzSzwww.cs.wisc.eduzSz~zilleszSziros.pdf/zilles95constraintbased.pdf. 1995