# NOTE TO USERS

uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Jilin Zhou

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Collaborative Tele-haptic Surgery Simulation

TITRE DE LA THÈSE / TITLE OF THESIS

N. Georganas

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

J. Lang

P. Liu

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

# Collaborative Tele-haptic Surgery Simulation

by

Jilin Zhou

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of
Master of Applied Science

in

Electrical Engineering

The Ottawa-Carleton Institute for
Electrical and Computer Engineering
School of Information Technology and Engineering
University of Ottawa

*Your file Votre référence*
*ISBN: 0-494-11479-7*
*Our file Notre référence*
*ISBN: 0-494-11479-7*

# Abstract

The main thesis objective is to develop a Collaborative Tele-Haptic Surgery Simulation based on a generic system architecture for Collaborative Haptic Audio-visual Virtual Environments (C-HAVE) with integrated haptics and Collaborative Virtual Environment (CVE) object modeling. A haptic device is a kind of human computer interface that can generate tactile and force feedback to the users. Thus, the users can get the feel of 'touch' while manipulating the virtual objects. Three prototypes based on different architectures are quantitatively compared to demonstrate the effects of adding haptics to a task. In the generic system architecture, a Haptic Real-Time Controller (HRTC) is used to compensate the effects of network latency on the haptic coupling.

A collaborative tele-haptic surgery simulation for tracheotomy is designed and implemented, based on the generic architecture. In our simulation, users from different physical locations are coupled together through haptic devices over some networks and operate collaboratively to perform the surgery. In addition, the simulation provides the functionality that a trainer can coach a trainee on how to perform the surgery successfully in a tele-mentoring manner.

This thesis also contains in-depth overviews of virtual reality and haptic interfaces. The stability issue of haptic interaction with a virtual environment is discussed. Some software implemention issues of the virtual reality system are also included.

## Acknowledgements

I would like to thank Professor Nicolas D. Georganas for supervising me throughout the development of this thesis and for his opportunities that he gave me.

I would also like to take this opportunity to thank Dr. Xiaojun Shen for guiding the project from its beginning, and for his suggestion, corrections and help in bringing this thesis to completion.

I would like also to thank my friends and colleagues in the DISCOVER Laboratory for their advice during the development of this thesis. Especially, I would like to acknowledge Francois Malric for his administrative support.

Last, but not least, I would like to thank my family for their endless love and support, which helped me to complete this work.

Jilin Zhou

Ottawa, December 2004

# Acronym

| | |
|---|---|
| A/D | Analog-to-Digital |
| API | Application Programming Interface |
| BIFS | BInary Format for Scene |
| C-HAVE | Collaborative Haptic Audio-visual Virtual Environment |
| COSMOS | COllaborative System based on MPEG-4 Objects and Streams |
| CVE | Collaborative Virtual Environment |
| D/A | Digital-to-Analog |
| DDM | Data Distribution Management |
| DMIF | Delivery Multimedia Integrated Framework |
| EAI | External Application Interface |
| FSM | Finite State Machine |
| GUI | Graphic User Interface |
| HCG | Haptics Collaborative Group |
| HCI | Human Computer Interface |
| HIP | Haptic Interaction Point |
| HLA | High Level Architecture |
| HMD | Head Mounted Display |
| HRTC | Haptic Real Time Controller |
| IHIP | Ideal Haptic Interaction Point |
| JNI | Java Native Interface |
| MPEG | Motion Picture Expert Group |

| | |
|---|---|
| PCI | Peripheral Component Interconnect |
| PID | Proportional-Integral-Derivative |
| RTI | Run Time Infrastructure |
| SCTP | Synchronous Collaboration Transport Protocol |
| SIRR | Synchronized Interaction Request Resolving |
| TSCC | Tele-Surgery Center Controller |
| UDP | User Datagram Protocol |
| VE | Virtual Environment |
| VM | Virtual Machine |
| VR | Virtual Reality |
| VRML | Virtual Reality Modeling Language |

# Table of Figures

# Table of Contents

# Chapter 1

# Introduction

## 1.1 General

With the advances in computer technologies and human computer interfaces, a new medium to convey and express people's ideas, namely Virtual Reality (VR), has attracted a lot of attention. A VR system is an interactive computer simulation which senses the physical world stimuli, especially participants' positions and actions, and gives one the feeling of being mentally immersed in the simulation (Virtual World) by delivering feedback to one or more modalities [1]. Particularly, the implementation of various collaborative virtual environments has been very intensive in the past few years. Their feature of allowing users to experience and interact with a life-like virtual environment makes them suitable for a wide range of applications, including scientific visualization, entertainment, hands-on training, e-commerce, education and many more. This thesis aims at the development of a collaborative tele-haptic surgery simulation for a tracheotomy operation. It is based on a heterogeneous scalable architecture proposed for Collaborative Haptic Audio-visual Virtual Environments (C-HAVE).

## 1.2 Background

All the programs for the implementations of our design are written in Java. In particular, the Java 3D Application Programming Interface (API) is used for the graphics rendering

in the application. Since the driver of haptic device is written in C++, Java Native Interface (JNI) is used to access those native functions. The Virtual Reality Modeling Language (VRML) is used to describe the 3-D virtual environment model.

### 1.2.1 Java 3D API Overview

The Java 3D API is a scenegraph API developed at SUN Microsystems and used for rendering interactive 3D graphics with the Java programming language. In Java 3D, the native rendering of the 3D scene is based on OpenGL or DirectX, while its description, application logic, and scene interactions reside in Java. In contrast to OpenGL which describes the 3D scene in the level of points, lines, and triangles, Java 3D can describe a scene as a collection of objects. All the objects are connected to form a scene with a high-level scene description model, the scenegraph, which allows scenes to be easily described, transformed, and reused [2]. Hence, it gives developers a high level control of an interactive 3D virtual world. With the Java 3D API, the user can develop a very rich and interactive 3D virtual world suitable for many applications.

A scenegraph is a hierarchical data structure that keeps the spatial relationships between objects. A scenegraph description also enables the developers to reuse their design. Figure 1 shows the scenegraph for a simple application. Currently there are three basic classes of objects within the scenegraph [2]:

- Management nodes to manage a collection of child objects: Locale, BranchGroup, TransformGroup, ViewPlatform, and so forth.
- Geometry nodes to define visible geometries in the virtual world: Shape3D, Background, and so forth.

- Control or influence nodes to define application behaviour of the virtual world: Behaviours, Light, Sound, and so forth.

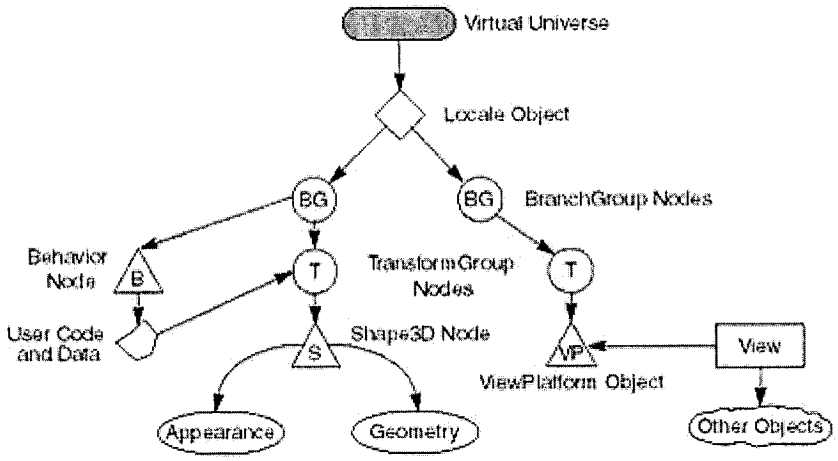



Figure 1: Java 3D Scenegraph [2]

In Java 3D, the virtual world is called the VirtualUniverse, the top-level class that contains all the elements that you define within your virtual world. The VirtualUniverse contains not only geometric objects but also non-geometric objects which are used to control or influence the world. A VirtualUniverse object contains at least one Locale object. A Locale defines a geographical region within your scenegraph using high-resolution coordinates. The high-resolution coordinates serve as the origin for all scene graph objects contained within that Locale. A BranchGroup object is the root of every sub-graph. A TransformGoup object is used to control the position and the orientation of its children nodes in the virtual universe. A geometrical object in the scene is defined as a Shape3D object which is based on a Geometry object and an Appearance object. The Appearance object sets the rendering attributes for the geometric primitives in the scene – at start-up, during scenegraph creation, or dynamically at runtime [3].

In Java 3D, there are two distinct branches within the scenegraph: scene and view. The view branch is responsible for rendering the scene side of the scenegraph. A

ViewPlatform object is used to control the position, orientation and scale of the viewer. The View objects specify some view parameters such as field of view angle and clipping distance which are needed to render the scene graph. An avatar, which is a 3D abstraction of the user in the virtual world, can be combined with the View object.

### 1.2.2 Java Native Interface Overview

The Java Native Interface (JNI) is the native programming interface for Java. The JNI allows Java code that runs within a Java Virtual Machine (VM) to operate with applications and libraries written in other languages, such as C, C++, and assembly [4].

Programmers use the JNI to write native methods to handle those situations when an application cannot be written entirely in the Java programming language. For example, in haptic virtual environment applications, the libraries of the haptic device are mostly written in C or C++. Or we want to implement a portion of code in C or C++ to satisfy real time constraint. In either situation, we have to use JNI to interface these drivers or codes, if we choose Java as the programming language.

On the other hand, the JNI framework also allows the native method utilizing Java objects in the same way. A native method can create Java objects such as strings and arrays. A native method can also receive and use objects created by Java application code. Thus, both the native language side and the Java side of an application can create, update, and access Java objects and then share these objects between them [4]. Figure 2 shows the relationship among the Java side application, JNI, and native code side. It is easy to see that the JNI serves as the glue between Java and native applications.

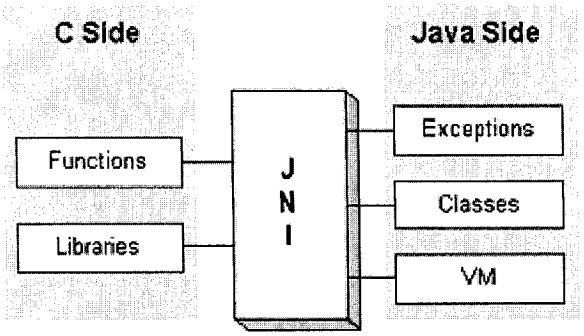



Figure 2: JNI Relationship with Java and C Language [4]

### 1.2.3 Virtual Reality Modeling Language Overview

The Virtual Reality Modeling Language (VRML) is a file format for describing interactive 3D objects and worlds. VRML is not a real programming language as C or Java, but rather a make-up language. The Programmer just describes how the scene should look like. The VRML browser will create the actual 3D models in real time, when the VRML file is loaded. The earliest standard, VRML 1.0, was very successful as a file format for the description of 3D objects and worlds, but it did not support user interaction with objects and worlds, as well as animation of the objects. Then VRML 2.0 was created to extend the capabilities of VRML 1.0 by including the user interaction and object animation. The current version of the language, VRML 97, became a standard of the International Organization for Standardization (ISO) in December 1997. VRML 97 consists of two parts. Part 1 (ISO/IEC 14772-1) defines the base functionality and text encoding for VRML. Part 2 (ISO/IEC FDIS 14772-2) defines the base functionality and all bindings for the VRML External Authoring Interface [5].

With VRML, a scene is made up of nodes that contain objects and some other control elements. A node is defined as an individual object or operation in a 3D scene. For example, a node can represent a shape like a sphere or a box. The properties of the

objects such as shape or color are described with the values stored in the fields of the nodes. VRML 97 defines 54 different node types, including geometry primitives, appearance properties, and some grouping nodes. Also, it defines 20 different types of fields that can be used to store data. Finally, all the nodes that build up the 3D world are combined in a scene with a tree-like structure. The tree describes the data-structure of the 3D world and how the objects relate to each other in the spatial domain. This hierarchical scene graph structure is the main feature of VRML.

A VRML file may contain a list of ROUTE statements. This is used to trigger an event or message-passing mechanism by which nodes in the scene graph can communicate with each other. User interaction is mainly achieved through a set of nodes called sensors. In addition, the previous defined nodes can be reused with the DEF and USE statements. VRML also allows distributed scenes with the Inline node where the URL of a new scene is defined in one of its fields.

Another very important feature of VRML is that it allows the access to the nodes in the scene graph from external environments through its External Application Interface (EAI) [6]. The EAI forms this ability by giving access to the top-level named nodes in the scene graph to the external environment. All these features enable VRML to be used in a variety of application areas, such as engineering and scientific visualization, multimedia presentations, entertainment, and shared virtual worlds [5].

### 1.3 Problem Statement

In a collaborative haptic audio-visual virtual environment, participants are connected together through some networks. A haptic device is a kind of Human Computer Interface (HCI), which can generate force or tactile feedback to the users. To get a realistic sense

of touch, haptic rendering, the process of generating force feedback to the users in response to the interactions with the environment, typically requires a 1 KHz update rate. Since all the participants share the same virtual environment and may collaboratively perform a specific task, the manipulation on the same virtual objects by users at different sites might result in diverging representations [7]. Moreover, if the haptic information needs to be sent over the network, the unpredictable network latency may significantly perturb the stability of force feedback by affecting the effective haptic rendering and degrading the sense of immersion. Thus, both the network latency and system architecture affects the overall application performance. Therefore, the first objective of the thesis is to find a system architecture which can compensate a certain amount of network latency and guarantee the synchronization of virtual environment state at each site.

The second objective of the thesis is to develop a collaborative tele-haptic surgery simulation based on the proposed architecture. Haptic surgical simulations have been a hot topic in the virtual environments research recently. Compared with traditional medical education approaches where surgeons build their skills from practice on animals, cadavers and patients, haptic surgical simulation has several advantages [8]. The first benefit is that surgeons do not need to practice on patients, which might endanger their lives. The second benefit would be the reproducibility: a procedure can be practiced again and again at any time. Finally, the cost of such simulations is comparatively low as compared to a real operation.

## 1.4 Proposed Solutions

To meet the requirement of the first objective, we experimentally compared a generic architecture for C-HAVE against the conventional client/server architecture. The generic architecture is based on the HLA/RTI (High-Level Architecture/ Real-Time Infrastructure) IEEE 1516 standard for Distributed Simulations [9] [10], which provides scalability and general federation/federate and object management services. A Haptic Real-Time Controller (HRTC), which is based on a real-time operating system, is used to offload the compensation for the underlying network latency from the host computer. In order to get objective measurement of this architecture, three prototypes of implementation are developed to demonstrate quantitatively the effects of introducing haptics to a task. A bilateral tele-haptics testbed is designed and implemented for the sluggish response and task–oriented evaluation. The experimental results show that in a C-HAVE environment the performance of the generic architecture is better than that of client/server architecture.

After the evaluation of different architectures for C-HAVE, we implemented a collaborative tele-haptic surgery simulation, tracheotomy, based on the generic architecture. Java is used to develop the whole application. Since the tactile sense is particularly important for a surgeon in a real operation, a six-degree freedom haptic device is chosen to simulate the sense of touch. With a general tool-object haptic rendering technique, the user can feel both force and torque feedback from the simulation.

Jave 3D API is responsible for the graphics rendering. The virtual surgery room was firstly modeled with VRML and then loaded into Java 3D with a VRML loader. Since the

haptic device API is given in C++, JNI is used to provide communication between the device and the application programs.

The procedure of tracheotomy is described with a Finite State Machine (FSM). A center state machine controller is designed to coordinate each component of the system. TiDeC® from HandShake VR Inc. is used as the haptic real-time controller in our system, which not only provides the network time delay compensation but also includes a tele-mentoring functionality.

### 1.5 Contribution of the Thesis

Different system architectures for C-HAVE are analyzed in the thesis. In order to get objective measurements of these architectures, three prototypes are implemented to quantitatively demonstrate the effects of introducing haptics in a collaborative virtual environment application. A bilateral tele-haptics testing pattern is designed for sluggish response and task –oriented evaluation.

Based on the selected architecture, a collaborative tele-haptic surgery simulation is developed. In this simulation, there will be two scenarios. In the first, two surgeons operate collaboratively from different locations. In the second scenario, the trainer coaches the trainee on how to perform the surgery successfully in a tele-mentoring manner.

### 1.6 Structure of the Thesis

The rest of the thesis is organized as follows:

Chapter 2 provides an overview of the virtual reality system and focuses on the collaborative haptic audio-visual virtual environment.

Chapter 3 discusses haptic interface issues which include the haptic rendering and the stability of haptic interaction with a virtual world. As the haptic device used in our design, the MPB Freedom 6S hand controller is presented in details.

Chapter 4 discusses in details the effects of adding a haptic device in a collaborative virtual environment. The chapter focuses on the analysis of different system architectures for the collaborative haptic audio-visual virtual environment (C-HAVE). A system architecture using a haptic real time controller (HRTC) is discussed in great detail and experimentally compared with other two architectures.

Chapter 5 focuses on the implementation of the collaborative tele-haptic surgery simulation using the selected architecture.

The last chapter, Chapter 6, summarizes the results of the thesis and discusses possible future research.

### 1.7 Publications arising from this thesis

Two papers arising from this thesis are as follows:

- J. Zhou, X. Shen, and N. D. Georganas, "Haptic Tele-surgery Simulation" , *Proc. IEEE HAVE, 2004*, Ottawa, October 2004
- X. Shen, J. Zhou, A. El Saddik, and N.D. Georganas, "Architecture and Evaluation of Tele-Haptic Environments", *Proc. IEEE DS-RT 2004*, Budapest, October 2004

# Chapter 2

# Virtual Reality

## 2.1 Introduction to Virtual Reality

The term 'Virtual Reality', was initially referred to 'Immersive Virtual Reality' in 1980s. In an immersive VR, the user is fully immersed in a three-dimensional world completely generated by computers. Webster's Dictionary defines the *virtual* as "being in essence or effect, but not in fact" and the *reality* as "the state or quality of being real" [11]. If we combine them together, it means "something virtually existing". Fleming stated "A thing has a virtual existence when it has all the conditions necessary to its actual existence" [12]. However, the most accepted definition of VR so far is "Virtual Reality is a way for humans to visualize, manipulate and interact with computers and extremely complex data" [13]. The visualization here not only refers to the visual output of a virtual world but also to computer generated auditory or some other sensual outputs such as force and tactile feedback. With some human computer interfaces, users are able to interact with the computer generated virtual worlds and directly manipulate objects within them. The term 'Virtual Environment (VE)' is often used as a synonym for virtual reality. Figure 3 represents a generalized VR system.

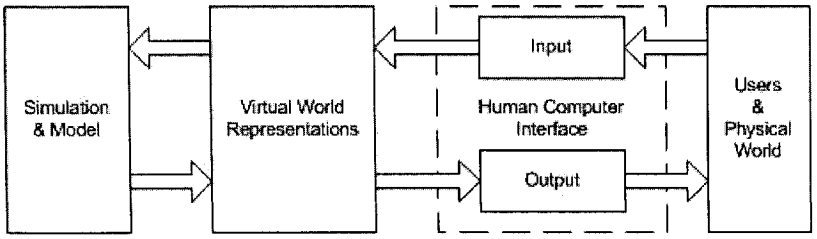



Figure 3: Virtual Reality System

Typically, there are four key elements in virtual reality: virtual world, immersion, sensory feedback, and interactivity [1]. A virtual world is a computer generated space which can be manipulated by participants in an interactive way. The objects inside the virtual world may have a concrete form, physical properties, texture, or sound attached. Generally there are some dynamic rules used to describe and manage the behavior of the whole virtual world.

Immersion, whether physiological or psychological in nature, is a sensation of being in an environment. 'Presence' is an alternative word to describe it. This sense of actual presence in a virtual environment is one of the most fascinating features of virtual reality. It is achieved, to greater or lesser extents, depending upon the goals of the particular application and the design of the user interface. Normally, achievable immersion is proportional to the cost and technical complexity of the application.

In a VR system, sensory feedbacks typically refer the visual, audio, and touch signals. The touch feedback includes both force and tactile. Visual and audio senses receive the feedback from computer controlled displays and speakers, while the sense of touch is provide by a kind of bi-directional human computer interfaces called haptics (Chapter 3). In order to correctly generate the sensory outputs, the system must track the movements of the participants. For example, in an immersive VR system, one player wears a Head

Mounted Display (HMD) to navigate the virtual world. If the player turns his/her head around, the image showed on the HMD visual display should be synchronized with the rotation. In this case, the orientation of the player's head must be captured and passed to the application program.

The last key element of VR is its interactivity. If the display of a virtual world does not respond at least to a user's physical movement, it should not be considered virtual reality [1]. The interactivity of a VR system is mainly expressed in terms of its abilities to navigate the virtual world and manipulate the virtual objects. In a multi-user VR system, it may also include how the participants communicate with each other. Obviously, how we interact with the virtual world is fully dependent on the requirements of a particular application.

The example applications for virtual reality range a wide spectrum. In recent years, the major application fields that show the most encouraging growth tendencies are scientific visualization and representation, architectural modeling, training and education, and industry manufacturing. Some examples for these applications could be found at [14] [15] [16] and [17].

### 2.2 Types of VR Systems

Based on the different rules, VR systems can be categorized in different ways. Typically it is categorized into non-immersive VR system, semi-immersive VR system, and fully immersion VR system, based on the sense of immersion or the degrees of presence the system can provide. Immersion or presence refers the sensation of being in an environment. Immersion presence is generally believed to be a function of several

parameters, including level of interactivity, image complexity, stereoscopic view, and the update rate of the display [18].

**Non-Immersive (Desktop) Systems**

Non-immersive systems are the least immersive implementation of VR techniques. Most of them use conventional computer monitor to display the visual output. Interaction with the virtual environment is done through conventional means such as keyboards, mice, and data gloves. The advantage of non-immersive system is its low cost since there is no need to purchase special hardware.

**Semi-Immersive VR Systems**

Semi-immersive systems are a relatively new implementation of VR technology. A semi-immersive system will typically comprise of a relatively high performance graphics computing system coupled with a large screen monitor, a large screen projector system, or multiple television projection systems [18]. Even without these higher performance graphics stations, a VR system can still be considered as semi-immersive, if some force display devices are included. Apparently, the costs of semi-immersive systems are higher than that of non-immersive ones.

**Fully Immersive VR Systems**

The fully immersive VR systems completely immerse the user's personal viewpoint inside the virtual world. The participant usually wears a HMD to get the direct experience of virtual world. HMD is a helmet which holds the visual and auditory displays (Figure 4). Another kind of immersive systems is to use multiple large projection displays to create a Cave-like room in which the viewers can experience. Compared with the other two VR

systems, fully immersive VR systems obviously have the highest requirements on both hardware and software to achieve a satisfactory degree of presence.

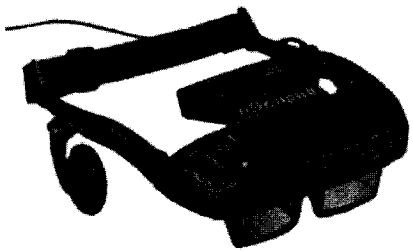


Figure 4: i-Scape II Head Mounted Display [19]

Sometimes it is not easy to decide whether a VR system is immersive, semi-immersive or non-immersive system, since the creative use of display and auditory peripherals can promote the sense of presence, even in the absence of the ability to fully control the virtual environment [20].

### 2.3 Introduction to Collaborative Haptic Audio-visual Virtual Environments

A Collaborative virtual environment refers to the one where multiple users are sharing and interacting with the same virtual world or simulation. The users' representations are referred to as their avatars. Avatar is a virtual object used to represent a participant or a physical object in a virtual world. In a collaborative VE, participants are connected together through some network. Basically they share the same virtual environment and collaboratively perform a specific task. The participants may act in different roles in the application. Some of them may actively interact with the virtual world through different kinds of input devices or collaboratively perform some tasks with other users, while other users may be just passive. For instance, some of the participants in the virtual world may only provide virtual objects as a service to the remaining users or may just be purely

observers. If haptic devices are used as a tool to interact with the virtual world, the sense of touch could be delivered over a network. We term such a scenario Collaborative Haptic Audio-visual Virtual Environment (C-HAVE).

There are so many different haptic devices available in the market today, therefore it is reasonable to assume that in a collaborative Virtual Environment, there may be a heterogeneous assortment of haptic devices. Our focus is on tele-haptic collaboration over a non-dedicated channel (such as an Internet connection, for example) where users experience unbounded communication delays [21] [22] [23] [24] [25]. The final goal of this research is to design and implement a generic architecture that will support collaborative touch in virtual environments.

Technically, a C-HAVE environment consists of a network of distributed nodes. Usually each node corresponds to a work station which runs the VE applications and whose operator is a participant of the shared virtual environment. Figure 5 shows a typical example of C-HAVE environment.

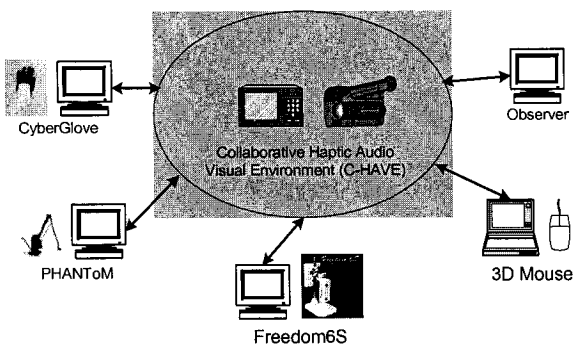



Figure 5: C-HAVE Environment

In the C-HAVE applications, collaboration involves independent or joint manipulation of the virtual objects, or mentoring. Independent manipulation of virtual objects is the

simplest form of interaction with the virtual world. At a given time, an object is uniquely owned by a single participant. The owner may haptically manipulate the object, while other participants may see the owner's manipulation of an object visually in their own displays. But the owner of the object does not receive haptic feedback from those other participants. This is termed unilateral tele-haptics [10]. In contrast to independent manipulation, dependent manipulation is termed bilateral tele-haptics [10], in which multiple participants can haptically interact with the same objects concurrently, so each participant can feel the other's manipulation through the environment modification. Tele-mentoring is a little different concept which allows direct coupling of haptic devices over a network. The slave haptic devices are moved with the movement of the master haptic device. A typical application of tele-mentoring is the training of surgeons, whereby a trainee can feel an expert's guiding hand and follow with it. Tele-mentoring is different from the tele-operation where either forces or position feedback that the remote haptic devices got from the environment are transmitted back to the local haptic device, in turn the local operator get the sense of remote environment.

As mentioned above, unilateral tele-haptics of virtual environments involves multi-users sharing the same virtual environment and may be viewed as a simple integration of conventional CVEs with conventional hapto-visual VEs. Both applications are mostly implemented upon non real time operating systems. By contrast, dependent manipulation and tele-mentoring impose more stringent requirements and, like tele-operations, demand hard real time guarantees.

In a C-HAVE application, the corresponding positions of the haptic device in the VE and the data representing force feedbacks need to be sent over some network. These data

transfers are known as haptic communication. Adding haptics to a conventional CVE creates additional demand for frequent position sampling for collision detection and fast update [7]. The network latency hampers the stability of force feedback by affecting the synchronization of participants and degrading the sense of immersion. Thus, the network latency is a crucial factor that governs whether participants can truly share a haptic experience. Currently, real time transport protocols mainly consider how to reduce the time delay and packet loss when transferring audio and visual media data. Special protocols aiming at minimization of network latency for haptic communication still need to be studied. Obviously, how to deal with network latency to bring haptics into networked applications is a major research issue in C-HAVE.

# Chapter 3

# Haptic Interfaces

## 3.1 Introduction to Haptic Interfaces

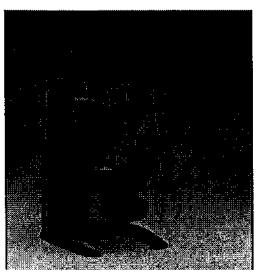
Haptic, from the Greek verb "haptesthai", means "to touch or able to come into contact with". In the virtual reality system, it is the study of the simulation of the touch modality and the related sensory feedback. A Haptic interface is a kind of human computer interface which can generate the sense of touch to the users. Haptic interface, haptic device, or haptic display can be used alternatively.

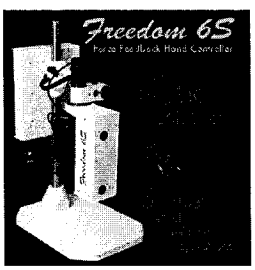
Generally there are two kinds of sensory information related with touch: kinesthetic and tactile. Kinesthetic is the human's perception on the relative movements among body parts and is determined by the movement of the links, therefore the touched object can be reconstructed in the mind by knowing the position of the corresponding links [26]. It is mainly related with the perception of the force feedback from the outside world. Tactile is the sense of touch that comes from some sensitive nerve sensors on the surface of the skin, such as information about pressure and temperature.

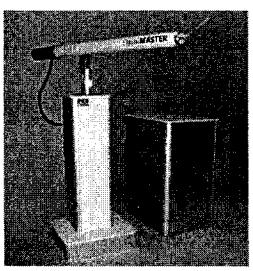
Haptic perception of touch involves both kinds of sensations. Through haptic devices and supporting software, the user can feel the shape, weight, surface texture, temperature, and dynamics of the virtual objects when interacting within the virtual environment.

Compared with visual and audio displays, haptic displays are more difficult to build since the haptic system not only senses the physical world but also affects it [1]. In contrast, we cannot change the object by listening or just looking at it, except in a magical world. In the haptic device world, tactile haptic devices provide the sense of sensing the temperature and pressure, grasping, and feeling the surface textures in details of an object While force haptic devices provide the users with the sense of general shape, coarse texture and the position of an object. Since the tactile sensing is naturally more complex than kinesthetic sensing, a tactile haptic device is more difficult to build than a kinesthetic one. Due to technical challenges, there is currently no single haptic device that could combine these two sensations together, although we cannot separate them in the real life at all. In this thesis, if not specifically mentioned, we consider force haptic devices that only provide sense of kinesthetic.

A Haptic interface has been applied into many VR applications such as surgery training, industrial CAD design, military simulations and so on. It is found that the haptic interface is extremely effective when the VR application involves manual tasks. Figure 6 shows some typical haptic devices available on the market.





(a)Phantom® Desktop™ [27] (b)MPB 6S Hand Controller [28] (c) Haptic Master [29]

Figure 6: Typical Haptic Devices Examples

In the following sections, the haptic rendering and its implementation is first discussed. Then, the stability issue of haptic interaction with VE is discussed. Following that, the haptic device used in this thesis, MPB Freedom 6S Feedback Hand Control, is also discussed in detail.

### 3.2 Haptic Rendering and its Implementation

Basically, haptic rendering refers to the process by which desired sensory stimuli are imposed on the user to convey information about a virtual object using a haptic device [30]. This process typically consists of two parts: collision detection and collision response (Figure 7). Collision detection checks whether a collision happens between the haptic probe and the virtual objects. Collision response computes the force and torque feedback with pre-defined algorithms and the state of collision detection.

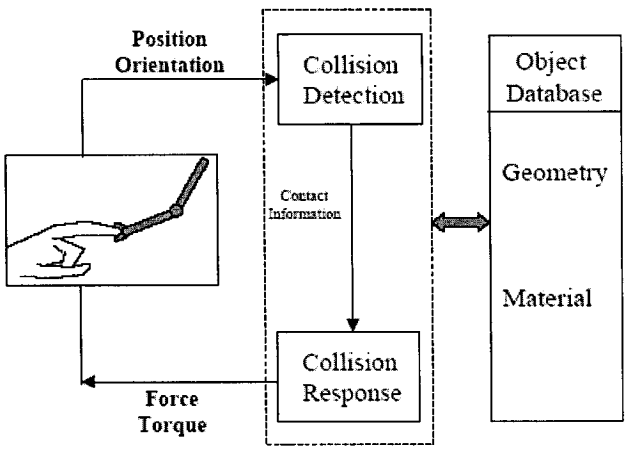



Figure 7: Haptic Rendering Diagram [31]

Without loss of generality, a simple example to explain the concept is the haptic rendering of a sphere located at the origin of the virtual world [31] (Figure 8).

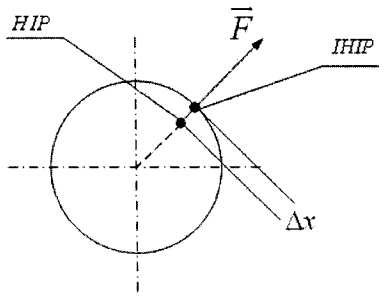



Figure 8: Haptic Rendering of a Sphere

Let us assume that the user interacts with the virtual sphere only with the end tip of the haptic probe known as Haptic Interaction Point (HIP) and the surface of the sphere is frictionless. The corresponding surface contact point is called Ideal Haptic Interaction Point (IHIP) by researchers [31]. If the end tip of the probe is outside of sphere, there is no force delivered to the user, in turn the user can freely move the probe. During this time, the HIP moves together with the IHIP. Once the probe penetrates into the sphere, collision happens and the collision response computes feedback forces to push the HIP outside of sphere. In this case, the generated force tries to bring the HIP back to the position of IHIP. Therefore, the shape of the object, sphere, can be felt. Generally the Hook's law is used to calculate this force:

$$F=k\Delta x$$

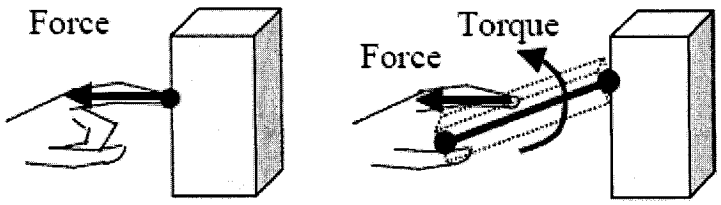
where $k$ is the stiffness of the sphere and $\Delta x$ is the penetration depth. Some researchers have recognized that by adding a viscous damping term to the above equation can greatly enhance the user's perception of a hard surface [32]. Damping is an important component of real world and reflects the energy absorbing quality of the object surface. Now the equation becomes:

$$F = k\Delta x + bv$$

where $b$ is the damping coefficient and $v$ is the velocity of the probe.

This example very well serves an explanation of the basics of haptic rendering, but in most real applications haptic rendering will not be that simple. For instance, the virtual environments may contain any numbers of objects with different shape, so the real time collision detection between the probe and the virtual objects will be very challenging. Although collision detection has been extensively studied in computer graphics [33] [34] e. g., the requirements for information about how the collision occurs and how it evolves over time for accurately computing the interaction forces limit their applications in haptic rendering [35] [36].

In terms of interaction with objects, haptic rendering is broadly categorized into two classes: point based rendering, and ray based rendering (Figure 9). In the point based haptic rendering, only the end tip of probe interacts with virtual objects (Figure 9.a). The sphere example showed above is a point based haptic rendering. Generally, the point based rendering is good enough to perceive the shape and coarse texture of virtual objects, but it is definitely not competent to simulate more general tool-object interactions. For example, it is very likely that the probe collides with two or more objects in the virtual environment at the same time. Hence, both force and torque feedbacks are required to give the user the same experience in the real world (Figure 9.b). This kind of rendering is termed ray based haptic rendering in which the haptic probe is imagined as a line segment and both the position and orientation are used for the collision detection and collision response.




(a) (b)

Figure 9: Point-based Rendering vs. Ray-based Rendering [31]

As more and more haptic devices are adopted in different VR applications, the studies of haptic rendering have been split into some sub-areas. They may include [31]:

- How to render surface details of the virtual object such as shape around the corner, friction characteristics, and textures so on [37].
- How to render deformable objects.
- How to render dynamic objects.
- How to make haptic interaction between network users in a collaborative virtual environment scenario [7].

### 3.3 Stable Haptic Interaction with Virtual Environments

A haptic device links a human operator with a virtual environment together, where the operator feels the objects in the virtual scene with the sense of touch. There is physical energy flowing to and from the operator. Since in the haptic interaction, the physical haptic device generates force feedback, instability of system can damage the hardware or physically hurt the operator. For example, the HapticMASTER from FCS Robotics can easily generate a few hundred Newtons' force in less than one second which is enough to seriously hurt a person [30]. Instability also destroys the illusion of a real object immediately. Furthermore, the instabilities degrade transparency in haptic simulation. A

transparent haptic interface should be able to emulate any environment, from free space to infinitely stiff obstacles [38]. Thus, stability is a very important issue in haptic rendering.

Many researchers have studied the stability issues in haptic simulation in different ways. In a typical haptic simulation, there are at least three components: human operator, a haptic interface, and a computer model of the virtual environment. Figure 10 describes their relationship using the two-port network model. The star superscript indicates the variable is discrete. Typically the haptic simulation can be classified into two categories: impedance control and admittance control. Impedance control means "measure position and display force", while admittance control means "measure force and display motion or position" [39]. All these three components affect the stability of the system.

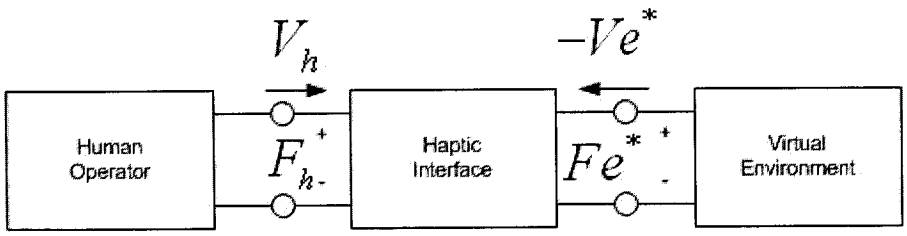



Figure 10: Haptic Simulation Diagram [40]

At the standpoint of control theory, we need to know the transfer function for each component of the system in order to study its stability. Also the sampling rate and the sample-and-hold effect of the discrete variables have to be taken into consideration. Once all this information is known, we can derive the characteristic function for the system and find out the conditions for the stability with the well known criterion: A linear two-port network with a given terminal impedance is stable if and only if the roots of corresponding characteristic equation are in the left half of s-plane for the continuous

system, and inside of the unit circle of the z-plane for the discrete system. However, in the haptic simulation, it is difficult to predict the level of human operator or virtual environment impedance. A linear two-port is considered absolutely stable if there is no set of passive terminating one-port impedances for which the system is unstable [40].

It is generally reasonable to assume that the human operator is passive, if the system he/she is interacting with is passive [40]. Actually, sometimes the human operator can help to stabilize the system by absorbing some energy from the haptic interface. However, if the human operator's own physical characteristics are considered in the control loop, the problem will become very complicated. The measurements showed that human elbow stiffness might vary from a minimum of $1.4 Nm/rad$ to a maximum of $400 Nm/rad$ [41]. At least a fourth order model needs to be used to model it properly [42]. For the virtual environment, if the simulation of physical effect is based on the conservation laws of physics, we can assume the virtual environment is passive. The experimental results from some researchers have shown such assumption is reasonable [43] [44]. Actually most of researchers did their theoretical derivation and experiments based on these two assumptions.

Minsky et. al. derived an expression for the guaranteed stable haptic interaction using the impedance control theory and then modified it with the experimental results [45]. The impedance here is used to describe a generalized relationship between force and motion of the haptic device. They noted a critical tradeoff between sampling rate, virtual wall stiffness, and device viscosity and analyzed the role of the human operator in system stability. In the experiment, they tried to simulate a virtual wall with stiffness $K$ and viscosity $B$. The derived equation for a guaranteed stability is given as follows:

$$B+b > \frac{KT}{2},$$

where $T$ is sampling time, and $b$ is the device viscosity. The results showed that the rate to sample the haptic's position and output the force to the haptic device is the most crucial factor that affects how the highest stiffness can be achieved while maintaining a stable system. The lower the sampling rate, the lower stiffness can be achieved. At the same time, the device viscosity plays a role in the stability issue. The higher the device viscosity, the higher stiffness can be obtained. Thus, it is possible to make the system stable by adding extra viscosity, or by reducing the stiffness of the simulated hard surface. But the human will feel resistance and sluggishness, even in free space, if the viscosity is too high.

Colgate et. al. introduced an artificial virtual coupling network between haptic interface and virtual environment to study the stability (Figure 11) [46]. Modeled as a spring and a damper, the coupling network defines the force feedback reflected to the user based on positions and velocities of both the user and the virtual object (Figure 12). They established an expression similar to Minsky's that guarantees the passivity of a 1DOF haptic interface interacting with a static wall, with the advantage that it is not necessary to characterize the human-operator impedance.

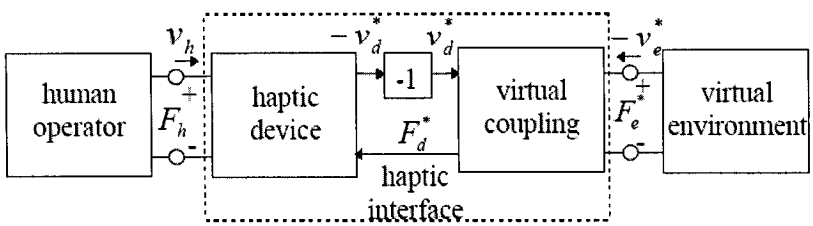



Figure 11: Haptic Simulation with Virtual Coupling Network [40]

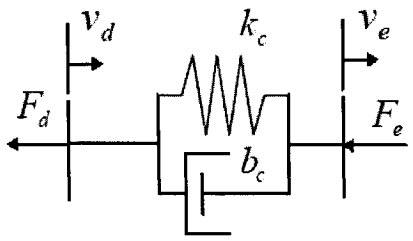



Figure 12: Mechanical Model for Virtual Coupling Network [40]

With these expressions, we can compute the highest achievable stiffness in the virtual environment or adjust those parameters to make the simulation stable.

From the above introduction, we can see all these experiments are conducted based on a one degree freedom haptic device and a static virtual environment. But how to guarantee a stable haptic interaction is still ongoing research, if the virtual environment is dynamic and the haptic device has multiple degree freedoms. The good news is that we not only have the feeling of touch when we interact with the virtual world, but we also get visual, or even audio feedback. These multiple sensory perceptions can reinforce each other and give us the experience of immersion.

### 3. 4 MPB Freedom 6S Feedback Hand Controller

Before we go to the details of one specific haptic device, it is good to know the general characteristics of a haptic device. For every force haptic device, there is a probe which can be held by the users. In a 3 dimensional coordinate system, there are totally 6 degrees of freedom (DOF) in free movement. Therefore, the number of DOF in a haptic device may vary from one to six. For instance, a common 3-DOF device allows three degrees of movement in the translation domain while a 6-DOF device can provide movement in not only spatial translation domain but also the rotation about the x, y, z axes. Haptic devices are also different in spatial resolution, workspace, cost, etc. When designing the haptic

device, Haptic engineers typically employ the following criteria [47]: free space movement must feel free, mass and friction of the device should be as low as possible, solid virtual objects must feel stiff, virtual constraints must not be easily saturated, and the haptic mechanism must be capable of produce enough force so that virtual objects feel solid. Currently, there is no single device that is suitable for all applications.

To test the performance of the haptic device and supporting control software, generally a virtual wall is simulated in 3-D space using a spring and damping model with variable stiffness. By gradually increasing the spring stiffness, record the highest stiffness that can be applied to the virtual wall before the system becomes unstable. Generally, the higher the achievable stiffness, the better the device is.

The MPB F6S is a six-degree of freedom force haptic device developed by MPB Technologies Inc [29]. The name "Freedom 6S" is derived from the number of degrees of freedom offered by the device [48]. The device features high spatial resolution, low friction, and accurate force reproduction with an elliptical working space. Freedom 6S is shown in Figure 6b.

The conceptual drawing of the Freedom 6S translation and distal stages is shown in Figure 13. Motor 0, Motor 1, and Motor 2 are responsible for the translational feedback. These three motors directly drive the respective joints without using belts or gears to minimize the friction and backlash. Joint 3, Joint 4, and Joint 5 provide pitch, yaw and roll feedbacks respectively which are powered by some motors located on the device base. Different from translational motors, each rotary motor drives a tendor loop through a capstan attached to the motor shaft. A spring tensioner behind each motor maintains the

tightness of tendon, so the device stays in the position without any rotation about x, y, or z axis if zero voltage is applied to each rotary motor.

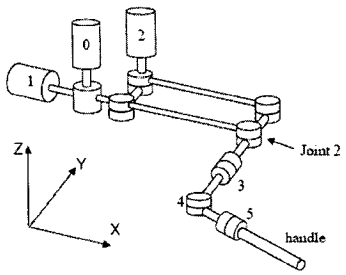



Figure 13: Conceptual Drawing of Freedom 6S [48]

Currently there are six high-resolution sensors bound with the 6 joints to measure the position of end tip of handle and the orientation of handle with respect to the base coordinate system.

The translation workspace (the movement space of the handle) is an elliptical volume $22 \times 24 \times 22\ cm$, which is suitable for the wrist movement. Within this elliptical volume, the distal stage rotates roughly in ±50° of pitch, ±50° of yaw, and ±160° of roll. Static friction in translation and rotation is near 0.06 N and 0.02 Nm respectively. Static inertia is near 200g around Motor 0 and 100g along other two axes. The A/D converter to convert the measured position data to the digital domain has 16 bits resolution, combining with the supported workspace, the Freedom 6S gives resolution $20\mu m$ and $0.2mrad$ for the translation and the rotation respectively. The D/A converter used for converting digital output torque to the analog domain has 16 bits resolution, so potentially total 65,536 different torque values can be sent to motors. Motors in Freedom 6S have different maximum allowed drive torques. The detailed specification of the

device can be found in [49]. The small static friction and inertia play important roles in the stability of device.

The Freedom 6S comes with a very good API. The update of library is very fast and the current version is Freedom 1.4.2 that offers a C language interface. [50] [51] give a detailed explanation of its API. Physically, in a virtual reality application, the Freedom 6S is connected with a control computer via an A/D and D/A card plugged in the PCI slots. If the simulation does not run on the controller computer, a network connection is required between the control computer and the host computer running the simulation. To use the device, a typical application program will perform the following steps:

- Initialize the device.
- Enable the computation of joint velocity according to the readings from the sensors.
- In the application servo loop (a thread dedicated to the interfacing with the device)
    - Get the position and orientation of hand controller
    - Compute the appropriate force and torque feedback in terms of collision status
    - Send the force and torque back to the device
- At the end of program, clean up and stop the device.

Freedom 6S is one of earliest 6 degrees of freedom haptic devices. It can be used in the surgery simulation and space tele-operation. The weaknesses of the Freedom 6s are also obvious. For instance, the tendor will be lightened to incur the slippery problem, thus calibration or tight the tendor manually is needed once a while. The repeatability of motors for some joints is not very good and this makes the motor positioning difficult to get highly accurate. In addition, the workspace is not big enough.

# Chapter 4

# Collaborative Haptic Audio-visual Virtual Environments Architecture Evaluation

## 4.1 Introduction

In early works on CVE applications, and particularly in distributed simulations, scalability, ensuring that the application could support a large number of users, has been one of the main concerns. There has been a shift in this trend, as the virtual reality community came to understand that for a virtual universe it was pointless to worry about supporting tens of thousands of users, if this universe did not succeed in interesting more than ten individuals at the same time. There is not really any concern for scalability in haptic CVEs. Although thousands of users may be taking part in a simulation, there will be typically no more than twenty users engaged in a given collaboration such as repairing a damaged tank or performing surgery on a patient.

This chapter mainly focuses on the effects of adding haptics to a given task and the effects of system architecture on the performance of C-HAVE. The rest of this chapter is organized as follows. The related work about tele-haptic applications and their corresponding mechanisms is analyzed. After analysis of the characteristics of tele-haptic applications, a generic architecture for the C-HAVE environment proposed by Shen et. al

[10] is analyzed in detail. Three prototypes to demonstrate quantitatively the effects of adding haptics to a task are developed. The detrimental effects of time delay on the performance of the task are also described. Then the experimental environments demonstrate the different implementations of the generic architecture presented above. Finally, a summary of the presented research and future work is given.

### 4.2 Related Work

As a research area, tele-haptic applications have been studied for some time. Not taking into account of the actual network configuration, the simplest way to tele-haptics is to deploy two haptic devices onto a single host workstation. [53] describes a tele-haptic collaborative game, where two PHANToM haptic devices placed in two different rooms were linked to a single host computer. Two computer screens were used to display the graphical information to the users, one for each user in the different locations. The screens, attached via a video splitter to the host computer, showed identical views of the virtual environment. [21] and [22] conducted research on the client-server architecture for tele-haptics. In this framework, the server manages the state of the virtual objects based on the position data transmitted from all clients' haptic devices. The server sends the position of virtual objects to each client, so the virtual scene at each client is synchronized. The reaction force reflected to the client is calculated locally, using local haptic data and the updated virtual objects. Obviously, this system is not applicable to a high-demand tele-haptic application due to the round trip propagation delay induced by the client-server architecture. The work in [23] proposed a hybrid architecture for shared haptic virtual environments. There a server is used to handle the request to the application, to synchronize the start and end of the game, and to monitor the functioning of the system.

Whenever the client joins the session, it communicates with the other clients in a peer-to-peer way. Each client has connections with all other clients. The clients contain the complete model of the system, which consists of the dimensions of the virtual environment, the position of all users, and the position and the speed of virtual objects.

The introduction of haptics enables the users to collaborate by transmitting the sense of touch in addition to the two general modalities, sight and hearing. However, when the haptic device is part of a closed-control loop that encompasses the network, instabilities appear [54]. Haptic feedback has been shown in the context of two applications: removing the fuel tank of an aircraft with screwdrivers and wrenches [7], and identifying friends and foes on a monitor using a gearshift like haptic device [54]. Both experiments show the effectiveness of force feedback in terms of the completion time for the task. It has been measured that the time to complete the task with haptic feedback was roughly half the completion time without it. If the haptic device is controlled remotely, or if the haptic device controls another device over the network such as in tele-operation applications, network delay has to be taken into consideration in the feedback of the control loop, since it is very possible that the data obtained are out of date. A good review of tele-operation and different control schemes with time delay can be found in [55] [56]. These works address the issues of closed-loop control of haptic devices over the networks.

## 4.3 Generic Architecture for C-HAVE

### 4.3.1 Generic Architecture

A generic architecture for C-HAVE, implemented over HLA/RTI, the IEEE standard for distributed simulations and modeling [9], is proposed by Shen et. al [10] (Figure 14). Tele-haptic platforms rely on hard real time operating systems to guarantee the stability

of the control loop and to minimize the delays in the network stack, especially when some real time tele-operation control algorithms are implemented with very high computation cost. In the main application program, there are already at least two threads running: one for haptics rendering and the other for graphic rendering. If we can offload the computation for the time delay compensation and tele-operation control from the host computer, the application performance should be better. In this architecture, the HRTC compensates for network latency, so the unreliable latency provided by the host operating system in the network stack may either increase the complexity of the latency compensation algorithms or decrease the effective separation of tele-haptic collaborators.

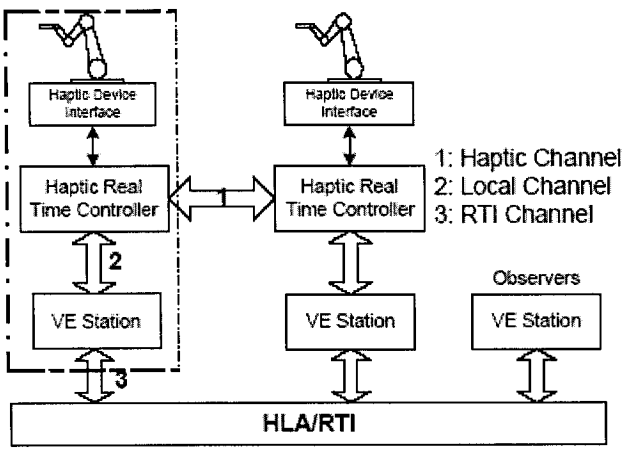



Figure 14: Generic Architecture for C-HAVE

Haptic control and rendering generally require high sampling rates (typically, in the order of 1 KHz) and low latency (typically, in the order of a few milliseconds). Limitations in the current RTI implementations impose considerable end-to-end latency that is not suitable for real time applications [10]. For this reason, [57] introduces the concept of Haptics Collaboration Group (HCG), where the haptics collaborative tasks are performed. One major advantage of the HCG architecture is its scalability to large virtual

communities, where a number of potential users participate. Data information for haptics collaboration is restricted to pass only within HCG groups, where broadband dedicated network connections are provided to transmit updates of objects that require haptics. A node in a C-HAVE environment with a haptic device is comprised of three parts: haptic device, HRTC and VE station. The HRTC controls the haptic device through a device driver. A local HRTC is linked with remote HRTC(s) through a haptic channel (1 in Figure 14) and communicates with its VE station through a local channel (2 in Figure 14). The VE graphics stations are interconnected over HLA/RTI (3 in Figure 14).

The separation of functionalities of haptic and graphic rendering makes the proposed architecture easier to extend to existing applications. As a multi-computer model solution, compared with conventional multi-thread or multi-process approaches for tele-haptics, it applies a hard real time operating system for haptic control, while applying a mainstream OS such as Win2K or WinXP for the application and graphics. It also provides the potential to switch between multiple protocols, one for large-scale distributed simulations and one adapted to collaboration, when several users meet and need to perform a collaborative task, for example, and require an architecture that supports those different protocols. Therefore, HLA/RTI can be used to exchange haptic information between federates (users in a same federation, or users of a common virtual universe). Once two federates have found enough data about each other, they can switch to a side-channel, using a collaboration-oriented protocol to carry haptic update messages [57].

### 4.3.2 Haptic Real Time Controller (HRTC)

The time-varying network delay degrades the performance of many network applications, and can cause instability in applications involving bilateral control of haptic devices.

Besides the obvious requirements of stability in these applications, another major concern has always been the so-called transparency, i.e. the achievement of the ideal situation of direct action of the operator on the remote environment [58] [59]. Typically a HRTC includes some sophisticated algorithms to compensate for the time-varying network delays. In essence, if the compensation were perfect, the time delays would be transparent to the system. The HRTC allows the time delay compensation to vary between full delay compensation which gives higher performance but may introduce more noise and overshoot, to partial delay compensation, to no compensation which results in low performance and instability.

The HRTC in a C-HAVE system provides a modular and flexible platform to allow real time control of a haptic device at one node of a C-HAVE by a haptic device at another node of C-HAVE, which is referred to as client/server mode. Actually, it is modular enough to support multiple clients or other network configurations.

To enable this technology, it is required that the clocks on both nodes be synchronized closely, have precise sampling periods and have the ability to perform complex control computations so that desired performance can be achieved at the client end. An important component of the HRTC is the use of hardware/software solutions to accomplish this synchronization and precision in timing. In order to enable real time control and data exchange, the software of the HRTC runs on a modular and robust real time operating system i.e. Windows CE. UDP is used for the communications protocol.

TiDec®, from Handshake VR Inc., which includes intelligent techniques to compensate for the detracting effects of latency, was chosen as the HRTC in the proposed architecture. The compensation module is located at both the input device node and the destination

device node. This module to provide the destination device with a latency compensated data modifies the network affected haptic data from one user. In addition, the feedback signals are processed by another latency compensation module at the input device node so that the applied haptic effect is consistent with what the destination device is experiencing at a given point in time [54].

### 4.3.3 Virtual Environment Station

The C-HAVE is a shared 3D virtual world supported by HLA/RTI on the Internet. Figure 15 depicts the basic three-layer architecture of the VE station on each haptic node. The top layer is the graphics-rendering layer, where there are two kinds of browser modes from which the user can choose: a VRML-enabled web browser, i.e., Netscape, or a Java3D-based browser.

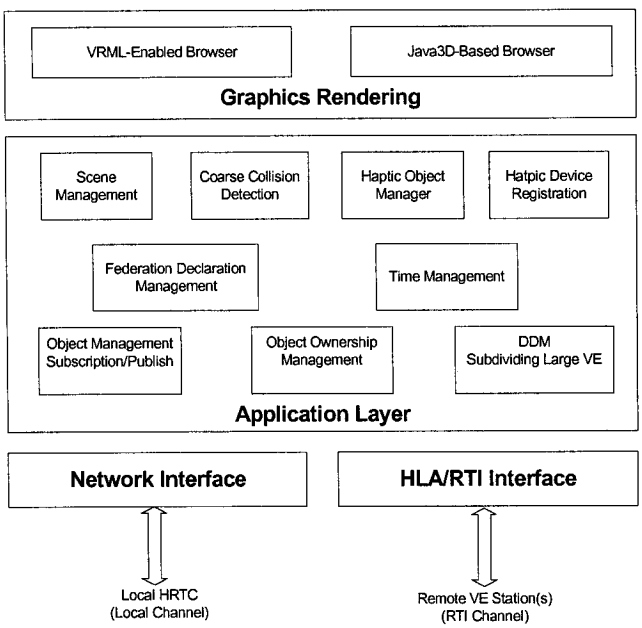



Figure 15: Architecture for VE Station

The application layer (middle layer) plays a pivotal role in the creation of a C-HAVE environment. It not only controls the virtual scenes but also communicates with other

graphics stations through the RTI channel and its local HRTC. The application layer also implements the services that RTI provides, such as Federation Declaration, Object Declaration, Object Ownership Management, Time Management and Data Distribution Management (DDM) [52].

To support heterogeneity in the C-HAVE system, the functionality of registering the haptic devices is provided in the application layer. The Haptic Object Manager receives the status data (output data such as position, orientation) from the haptic device, and sends it to the Scene Manager. HRTC acquires the status data from the device at the rate of 1 KHz through the local channel.

The Scene Manager updates and manages the C-HAVE's state using the data obtained from the RTI channel and the local channel. The object properties (position, rotation, size, weight, etc.) are shared beforehand among all participants.

Coarse Collision Detection detects a potential collision at a reduced, soft real time, using the Haptic Object Manager. The results of Coarse Collision Detection algorithms are predicted collisions between the local. Haptic device and the virtual objects, and are sent to the HRTC through the local channel to provide input to an accurate local collision detection module operating at a 1 kHz hard real time rate.

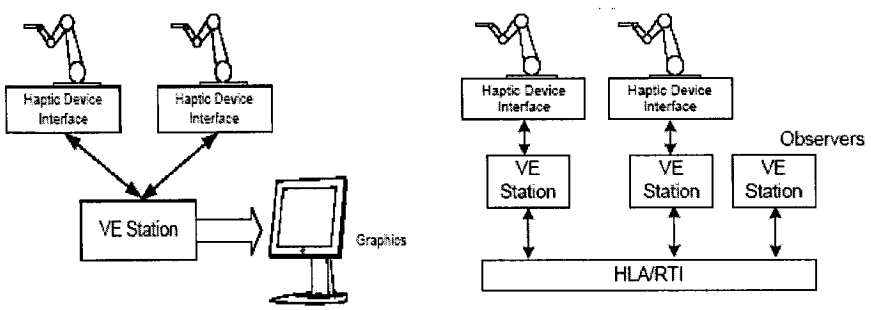
### 4.4 Experimental Environment

The purpose of the evaluation is to verify the effectiveness of network delay compensation. Three prototypes have been constructed to demonstrate quantitatively the effects of adding haptics to a task and then the detrimental effects of time delay on the performance of the task.

The first prototype (Model 1 in Figure 16.a) is actually an implementation of local tele-haptics, where two MPB Freedom 6S controllers are connected to a single VE station with no usage of HRTC and HLA/RTI. In this case, two collaborators share the same computer screen. Since no network delay has been introduced here, it will be the ideal condition of the evaluation.

In the second prototype (Model 2 in Figure 16.b), C-HAVE collaboration is implemented between geographically distributed computers based on HLA/RTI. Both the haptics and graphics are transmitted over RTI. In this case, there might be multiple passive observers with no haptic devices. The goal is to evaluate whether or not, the HLA/RTI is suitable for tele-haptic applications.

The third prototype, Model 3 (Figure 14), is indeed the complete implementation of the discussed architecture, where participants with haptic devices are connected with HandShake TiDec®, and whereas passive observers just have an RTI channel to access the system graphically.




(a) Model 1 (b) Model 2

Figure 16: Prototypes of the Experimental Evaluations

### 4.4.1 Testing Scenarios and Environment

The major goals of collaboration in tele-haptics are to convey one's ideas and concepts to

the other participants, to understand the intentions of all the collaborators and to reach joint decisions and actions. Providing meaningful real-time manipulation of a single object among multiple geographically distant users is a significant research challenge, given the limitations of today's CVE and tele-haptics technology, good solutions will inevitably require significant usage of both network and computer facilities. Knowing in advance that the control of a single shared object is in fact passed back and forth between individual users allows the system to focus its network resources on those objects that might be genuinely manipulated simultaneously. The above is termed bilateral tele-haptics or joint object manipulation in the context of conventional CVE.

In the previous work of [57], three generic approaches for the management of joint manipulation were proposed: 1) constraint-based interaction request resolving; 2) synchronized and 3) non-synchronized interaction request resolving. The second method, synchronized interaction request resolving, is implemented as the means of bilateral tele-haptics in the experiments. The basic ideas of the synchronized approach are described in the following. When multiple users intend to manipulate an object jointly, if two or more interaction connections are established at the same time from different users, interaction requests are no longer processed immediately. Such requests are instead sent to a central server, which stores these requests temporarily, processes and combines them, and sends the reflection updates back to all the participants. This approach is referred to as Synchronized Interaction Request Resolving (SIRR). In SIRR, a central server synchronizes all interaction requests from different users and all the participants have consistent object status after receiving the feedback from the server. Figure 17 is an example of SIRR with two participating sites and one interaction manager. Interaction

requests from participants are sent to the manager site where the execution time is divided into time slots ( $\Delta t$ ) by a specific timer. During the given period $\Delta t$ , the manager will combine all the received requests and respond to the participants. In Figure 17, $t_{init}$ is the delay of interaction initialization, $t_{arrival}$ is network propagation delay and $t_{reflect}$ is the reflection delay on clients.

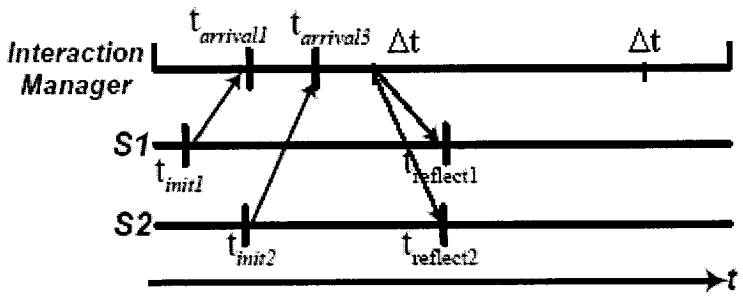



Figure 17: Synchronized Interaction Request Resolving

All the tests were performed on Pentium IV class PCs with dual processors running Windows 2000 and connected to a 100 Mbps Ethernet switch. The graphics were set at 1280X1024 24bit color screen resolution with 3D accelerator cards. All the machines were running typical operating system and networking background processes. MPB Freedom 6S controllers are used as haptic devices in all tests. DMSO RTI NG1.3v6 [9] is applied for the testing of model 2, collaboration over RTI. In the test, $\Delta t$ is fixed to 4 ms and the sampling rate in haptic control loop is set to 1 KHz.

### 4.4.2 Sluggish Response Evaluation

One of the drawbacks of network delay on bilateral tele-haptics is the sluggish response, which is defined as the time interval between the collaborator generating its interaction and the corresponding consequence of the action being "viewed" by the collaborator. Taking the above synchronized interaction approach as an example, there are two

collaborators intended to manipulate a single object such as a cube simultaneously. One of the collaborators, $S1$, takes two roles serving as both the interaction manager and the participant. The other collaborator, S2, is just the client in the bilateral collaboration. Since the computation of the interaction combination occurs on $S1$, there is no network delay effect on sluggish response on $S1$. Obviously, in the worst case scenario for $S1$, the maximum sluggish response time will be $\Delta t$. However, the situation of $S2$ is different. Before $S2$ carries out the corresponding action, the generated interaction will be first sent to $S1$, the interaction manager. After that, $S1$ responds with the combined result. Therefore, the sluggish response time for $S2$ is between $2T_d$ and $2T_d + \Delta t$, where $T_d$ is network delay between $S1$ and $S2$. With the benefits of the time delay compensation module implemented in TiDec®, the interaction manager is able to predict upcoming interactions from the clients before the actual interaction arrives. The predictions are based on the received interactions. Therefore, in an ideal condition, the interaction manager predicts the interactions and therefore does not have to wait for the interactions from $S2$ which are transmitted over the network to arrive at $S1$. The sluggish response time can be minimized up to $T_d$. The purpose of the sluggish response evaluation is to verify the profits that the system gains from the time delay compensation approach.

### 4.4.3 **Task-oriented Evaluation**

In the task-oriented experiment, we evaluate the performance of three prototypes on a given complex task, where a 3D maze game is designed and developed in Java3D. In the maze game, the 3D virtual universe is constructed in physical simulation including gravity, object collision and friction (some screenshots are shown in Figure 18). The cube in the maze is simulated as a rigid body based on the Newton's Second Law:

$$F = \sum ma .$$

If the resultant force from the haptics is larger than the static friction force, the cube starts moving; otherwise it stays stationary. Two collaborators control the spheres by using the MPB Freedom6S manipulators. The given collaborative task for the two participants is to carry the cube from the start position to the destination. Each of the participants is able to move the cube horizontally, but they have to work cooperatively in order to lift the cube assuming that the cube is too heavy for each of them to lift. Upon this task, the total completion time *Tc* is assumed to be a task-oriented evaluation score. A synchronized collaboration approach is applied in this experiment.

### 4.5. Test results

#### 4.5.1 Sluggish Response Evaluation Result

An application is designed and implemented to evaluate the sluggish response time. Figure 18 shows the results of models 1 and 2, and Figure 20 indicates the results of model 3 with various artificially inserted delays. As the ideal case, model 1 implements the *bilateral tele-haptics* on a single host computer to which the two MPB haptic devices are connected. Its sluggish response is negligible, as it is $1ms$.The significant sluggish response time of the HLA/RTI solution, $72ms$, limits the tightly coupled tele-haptic collaborative applications to be built over RTI. The main reason of the large delay of RTI is because the current RTI implementation is using a single thread. In the single-threaded RTI implementation, RTI and the federates share a single thread of execution; and the federates must explicitly pass control to the RTI.

Figure 19 shows that the sluggish response $T_R$ has been improved around 20% to 30% with the time delay compensation. The $T_R$ value varies between $2T_d$ and $2T_d + \Delta t$ in the cases of no compensation. $T_R$ is actually the average of 250 samples during 1 second period. Figure 19 present the specific case of time delay compensation while a delay of $50ms$ is inserted. Because the predications are based on previous received information, initially, the sluggish response is roughly $2T_d + \Delta t$. Upon receiving information, the interaction manager starts to predict the interactions of the clients. The experimental result indicates that when the time is over $100ms$, the sluggish response is improved to $80ms$.

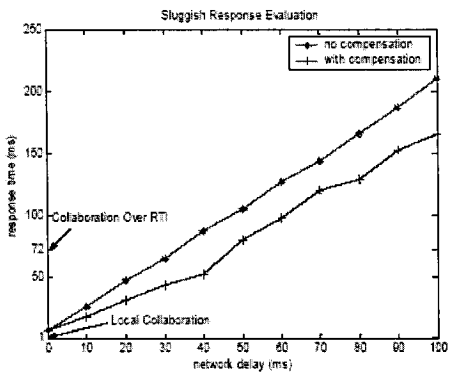



Figure 18: Sluggish Response Evaluation

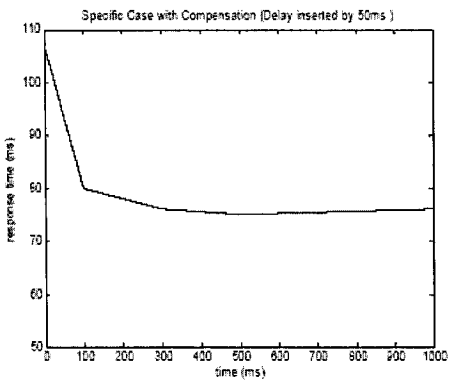



Figure 19: Specific Case with Time Compensation

### 4.5.2 Task-oriented Evaluation Result

In this experiment, we first test the case of model 3 with no inserted delay and no haptic feedback from/to the collaborators. The results show that it takes a very long time to finish the given task. After the haptics is added into the application, the completion time with haptic feedback is roughly 1/3 of the completion time without it. The results also demonstrate that the time delay compensation approach improves the performance in terms of stability felt by the users. Figure 20 shows some snapshots of the experiments.

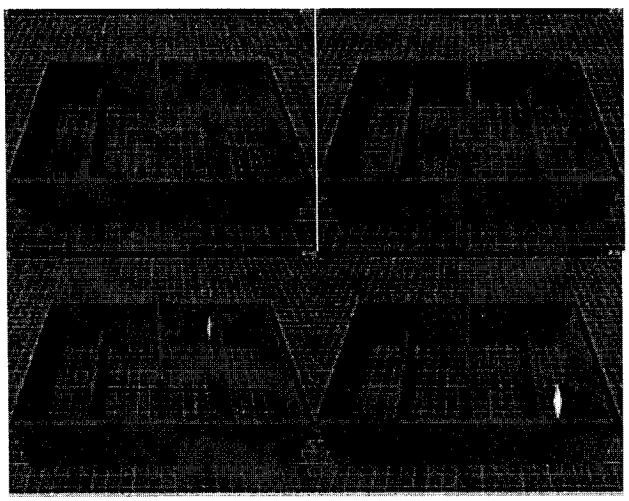


Figure 20: Some Screen Snapshots for the 3D Maze Game

# Chapter 5

# Collaborative Tele-Haptic Surgery Simulation

## 5.1 Introduction

In the last decades, with the rapid development of electronic and computer technologies, advanced medical equipment has been introduced to facilitate medical diagnosis and treatment. Yet, medical education and training approaches have not benefited much from these developments. As the kinds of surgical operations increase, specialized procedures become more and more sophisticated. Thus, the medical specialists have to be well trained before they perform surgery on real human beings.

With the advances in computer technology and haptic devices, haptic surgical simulation has become a hot topic in the VE research field. Compared with traditional medical education approaches where surgeons build their skills from practice on animals, cadavers and patients, haptic surgical simulation has several advantages [8]. The first benefit is that surgeons do not need to practice on patients, which might endanger their lives. The second benefit would be the reproducibility: a procedure can be practiced again and again at any time. Finally, the cost of such simulations is comparatively low.

The purpose of this research is to develop a collaborative haptic simulation prototype for tracheotomy surgery and its training program based on the architecture proposed in chapter 4. In this simulation, there will be two scenarios. In the first, two surgeons

operate collaboratively from different locations. In the second scenario, the trainer surgeon coaches the trainee on how to perform the surgery successfully in a tele-mentoring manner.

In chapter 3, haptic rendering has been discussed in details. In haptic surgical simulations, the haptic probe is represented as surgical instruments with which different surgical operations can be simulated. Since surgery is a very delicate process and actions performed by surgeons largely depend on force feedback, the performance of haptic rendering is the most important criterion in order to evaluate a haptic surgical simulation.

However, it seems there is still a long way to go before surgical simulations can be adopted in medical education, due to their nature of complexity. The complexity lies in the technologies behind simulations, such as computer visualization, 3D deformable objects modeling, collision detection algorithm, and haptic rendering. Although researchers have extensively studied all of these topics, the fact that surgical simulations have to integrate all these technologies together makes research and development very challenging [60]. If the collaborative distributed simulation system itself is demanding, the consideration of variant network latency will make the system even more complex.

The remainder of the chapter is organized as follows. Sec. 5.2 reviews related work about surgical simulation and collaborative VE. Sec. 5.3 details the application scenario of tracheotomy. Sec. 5.4 presents the system architecture. Sec. 5.5 focuses on the implementation of the simulation. Sec. 5.6 describes the tele-mentoring issues. Finally Sec. 5.7 concludes the chapter and points out the possible future improvements for the simulation.

## 5.2 Related Work

Various haptic surgical simulations have been developed recently. Langrana et. al. developed a training simulator for subsurface liver tumours palpation with the Rutgers' Master II haptic device [61]. The tumours are modeled as harder spheres within softer spheres. They applied Finite Element Analysis to compute the force feedback corresponding to graphic deformation. Researchers at Immersion Medical developed CathSim®, a Vascular Access Simulator to train nursing students in starting an IV or drawing blood [62]. The application-specific haptic device gives 3 degrees of freedom (DOF) data and one DOF haptic force feedback along the direction of needle insertion. At Stanford, Balaniuk and Costa developed some fluid-filled deformable objects which are suitable for real time haptic interactive deformation by "cutting" and "suturing" [63]. Machaco et. al. used both haptic devices and stereo view glasses to develop an immersive simulator of bone marrow harvest for transplant [64]. For a detailed survey of surgical simulations the reader is referred to [8].

All the surgical simulations mentioned above are stand-alone haptic applications where no network is involved. For collaborative haptic applications, however, we have to be concerned with two additional problems: how to keep the coherency of virtual scenes among all users and how to get stable force feedback when haptic information is sent over non-dedicated channels, such as an Internet connection, where there is some latency and jitter [7]. SPIDAR was the first successful implementation of a collaborative haptic simulation where two users are simultaneous manipulating the same virtual object [65]. At MIT, Basdogan, Ho, and their colleagues designed an experiment where users at different locations were asked to jointly moving a ring back and forth along a wire

without touching it [66] [67]. The experiments were performed with visual feedback only, and then with some haptic force feedback. In [7], P. Buttolo, R. Oboe, and B. Hannaford discussed the effects of different system architectures on performances of shared haptic VE applications.

### 5.3 Application Scenario

The virtual surgery simulation implemented is tracheotomy. Tracheotomy is a surgical procedure that is usually performed in an operating room with local anesthesia. A tracheotomy is performed if the person cannot breathe on his/her own, or if there is not enough air getting into the lung [68]. A tracheotomy is an incision into the trachea that forms a temporary or permanent opening which is called a tracheotomy. The incision is usually vertical and runs from the second to the fourth tracheal ring. Figure 21 shows the position of the incision made on the trachea.

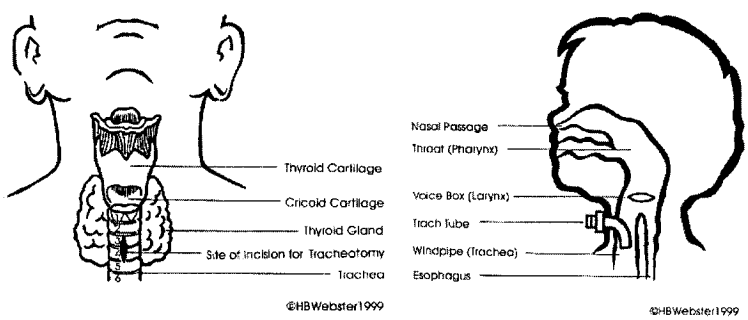



Figure 21: Top and Lateral View of Tracheotomy [69]

A trachea tube is then inserted through the opening to allow passage of air and removal of secretions. Instead of breathing through the nose and mouth, the patients will now breathe through the tracheotomy tube. Figure 21 also shows a lateral view of trachea after surgery.

The procedure for the tracheotomy is very straightforward. There are mainly six steps [70]:

1. Patient lies flat on the back and the neck is hyper extended with a shoulder roll.
2. Surgeon makes an incision on the skin of the neck right above the trachea.
3. Two or four hooks are used to make an opening.
4. The overlying isthmus of the thyroid gland is retracted or divided and the exposed third or fourth tracheal ring is incised.
5. Tracheotomy tube is inserted into trachea.
6. Tracheotomy tube is secured with neck ties.

For more detailed information about tracheotomy, the reader is referred to [71]. From its procedures, we know one surgeon cannot perform the whole surgery alone. Another two persons are needed to hold the hooks while the surgeon cuts the trachea and inserts the tube. We choose the tracheotomy as our application because its procedure is easier compared with other surgeries. But its implementation does have to take account of all the issues of a complex virtual reality system mentioned in previous chapters, such as high quality virtual world modeling, realistic force feedback, interactivity, and network variance.

Without loss of generality, we simplified the procedure of the tracheotomy in our simulation for an easier implementation. A FSM is used to illustrate the simplified procedure. In the simulation, only two hooks are used to pull the incision, therefore two users working together can perform the surgery. The transition diagram of the FSM is as follows:

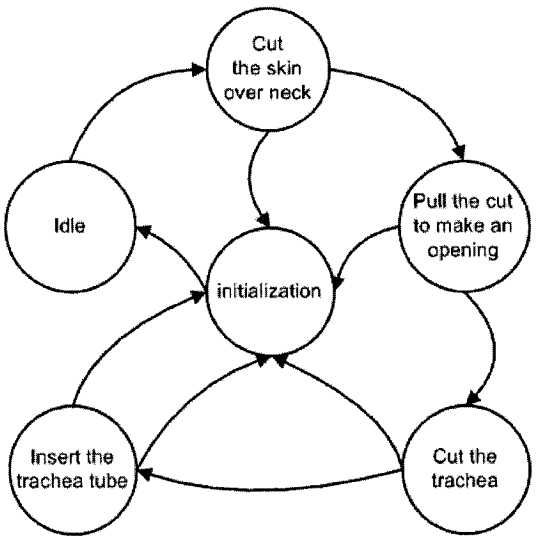



Figure 22: State Transition Diagram for Tracheotomy

The simulation starts in Initialization State in which all necessary flags and variables get reset. The simulation then automatically enters the Idle State and the users can start the surgery. First, one user cuts the skin with a scalpel and then another pulls the cut with two hooks to make an opening. After this, the first user can make a cut on the trachea and then insert the trachea tube. Both users have the right to stop the operation at any moment and the simulation will automatically be reinitialized. If a user cuts a blood vessel defined under the skin, the bleeding event is triggered. Either user has to use a piece of gauze to clean the blood before they can do anything else. For clarity's sake, the bleeding is not put into the state diagram. A State Controller is used to control and coordinate each state's transition.

Our simulation also provides a tele-mentoring mode in which a surgeon-trainer supervises the trainee on how to perform the surgery correctly. In this mode, the trainer controls the movement of the trainee's device through the HRTC. The TiDeC® toolbox from Handshake VR Inc [72] is used for its implementation in our simulation.

### 5.4 Application System Architecture

The generic architecture for C-HAVE proposed in Chapter 4 is adopted for our collaborative tracheotomy simulation. The only difference is that the simulation is based on the MPEG-4 COSMOS framework developed at the DISCOVER Lab [73] but not on RTI/HLA (Figure 14). Since only graphic information is transmitted through this channel, there is no big difference between using these two underlying framework in our tele-surgery application. The MPEG-4 COSMOS framework can be used for efficient distribution and updates of VE scenes in the future. A real-time operating system (Windows CE) based HRTC is used to guarantee a stable haptic control loop and to compensate for the network delays. A local HRTC is connected with a remote HRTC through some haptic channel (1 in Figure 23) and communicates with the VE station through some local channel (2 in Figure 23). The VE Station is mainly responsible for the haptic rendering, the graphic rendering, and the application control. All the VE stations are interconnected together through some COSMOS channels (3 in Figure 23). The following paragraphs explain in more details each of these components.

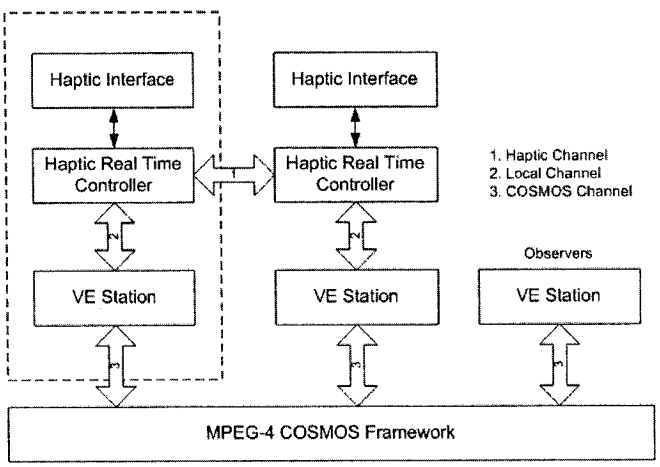



Figure 23: System Architecture for Tracheotomy Simulation

### 5.4.1 MPEG-4 COSMOS Framework

Some VEs' contents may be very diversified. They may include any type of synthetic or natural media objects. To distribute such rich multimedia contents over a network in real time and keep their interactivity at the same time, traditional compression standards are no longer suitable. MPEG-4 is an appropriate IEC/ISO standard for interactive multimedia applications developed by MPEG (Motion Pictures Expert Group) [74]. Its functionalities such as content-based interactivity, universal accessibility and sufficient compression make it very attractive to the collaborative VE applications.

MPEG-4 defines its scene as the composition of audio-visual objects which may be synthetic or natural. The scene is compressed and streamed using MPEG-4 BInary Format for Scene (BIFS). The user may have the possibilities to interact with objects of the scene, e.g. by navigating into 3D worlds, modifying the position of the objects and even collaborating with other users. Since the scene is object based, only the updates need to be sent over the network. This reduces bandwidth requirements in the real time applications.

MPEG-4 also provides a delivery framework, called Delivery Multimedia Integration Framework (DMIF). DMIF is an interface between the application and transport layers. Any application relied on DMIF does not have to be concerned with the underlying communication technologies [74]. With MPEG-4 DMIF, new underlying transport protocols, such as special protocols for haptic communications, can be easily plugged into the whole system without any modification of the application implementation.

COSMOS stands for Collaborative System based on MEPG-4 Objects and Streams. Figure 24 shows a simplified architecture of COSMOS' framework. The application

loads a VE through the VE Loader. The Scene Manager provides a set of methods to enable the management of the scene content [73]. The BIFS Encoder & Decoder encodes and decodes BIFS streams respectively. The DMIF is responsible for establishing and terminating the communication session and for handling all the requests for media access.

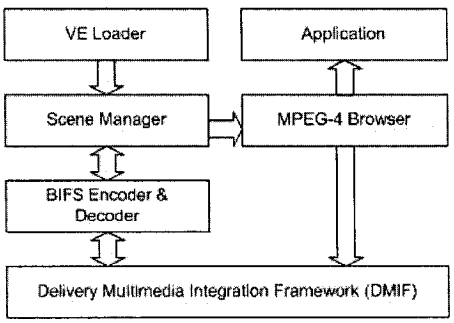



Figure 24: Simplified Architecture of COSMOS

MPEG-4 BIFS describes objects with nodes and fields. In the haptic VE applications, some physical properties are added to the objects so that users interacting with them feel the corresponding sensations. Some examples of these physical properties may be mass, stiffness, surface frictional resistance, stickiness. Much of the information is specific to haptics, and cannot be handled with the current MPEG-4 specifications. Walsh et. al. extends the MPEG-4 BIFS nodes in order to cover these additional properties [75].

Haptic Real Time Controller has been discussed in chapter 4, so it is not repeated here.

### 5.4.2 Surgical Simulation Station

A Surgical Simulation Station is the core for the chosen application. Figure 25 shows the architecture of a VE Station for our surgical simulation.

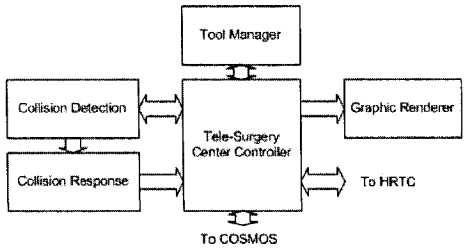



Figure 25: VE Station Architecture for Tracheotomy Simulation

The Tele-Surgery Center Controller (TSCC) can be seen as a state machine controller, which coordinates all the components together and makes the simulation work properly. The ToolManager has controls of all the surgical tools such as scalpel, hook, and trachea tube, etc. The Collision Detection unit checks the collision between surgical tool and neck part of the virtual patient. The Collision Response unit computes the force and torque feedback to the operator. Its algorithm depends on the state of the operation and the virtual object under interaction. Java 3D is responsible for the graphic rendering. Our work uses the MPB Freedom 6S hand controller as the haptic input device which has been introduced in chapter 3. It features a total of six degrees of freedom in terms of position and orientation which enables the user to feel the force and the torque at the same time.

### 5.5 Application Implementation

Java is chosen as the programming language for the implementation of the whole application. Virtual Reality Modeling Language (VRML) is used to describe the virtual surgery room. Graphics is rendered by Java 3D. Since the API of the MPB haptic device is written in C++, the Java Native Interface (JNI) is used to communicate with the native API functions. The reasons we chose Java as the programming language are as follows:

- Java is an object-oriented program language with high portability. It is written once and runs everywhere.
- Java 3D API gives user high level control of objects in the virtual environment and takes care of graphic rendering. With OpenGL, the user has to deal with geometric primitives such as point, line and triangle.
- Most of haptic VE applications are implemented with C or C++ today. We want to make a performance comparison between them on haptic VE applications later on.

### 5.5.1 Virtual Surgery Room Modelling

Since modelling the virtual surgery room directly using Java 3D is very time consuming and inefficient, the software COSMO Worlds 2.0 is used to model the virtual operation surgery room and it generates a file in VRML format. Then the virtual room in VRML format is loaded into Java 3D using the VRML loader. Figure 26 clearly shows the process.

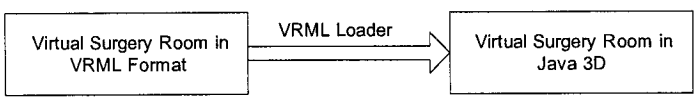



Figure 26: Process from VRML to Java 3D

The existing surgery room was originally created by some former Co-op students in the DISCOVER Lab. We modified some objects in order to fit into the application. In the virtual room, a patient is lying down on the surgery table waiting for the surgery. It also includes some Surgical instruments and equipment such as trays, respiration and circulation monitoring equipments, surgical gauzes, and intravenous infusion equipment, X-ray displaying panel, and so on. Texture mapping is used for some objects to make them look more realistic. The Figure 27 is a snapshot of the virtual surgery room.

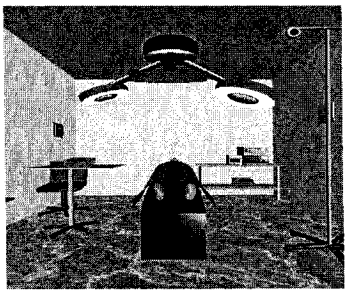


Figure 27: Virtual Operation Room

The main operation region of tracheotomy is the neck, so it is more elaborately modelled. The top skin on the neck is modeled as a collection of 6400 skin elements. Each skin element is a small square which is created using the *Shape3D* class in Java 3D. As mentioned earlier in the Java 3D introduction, each *Shape3D* object is defined with a *Geometry* object and an *Appearance* object. The *QuadArray* class is used to create the geometry of the skin elements. An *Appearance* object is used to define some color parameters for the element. This skin is created using Java 3D classes directly since we want to have control of each small element during simulation. We simplified the skin as two layers of soft tissues: the dermal layer and the muscle layer. These two layers can be felt haptically in the simulation, but we did not model the two layers graphically. To make the collision detection simpler, we assume the skin within the operation region is flat.

### 5.5.2 Application Graphic User Interface

Figure 28 shows the application Graphic User Interface (GUI). There are four components in the GUI: Mode, View, Tool, and Active Tool. Operations in the Mode component also can be triggered with the keyboard. View component is used to change the viewer's position to some important points. The user can navigate in the virtual room with the mouse directly. Tool component is used to load the necessary surgical tool. At a

given time, only one surgical tool is allowed to be active and coupled with haptic device. Active Component is used to select the active tool.

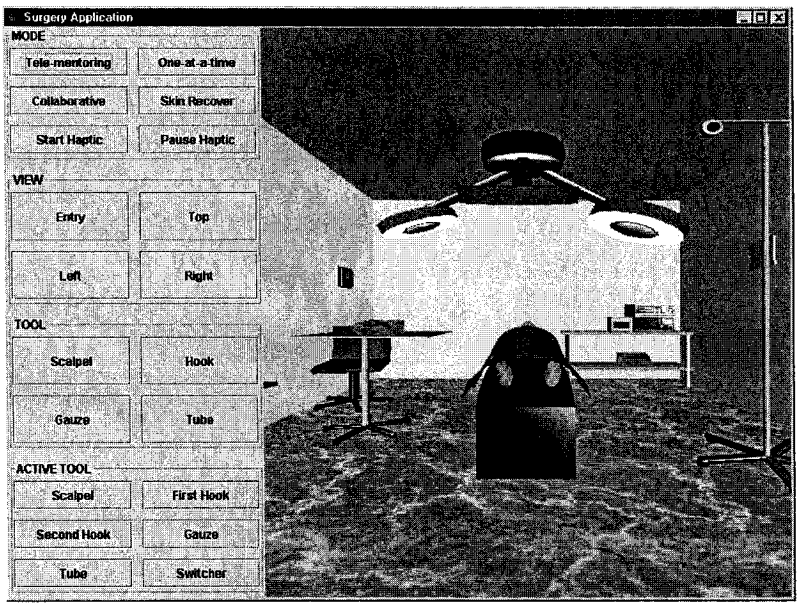



Figure 28: Graphic User Interface for the Application

### 5.5.3 Simulation of Cut

The first action in the tracheotomy is to make a vertical incision on the skin right above the trachea. The handle of MPB Freedom 6S is now modeled as a scalpel in the virtual world. The user just takes the handle as the scalpel and makes the cut. The position and the orientation of handle are retrieved with a servo loop in the application program, so the movement of scalpel in the virtual world is synchronized with the handle. If there is collision between the end tip of scalpel and skin piece, we assume the user starts to make the cut. Then both the visual change of skin and haptic feedback are displayed to the user. So how to simulate the cut graphically and how to generate a realistic haptic feedback are the two main issues here.

To graphically display the cut, we take the advantages of Java 3D. As mentioned above, the skin piece is modeled as 6400 small skin elements. The *appearance* of the skin elements touched by scalpel is changed into transparent from skin color appearance, so the organs below the skin are seen. We could define some blood vessel under the skin. If the scalpel cuts the blood vessel, there will be a bleeding animation. Then the operator has to remove the blood with the gauze before any other action can be made. The bleeding simulation uses Java 3D for animation and it takes a lot of resources and affects the application performance. The figure 29 shows the cut simulation:

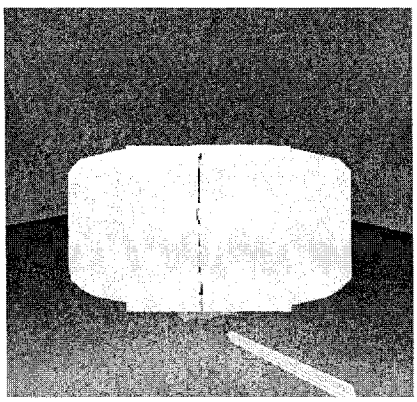


Figure 29: Visual Result of the Cut

The force simulation during the cut is more complex. T. Chanthasopeephan et. al. obtained the force versus displacement relationship through cutting pig liver [76]. In their experiments, the cutting blade was set to move at constant velocity of 0.04 inch/second with a travel distance of 5 inches. The cutting forces were measured by some force sensors and were plotted versus the displacement of the cutting blade. The results revealed that the cutting path was formed by a repeating sequence of localized deformation followed by localized micro fracture. The measured force versus cut-length curves are repeatable in the way that it starts from a small force during tissue deformation and increases to a higher force as impending localized fracture is about to take place, then

the force suddenly drops as onset of localized crack extension occurs. It was observed that the magnitude of the force directly correlated to the depth of cut.

We assume the force versus displacement characteristic in the cut of skin is similar to that of liver cut, since the only difference is that skin is modeled as several layers with different stiffness. To simulate this force versus displacement relationship, we use a moving spring model (Figure 30).

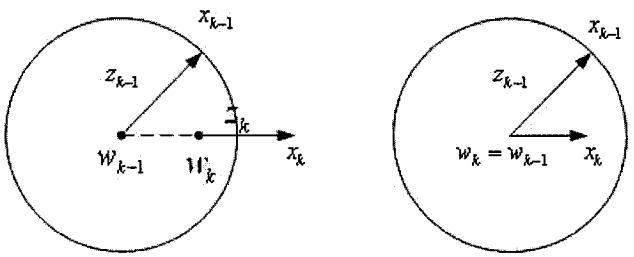


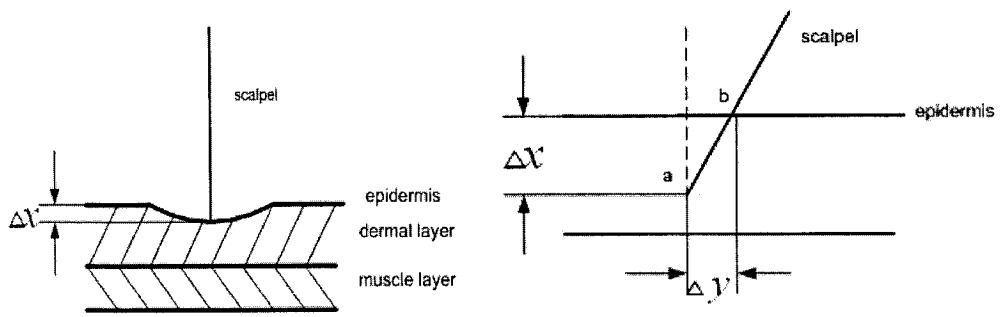

Figure 30: Moving Spring Model [77]

In the figure, $w$ is defined as an adhesive point on the skin and $x$ is defined as scalpel. $z = x - w$ describes the micro-movement between scalpel and adhesive point. At the start point, both points sit together. The force feedback is proportional to the micro-movement and the stiffness of different skin layer using hook's law. The following expressions are used to update the position of adhesive point:

$$w_k = \begin{cases} x_k - \dfrac{x_k - w_{k-1}}{|x_k - w_{k-1}|} z_{\max}, & if\ |x_k - w_{k-1}| > z_{\max;} \\ w_{k-1}, & otherwise. \end{cases}$$

This model was initially proposed to simulate the friction force [77] and did not consider the normal force effects on the friction. In our implementation, we consider the effects of the normal force.

In most haptic applications, point-based haptic rendering is used because of its simple and less computationally expensive implementation, in which the haptic probe is simply considered as a point. This technique may be enough to discover the shape and the surface properties of some virtual object. But it is certainly not a good choice for surgical simulation where most surgical operations are tool-object interactions. For instance, when a surgeon makes an incision on the patient's skin, the scalpel may get in contact with the dermal and muscle layer at the same time. Both end tip and sides of the scalpel contribute to the sensory feedback. In this case, both force and torque should be sent to the user for a realistic feel. For this purpose, we divided the cutting process into 3 states: the end tip of scalpel punctures the epidermis, the incision is made through movement of scalpel inside the skin, and the scalpel is pulled out of the skin.




(a) (b)

Figure 31: Force Simulation of the Cut

In the first state (Figure 31.a), if the normal force is bigger than what the skin can hold, the end tip of the scalpel punctures the epidermis. The epidermis and dermal layer are considered as one layer for simplicity. Damping term could be added. But since almost no surgeon makes cut on the patients with very high speed in the real surgery, the damping effect can be ignored.

In the second and third state of the cut (Figure 31.b), both force and torque are considered. The torque is computed with expression:

$$T=\vec{r}\times\vec{F},$$

where $\vec{r}=\overrightarrow{ab}$, $\vec{F}=k\overrightarrow{\Delta y}$ and $k$ is the stiffness of the skin.

### 5.5.4 Simulation of Pull

After the incision is made, we are ready to make an opening to pull apart the skin and subcutaneous tissues with two hooks. These two hooks will pull the tissues to the opposite directions. This is also an interactive simulation, which means the operation area exposed by pulling will change from time to time according to the movement of the hook. When the hook goes away from the incision, the operation area should increase accordingly and vice versa. The polynomial interpolation is used to compute the pulling curve (Figure 33). The methods used for graphic display and computation of the force is similar to that of cut.

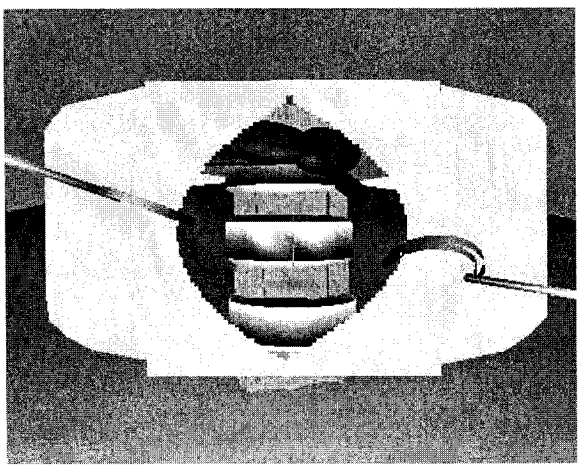


Figure 32: Simulation of Pull

Similar methods are also used for the simulations of trachea tube insertion and hook release.

### 5.5.5 Force Interpolation

As mentioned above, to get a realistic sense of touch, the update rate of haptic rendering has to be around 1 KHz. In our design, Java is used as the programming language and Java 3D is responsible for graphic rendering. Java 3D has its own thread to render graphics around 25~30 times per second. Since it runs on a Java virtual machine and a Java 3D thread has the higher priority, we found that a stable 1 KHz update rate cannot be achieved at all. The relationship between sampling rate and the stability of the system was discussed in Chapter 3. There is a tradeoff between the sampling rate and the highest achievable stiffness to make system stable. Although we do not need to simulate a very hard surface in our application, it is very possible that the computed forces or torque have big gaps between successive frames, thereby creating sudden jumps and degrading the sense of touch. To compensate this problem, we use a simple linear interpolation scheme to minimize discontinuity in force and torque display between successive frames.

Let $F_{k-1}$ be the force displayed at the previous frame, $F_k$ be the force generated during the current frame, and $\Delta F_{\max}$ be the maximum force difference allowed between successive two frames to ensure a continuous force feedback. Without loss of generality, let us assume $F_k > F_{k-1}$. The following scheme is used to adjust $F_k$:

If $(F_k - F_{k-1}) > 2\Delta F_{\max}$, then

$F_k = F_{k-1} + \Delta F_{\max}$;

Else if $(F_k - F_{k-1}) > \Delta F_{\max}$, then

$F_k = (F_k + F_{k-1})/2$.

The experimental result shows that this does improve the continuity of the output force. The same method is also used for the torque interpolation. However, this technique does introduce a little computation delay in the haptic rendering loop.

### 5.6 Tele-Mentoring

In the traditional surgical training, hand-on-hand training is always a very efficient way to teach the trainee how to do each action properly. In our simulation, we provide this training method by coupling two haptic devices together over the network. Thus, the slave haptic device moves together with the master device. Being master or slave can be switched in real time. Therefore, the trainer can know the difficulties with the trainee and give a suggestion immediately.

There are two main concerns to successfully implement tele-mentoring. The first one is how to make the slave device follow the master device's movement promptly and smoothly, while the underlying networks experience variable latency and packet loss. In the proposed system architecture for C-HAVE, this is solved by using a HRTC which includes the compensation for the network time delay. The second issue is that of how to move the slave device to the target position precisely. The settling time, which is defined as the amount of time elapsed between the moment when the device first reaches a commanded position and the one when it maintains the commanded position within an acceptable pre-defined error value [78], should be short. Thus, the second issue is just the classical position control of motors which sets the haptic probe to a certain position and orientation.

A Haptic device is a typically an open loop system. The user manipulates the probe of the device to control its position. Currently, most of haptic devices usually come with an API

library that includes functions for retrieving position and orientation of the probe and for sending force and torque commands to the motors but they rarely provide information about dynamics properties of the devices [31]. Thus, a close-loop system with a digital controller should be designed to command the device to a target position. Figure 34 is a block diagram showing this close loop control process where $X_d$, $X$, $e$, and $F$ represent desired target position, actual probe position, error, and the force sent to the device respectively.

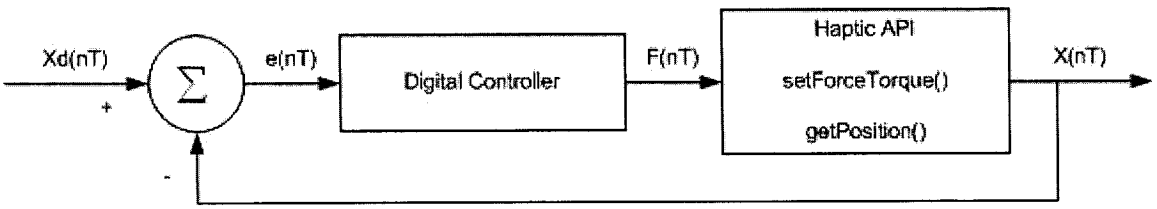



Figure 33: Closed Loop Haptic Position Control System

Depending upon how the feedback signals are processed by the digital controller, different levels of performance can be achieved. In the classical control theory, there are three types of control techniques: proportional, derivative and integral control. Usually the PID controller, which combines all three techniques, offers the best results.

In our surgical simulation, MPB Freedom 6S is chosen as the haptic device which features 6 degrees freedom output. One joint or two joints control each degree freedom, so the control system must consider this multi-axis characteristic. Generally a linear interpolation is required for multi-axis motion from one point to another in a straight line. The controller has to determine the speeds for each motor so that the movements are coordinated.

Currently we have not implemented such a digital controller for position control by ourselves. The experiments are still in the stage of positioning each joint separately and the tuning process for each joint are time consuming. Fortunately, TiDec®, a HRTC from Handshake VR Inc., already includes a digital controller for position control and makes it available through its API function calls.

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

The thesis starts with an overview of Virtual Reality and haptic interfaces. The Collaborative Haptic Audio-visual Virtual Environment (C-HAVE) was discussed in detail. Then, the effects of adding haptics to a collaborative virtual environment are demonstrated experimentally. The experimental results indicate that supporting haptic force feedback in a collaborative virtual environment makes manipulation of common objects significantly faster and more precise. In addition, the use of haptic devices can greatly improve the sense of immersion in the virtual environment.

In the thesis, a generic system architecture using HRTC for C-HAVE was analyzed in detail and experimentally compared with other architectures. The results indicate that users experience less sluggish response that provides a more realistic results and less completion time for a given task, with the time delay compensation approach in the HRTC.

The generic architecture is then successfully applied on a tele-haptic tracheotomy simulation. In the surgical simulation, the ray-based haptic rendering technique is used for the computation of both force and torque feedback which gives the user a more

general tool-object interaction sensation. A linear force interpolation improves the continuity of output feedback.

In terms of programming issues, VRML is a very efficient descriptive language to model realistic 3D virtual worlds. Java 3D API takes all the responsibility for the graphic rendering and gives the users a high level control of objects. On the other hand, this easiness for graphic rendering does come with the trade off of performance. Java 3D thread has a higher priority than others to guarantee the frame fresh rate. If the graphics are very complex, the rendering takes a lot of time and this will affect the haptic rendering thread, which requires 1 KHz update rate. In addition, Java, as a high level object oriented language running on Java Virtual Machine, its garbage collection will also affect the performance of the real time applications.

### 6.2 Future Work

As one of the first uses of the HLA standard in collaborative haptic environment settings, C-HAVE is a good start and more work will be done in terms of usability studies and architectural evaluations. Besides network delay, other network impairments such as jitter and packet loss also have dramatic impacts on the performance of tele-haptic; for instance, jitter makes the tele-haptic system unstable and imposes Parkinson effects on the system [10]. The researchers at DISCOVER have been developing a new transport layer protocols for tele-haptic applications. SCTP: the Synchronous Collaboration Transport Protocol [78] and Smoothed SCTP [79] will be further studied and applied in a haptic channel. SCTP and Smoothed SCTP provide efficient and reliable communication for collaborative tasks to deal with the network impairments. Other future research work

will include studies on how network impairments affect manipulated objects with different properties such as stiffness, deformation and damping.

For the collaborative tele-haptic surgery simulation, more tests and usability studies are required to test the robustness of the simulation. Currently, each local station supports only one haptic device, which limits the interactivity of the application. To tackle this issue, we are working on giving the possibility to the user to use concomitantly different input devices. In the near future, we will definitely use some efficient deformation algorithms to improve the graphics display.

# References

[1] W. R. Sherman, and A. B. Craig, "Understanding Virtual Reality: Interactive, Application, and Design", Morgan Kaufmann Publishers, 2003.

[2] SUN Microsystems Inc., "Java 3dTM API Collateral", http://java.sun.com/products/java media/3D/collateral/j3d_api/j3d_api_3.html.

[3] D. Selman, "Java 3D Programming", Manning Publications Co., 2002.

[4] SUN Microsystems Inc., "Java Native Interface Tutorial", http://java.sun.com/docs/books/tutorial/native1.1/concepts/index.html.

[5] R. Carey, and G. Bell, "The Annotated VRML 97 Reference Manual", http://www.cs.vu.nl/~eliens/documents/vrml/reference/BOOK.HTM.

[6] Web 3D Consortium, "VRML97 Functional specification and VRML97 External Authoring Interface (EAI) International Standard ISO/IEC 14772-1:1997 and ISO/IEC 14772-2:2002", http://www.web3d.org/x3d/vrml/index.html.

[7] P. Buttolo, R. Oboe, and B. Hannaford, "Architectures for shared haptic virtual environments", *Computers and Graphics*, 21 (4), pp.421-429, 1997.

[8] A. Liu, F. Tendick, K. Cleary, and C. Kaufmann, "A Survey of Surgical Simulation: Applications, Technology, and Education", *Presence*, Vol. 12, No. 6, pp. 599-614, 2003.

[9] Defense Modeling and Simulation Office (DMSO), "High Level Architecture for Simulations Interface Specification", Version 1.3.

[10] X. Shen, F. Bogsanyi, L. Ni, and N. D. Georganas, "A Heterogeneous Scalable Architecture for Collaborative Haptics Environments," *2nd IEEE Workshop on Haptic, Audio and Visual Environments and their Applications -HAVE*, September 2003

[11] Webster Online Dictionary, http://www.websters-online-dictionary.org/definition/virtual.

[12] Dictionary.com, http://dictionary.reference.com/search?q=virtual.

[13] S. Aukstakalnis, D. Blatner, and S. F. Roth, "Silicon Mirage: The Art and Science of Virtual Reality", Publisher: Peachpit Pr., 1992.

[14] The University of North Carolina at Chapel Hill, College of Arts and Sciences, http://www.cs.unc.edu/~walk/.

[15] Boeing Inc., http://www.vr-atlantis.com/vr_systems_guide/65.html.

[16] Silicon Graphics Inc., http://www.sgi.com/realitycenter/.

[17] The University of Michigan Virtual Reality Laboratory (VRL) at the College of Engineering, http://www-vrl.umich.edu/project/automotive/.

[18] P.J. Costello, "Health and Safety Issues associated with Virtual Reality – a Review of Current Literature", http://www.agocg.ac.uk/reports/virtual/37/37.pdf.

[19] Extreme Toys for Boys.com, "i-Scape II Head Mounted Display", http://extremetoysforboys.com/.

[20] P. T. Weiss, and A. S. Jessel, "Virtual Reality Applications to Work", *WORK*, 11 (3), pp. 277-293, 1998

[21] K. Hikichi, I. Arimoto, H. Morino, K. Sezaki and Y. Yasuda, "Evaluation of Adaptation Control for Haptics Collaboration over the Internet", *Proc. of IEEE Communications Quality & Reliability (CQR) International Workshop*, May 2002.

[22] K.Hikichi, H.Morino, I.Arimoto, I.Fukuda, S.Matsumoto, M.Iijima, K.Sezaki and Y.Yasuda, ``Architecture of Haptics Communication System for Adaptation to Network Environments", *Proc. of 2001 IEEE International Conference on Multimedia and Expo (ICME2001)*, August 2001

[23] I. Fukuda, and S. Matsumoto, "A Robust System for Haptic Collaboration over the Network", *Touch in Virtual Environments Conference*, University of Southern California, Los Angeles, Feb. 2001.

[24] J. Hespanha, and M. McLaughlin, "Haptic Collaboration with Heterogeneous Devices", *Touch in Virtual Environments Conference*, University of Southern California, Los Angeles, Feb. 2001.

[25] C. Gunn, "Collaborative Haptic Applications", *Reachin Users Group Meeting*, Stockholm, Sep. 2002.

[26] R. S. Kalawsky, "The science of virtual reality and virtual environments", Wokingham, England: Addison-Wesley, 1993.

[27] SensAble Technology Inc., http://www.sensable.com/.

[28] MPB Technologies Inc., http://www.mpb-technologies.ca/.

[29] FCS Robotics, http://www.fcs-cs.com/robotics/products/hapticmaster.

[30] K. Salisbury, F. Conti, and F. Barbagli, "Haptic Rendering: Introductory Concepts", *IEEE Computer Graphics and Applications*, pp.24-32, Vol. 24, No. 2, 2004.

[31] C. Basdogan, and M. A. Srinivasan, "Haptic Rendering in Virtual Environment", http://network.ku.edu.tr/~cbasdogan/-Papers/VRbookChapter.pdf.

[32] J. E. Colgate, P. E. Grafing, M.C. Stanley, and G. Schenkel, "Implementation of Stiff Virtual Walls in Force-Reflecting Interface", *Proc. IEEE-VRAIS*, pp.202-208, 1993.

[33] P. Hubbard, "Collision Detection for Interactive Graphics Applications", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 1, pp. 219-230, 1995.

[34] J. Cohen, M. Lin, D. Manocha, and K. Ponamgi, "I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scaled Environments", *Proceedings of ACM Interactive 3D Graphics Conference*, pp. 189-196, 1995.

[35] C. Ho, C. Basdogan, and M.A. Srinivasan, "Modeling of Force and Torque Interactions Between a Line Segment and Triangular Surfaces for Haptic Display of 3D Convex Objects in Virtual and Teleoperated Environments", *International Journal of Robotics (special issue on Tactile Perception)*, Vol. 19, No. 7, pp. 668-684, 2000.

[36] C. Ho, C. Basdogan, and M.A. Srinivasan, "An Efficient Haptic Rendering Technique for Displaying 3D Polyhedral Objects and Their Surface Details in Virtual Environment", *Presence: Teleoperators and Virtual Environments*, Vol. 8, No. 5, pp. 477-491, 1999.

[37] J.K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles, "Haptic Rendering: Programming touch interaction with virtual objects", *Proceedings of the ACM Symposium on Interactive 3D Graphics*, Monterey, California, 1995.

[38] J. J. Gil, A. Avello, A. Rubio, and J. Florez, "Stability Analysis of a 1DOF Haptic Interface Using the Routh-Hurwitz Criterion", *IEEE Transaction on Control Systems Technology*, Vol. 12, No. 4, July 2004.

[39] T. Yoshikawa, Y. Yokokohiji, T. Matsumoto, and X. –Z. Zheng, "Display of Feel for the Manipulation of Dynamic Virtual Objects", *Trans. ASME, Journal of Dynamic Systems, Measurement, and Control*, vol. 117, no. 4, pp. 554-558, 1995.

[40] R. J. Adams, and B. Hannaford, "Stable Haptic Interaction with Virtual Environments", *IEEE Transactions on Robotics and Automation*, vol. 15, No. 3, pp. 465-474, 1999.

[41] S. C. Cannon, and G. I. Zahalak, "The Mechanical Behavior of Active Human Skeletal Muscle in Small Oscillations", *Journal of Biomech.*, vol. 15, pp. 111-121, 1982.

[42] W. R. Murray, "Essential Factors in Modeling the Modulation of Impedance about the Human Elbow", *Ph.D dissertation*, Department of Mech. Eng., M.I.T, 1988.

[43] R. J. Adams, M. R. Moreyra, and B. Hannaford, "Stability and Performance of Haptic Displays: Theory and Experiments", *Proc. ASME International Mechanical Engineering Congress and Exhibition*, Anaheim, CA, pp. 227-234, 1998.

[44] J. M. Brown and J. E. Colgate, "Minimum Mass from Haptic Display Simulations", *Proc. ASME International Mechanical Engineering Congress and Exhibition*, Anaheim, CA, pp.249-256, 1998.

[45] M. Minsky, M. Ouh-Young, O. Steele, F. P. Brooks, and M. Behensky, "Feeling and Seeing Issues in Force Display", *Computer Graphics*, vol. 24, no. 2, pp. 235-243, 1990.

[46] J. E. Colgate, M. C. Stanley, and J. M. Brown, "Issues in the Haptic Display of Tool Use", *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Pittsburgh, PA, pp. 140-145, 1995.

[47] T. Massie, and J. K. Salibury, "The PHANTOM haptic interface: A device for probing virtual objects", *Proc. ASME Winter Annual Meeting: Symposium on Haptic Interfaces for Virtual Environment and TeleOperator Systems*, 1994.

[48] J.G.S. Demers, J.M.A. Boelen, and I.P.W.Sinclair, "Freedom 6S Force Feedback Hand Controller", *SPRO'98 1st IFAC Workshop on Space Robotics*, Montreal, Canada, October 19-22, 1998.

[49] MPB Technology Inc. "Freedom 6S Feedback Hand Controller System Specification", July 2000.

[50] MPB Technology Inc. "Freedom6S.dll v1.4 Source Code Overview", 2004.

[51] MPB Technology Inc. "Freedom6S library v1.4 API User Manual", 2004.

[52] X. Shen, J. Zhou, A. El Saddik, and N.D. Georganas, "Architecture and Evaluation of Tele-Haptic Environments", *The 8-th IEEE International Symposium on Distributed Simulation and Real Time Applications*, Budapest, Hungary, Oct. 2004

[53] E. Sallnas, "Supporting Presence in Collaborative Environments by Haptic Force Feedback", *ACM Transactions on Computer-Human Interaction*, Vol. 7, No. 4, pp. 461-476, December 2000.

[54] D. Wang, K. Tuer, M. Rossi, L. Ni, and J. Shu, "The effect of time delays on tele-haptics," *2nd IEEE International Workshop on Haptic, Audio and Visual Environments and their Applications - HAVE*, pp. 7-12, September 2003.

[55] T. B. Sheridan, "Space Teleoperation through Time Delay: Review and Prognosis," *IEEE Trans. on Robotics and Automation*, vol. 9, pp. 592-606, October 1993.

[56] P. Arcara, and C. Melchiorri, "Control Schemes for teleoperation with time delay: A comparative study", *Robotics and Autonomous Systems*, Vol. 38, pp. 49-64, 2002.

[57] X. Shen, and N. D. Georganas, "Joint Manipulation Management in Collaborative Virtual Environment", *IEEE Workshop on Haptic, Audio and Visual Environments and their Applications -HAVE*, September 2003.

[58] R.J. Anderson, and M.W. Spong, "Bilateral control of teleoperators with time delay", *IEEE Transactions on Automatic Control*, 34 (5), pp.494-501, 1989.

[59] R.J. Anderson, M.W. Spong, "Asymptotic stability for force reflecting teleoperators with time delay", *International Journal of Robotics Research*, 11 (2), pp. 135-149, 1992.

[60] M. McLaughlin, "Touch in Virtual Environment: Haptics and the Design of Interactive System", Prentice-Hall, December 2001.

[61] N. Langrana, G. Burdea, J. Ladeiji, and M. Dinsmore, "Human performance using virtual reality tumor palpation simulation", *Computers and Graphics,* vol. 21, No. 4, pp. 451–458, July 1997.

[62] Immersion Corporation, http://www.immersion.com.

[63] R. Balaniuk, and I. F. Costa, "LEM - An Approach for Physically Based Soft Tissue Simulation Suitable for Haptic Interaction," *Proc. of IEEE-ICRA 2001 Robotics and Automation,* Vol. 3, pp. 2337-2343, 2001.

[64] L. S. Machado, R. M. Moraes, and M. K. Zuffo, "Fuzzy Rule-Based Evaluation for a Haptic and Stereo Simulator for Bone Marrow Harvest for Transplant," *Preprints of the Fifth Annual PHANToM Users Group Workshop*, Given Institute, Aspen, Colorado 2000.

[65] M. Ishii, M. Nakanta, M. Sato, "Networked SPIDAR: A Networked Virtual Environment with Visual, Auditory, and Haptic Interactions, *PRESENCE*, vol. 2, no. 4, pp. 351-359, 1994.

[66] C. Basdogan, C-H. Ho, M. Slater, and M. A. Srinavasan, "The role of haptic communication in shared virtual environments", *Proceedings of the Third PHANToM Users Group Workshop, PUG98.* AI Technical Report no. 1643 and RLE Technical Report no. 624. Cambridge, MA: MIT.

[67] C. Ho, C. Basdogan, M Slater, N. Durlach, and M. A. Srinivasan, "An experiment on the influence of haptic communication on the sense of being together", Paper presented at the British Telecom Workshop on Presence in Shared Virtual Environments, http://www.cs.ucl.ac.uk/staff/m.slater/BTWorkshop/touchexp.html

[68] J. Barone, "Tracheotomy", http://www.healthatoz.com/healthatoz/Atoz/ency/tracheotomy.html.

[69] Twin Enterprises Inc., "What is Tracheotomy", http://www.tracheostomy.com.

[70] K. M. Bleile, "The Care of Children with Long-Term Tracheotomies", Singular Pub Group, 1993.

[71] R. M. Younson, "In the Surgery Book: An Illustrated Guide to 73 of the Most Common Operations", New York: St. Martin's Press, 1993.

[72] HandShake VR Inc., http://www.handshakeinteractive.com/index.php.

[73] V. Darlagiannis, and N. D. Georganas, "COSMOS: Collaborative System based on MPEG-4 Objects and Streams", MCRLAB, University of Ottawa.

[74] R. Koenen, "MPEG-4 Overview", *ISO/IEC JTC1/SC29/WG11 N4668*, pp.2, 2002.

[75] D. Walsh, C. Gunn, M. Adcock, and M. Hotchins, "Haptics Nodes for MPEG-4 BIFS (Object Surface)", ISO/IEC JTC1/SC29/WG11, MPEG2001/m7409, July 2001.

[76] T. Chanthasopeephan, J. P. Desai, and A. C. W. Lau, "Measuring Forces in Liver Cutting: New Equipment and Experimental Results", *Annals of Biomedical Engineering*, vol. 31, issue 11, pp. 1372-1382, December 2003.

[77] V. Hayward, "A New Computational Model of Friction Applied to Haptic Rendering", Experimental Robotics VI, New York, LNCS 250, pp.404-412, 2000.

[78] S.Shirmohammadi, and N. D. Georganas, "Collaborating in 3D Virtual Environments: A Synchronous Architecture", *Proc.IEEE 9th Inter. Workshops on Enab. Technol. Infr. For Collabor. Entreprises (WETICE) Knowledge Media Networking workshop*, NIST, Washington DC, June 2000.

[79] S. Dodeller and N.D. Georganas "Transport Layer Protocols for Tele-haptics Update Messages", *Proc. Bienneal Symposium on Communication*, Kingston, June 2004.