DESIGN AND PERFORMANCE ANALYSIS OF A VIRTUAL REALITY-BASED TELEREHABILITATION SYSTEM

BY

GEORGE V. POPESCU

A Dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Dr. Grigore Burdea and Dr. Manish Parashar

and approved by

____

____

____

____

____

New Brunswick, New Jersey

January, 2001

ABSTRACT OF THE DISSERTATION

Design and Performance Analysis of a Virtual Reality-based Telerehabilitation System

By George V. Popescu

Dissertation Directors: Dr. Grigore Burdea and Dr. Manish Parashar

In recent years the area of medical VR applications has continuously expanded, addressing new domains such as home healthcare, clinical neuropsychology, and rehabilitation. The research presented here explores the use of Virtual Reality (VR) for telerehabilitation applications. A prototype platform for VR-based telerehabilitation was defined first. The main component of the platform is the hand force feedback unit. A programming library – the Rutgers Haptic Library – was developed for modeling hand haptic interactions. The software was used to build real-time VR simulations that involve elastic and plastic deformations and physical modeling.

The VR-based platform was the basic component of the telerehabilitation architectures we developed. These architectures use Virtual Reality as an advance interface for therapy as well as to enable communication between the therapist at the clinic/hospital and the remote patient or group of patients. The first prototype supports offline interaction between the therapist and the VR-enabled patient site. This "store and forward" system uses a Client/Server architecture. The client (patient home) runs VR rehabilitation exercises with force feedback and collects patient data. The exercises simulate physical and functional rehabilitation routines. Patient data are forwarded to the server (clinic site), which stores medical records and runs data analysis software. System performance over several types of connections was measured in laboratory experiments. The guidelines extracted from these experiments help sizing the system in terms of recorded data and number of concurrent users. Clinical trials were conducted at the Stanford University Medical School. Data collected during these trials indicates that patient's level of effort and grasping strength increased after using the VR-based rehabilitation system.

"Store and forward" systems are insufficient for implementing the whole range of potential telerehabilitation services. The second architecture developed in this thesis uses a Shared Virtual Environment to enable real-time patient-therapist interaction. The prototype system allows the therapist to perform remote physical therapy and collect patient data. Simulated physical interactions between therapist and patient were implemented using force feedback.

Acknowledgements

I would like to thank Dr. Grigore Burdea for his advice, support and encouragement during my research years at CAIP. His effective research style and visionary goals motivated me to complete this long thesis research journey.

I am grateful to Dr. Manish Parashar for providing me with valuable guidance and support during my thesis research.

I would like to thank Dr. James Flanagan for encouraging my work and promoting the human-machine interaction research at CAIP.

I would like to thank Dr. Deborah Silver and Dr. William Craelius for providing valuable suggestions regarding my thesis research.

Special thanks to all people I had the privilege to work with during my years at CAIP's Human-Machine Interface Laboratory. In particular I would like to thank Mourad Bouzit, Rares Boian, Mike Girone and George Patounakis for their contributions.

Many thanks go to all colleagues involved in Stanford-Rutgers Telerehabilitation project for their dedicated work.

The research reported in this thesis was supported by grants from the National Science Foundation (NSF grant BES-9708020 and IRI-9618854) and from the CAIP Center at Rutgers University.

Dedication

To my family.

Table of Contents

Lists of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In recent years, information technology has radically changed the way health care is delivered. On-line patient databases, pre-surgery simulations, telemedicine, medical robots, are illustrative of technological advances in medicine. The 1997 ARPA report (Weghorst and Furness, 1997) identifies a need to take advantage of emerging human interface technology for advanced medical systems. Among the motivating reasons were the need to assimilate and display large volumes of data, the need for seamless ubiquitous access to medical information, and the need for interoperability among devices and databases. Along with traditional workstations and device monitors, the medical interface environment of the future envisioned by the report authors will incorporate emerging technologies such as:

- Virtual Reality - mediated by head-mounted displays (HMD), "cave"-type environments, and "holographic" or stereographic systems (Middleton and van Millingen, 1999);
- Augmented Perception - mediated by HMDs and other see-through devices;

- Ubiquitous Computing - including interface devices, such as wall panels, wireless pads and PDAs.

Among these new technologies, Virtual Reality has great potential for innovating the medical field. According to Satava & Jones, (2000), "the benefits of Virtual Reality for healthcare can be summarized in a single word: revolutionary". Virtual Reality technology has been used in several medical domains (see (Satava & Jones, 2000) for a detailed review of medical applications in VR): diagnosis (virtual endoscopy, pre-operative planning), therapy (computer-assisted surgery, telesurgery), psychiatry and rehabilitation (phobia therapy, cognitive rehabilitation, and education and training (virtual environments for teaching anatomy, medical simulators). Our work focuses on using Virtual Reality to develop telerehabilitation[1] systems. We believe that haptic Virtual Environments can provide effective treatment for home-based orthopedic rehabilitation. Such a VR rehabilitation system can be used as a foundation for developing network architectures for home telerehabilitation.

Haptics is a recent enhancement to virtual environments allowing users to "touch" and feel the simulated objects they interact with (Burdea, 1996). This sensorial channel complements the usual visual and auditory feedback modalities used in current VR simulations. The use of haptic feedback is mandatory for medical training simulators (surgery, palpation) which require increased realism of the Virtual Environment. Therapy and rehabilitation can also benefit from using haptic Virtual Environments. Haptics increase patient immersion and participation and can potentially led to faster recovery.

[1] For definition of "telerehabilitation" see section 2.3

The first challenge for haptic-based medical applications is the design of compact, light and high-fidelity haptic interfaces. An even bigger challenge is the modeling of the haptic interaction. Physical modeling tools are needed to calculate real-time contact forces, surface deformation, rugosity, object weight, etc. Haptic modeling is still in its infancy. Recent work concentrates on point-based haptic interactions with the virtual models (Srinivasan & Basdogan, 1997). Several commercial libraries provide development tools for finger-based haptic interactions: GHOST (Sensable Technologies, 1998), VPS (McNeely et al., 1999), MAGMA (ReachIn Technologies, 2000). However, haptic interaction modeling for hand haptic devices requires more elaborate models.

Our exploration of haptic rendering methods is motivated by the current lack of modeling techniques for complex haptic devices. We believe that advances in haptic modeling and actuator technology will eventually generate a widespread use of haptic interfaces in medical applications. The medical scenario of the future will include a mix of interface approaches supported by a broad array of input and feedback modalities and methods, telepresence technologies, and computer-mediated assistance.

Along with user interface technologies, communication technologies are expanding the scope of medicine. Advances in communication technologies and the proliferation of experimental and commercial high-speed communication networks have raised the level of connectivity and information accessibility. New generation communication infrastructures such as the Next Generation Internet (NGI, 2000), Internet 2 (Internet2, 2000), and very-high-performance Backbone Network Service (vBNS, 2000), aimed at offering a 100-fold increase in communication speeds and capacity, will be capable of providing sustained real-time communication services. These networks have the potential of supporting an increased range of telemedicine services. Furthermore, recent trends towards integration of telephone, television and data networks will result in very powerful communication frameworks for household users. The rapidly developing network infrastructure for high-speed communication and the dramatical increase in the number of households with PCs (Reinhardt, 1998) is fueling the expansion of home care delivery systems.

At the present time, few medical VR applications are networked. The best known networked medical VR application is telesurgery. Many other emerging telemedicine applications would benefit from using advanced Virtual Reality interfaces. Among these are teletherapy (providing remote therapy), telemonitoring (remote monitoring of patient health), telerehabilitation (providing home therapy for disabled patients), etc. In our work, we explore the use of Virtual Reality interfaces for telemedicine systems, focusing on telerehabilitation applications.

1.2 Approach

The scope of today's telemedicine is limited to remote collection of patient data, information exchange between medical personnel (teleconsultation, telediagnosys), training (telementoring) and patient-physician teleconferencing (mental health, teletherapy). This limitation is related to videoconferencing and "store and forward" technologies currently used to implement telemedicine systems. Patient-physician interaction is generally overlooked, even though that is a fundamental aspect of medicine. Virtual Reality has the potential to further advance the telemedicine, by extending its scope to home care applications. In this context, VR could provide advanced treatment and data collection interfaces as well as enable remote patient-physician interactions. Videoconferencing tools offer a “looking through a window” metaphor which limits the scope of telemedicine applications. Shared Virtual Environments technology can support better interaction between physician and patient. Someone can imagine the physician and patient of the future wearing VR devices in order to meet in a shared VR “consultation” room (Figure. 1.1).

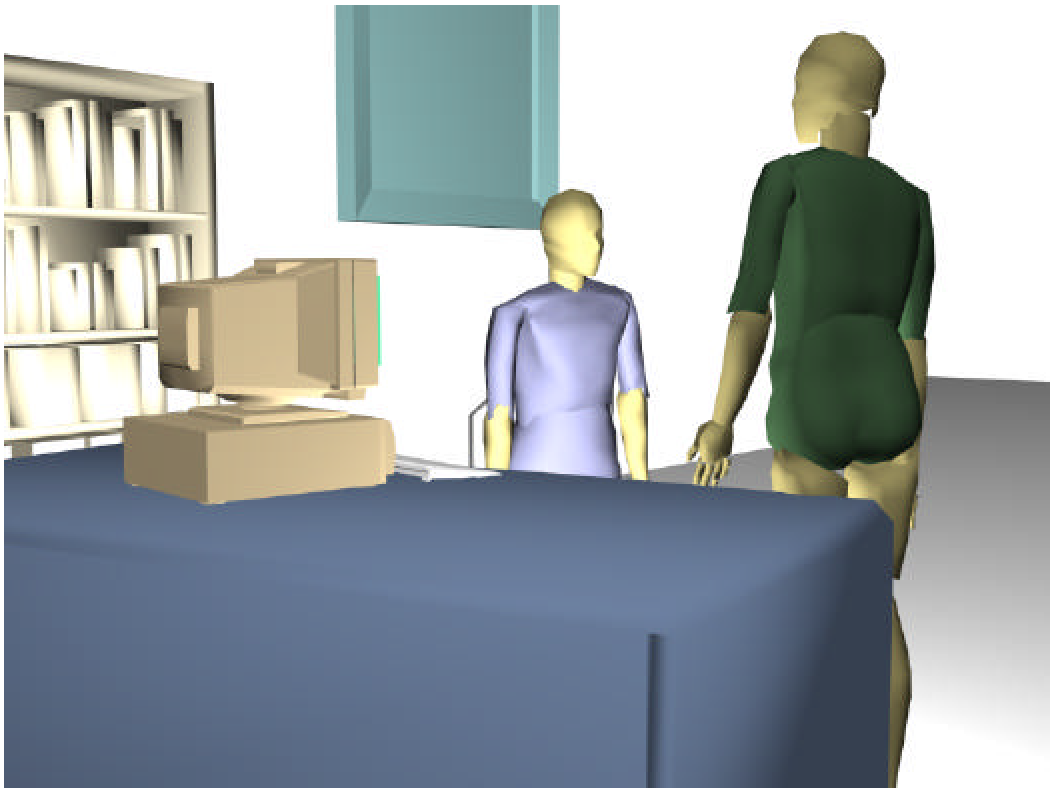


Figure1.1: Future Shared VE “consultation” room

As exciting as the above scenario might look, it is a distant goal for VR-based telemedicine. Among the immediate research issues are the design of VR interfaces, Virtual Reality simulations for home care applications, and overall telemedicine system architecture development. We focused on VR-based applications using force feedback, and the telemedicine system architecture. Our case study is the design of a telerehabilitation system prototype. Our research explores the use of force feedback for orthopedic rehabilitation. The experimental test-bed for this thesis is the NSF-sponsored Rutgers-Stanford home-based orthopedic telerehabilitation project (grant BES-9708020).

One of our research objectives was the definition of a Virtual Reality based platform for home telerehabilitation applications. The platform extends the typical telemedicine system (workstation + videoconferencing) with additional input (voice, hand gestures) and output (force feedback) modalities. We focused on the development of modeling tools for hand haptic interactions since force feedback is an important modality for telerehabilitation systems. Hand interactions with virtual objects involve grasping, pushing, throwing, deforming, etc. Adding force feedback produces more realistic interactions, increasing user participation. The hand haptic interaction developed in this thesis uses a novel approach based on interacting meshes. Fingertips are modeled as low-resolution meshes, which are then used to deform virtual object surfaces. Controlling mesh resolution allows varying the force feedback resolution. In addition, object plastic and elastic deformation is coupled with the force simulation. Based on these models, we created a haptic rendering library used for implementing VR telerehabilitation exercises.

Another objective for our research was to design and develop system architectures for VR-based telerehabilitation. These architectures should use Virtual Reality as an advance interface for therapy as well as to enable communication between the therapist (or physician) at the clinic/hospital and the remote patient or group of patients. Two architectures were proposed for VR-based telerehabilitation. The first one support offline interaction between therapist and VR-enable patient site. This Client/Server architecture implements a “store and forward” type of system. The application is suited for a low bandwidth connection between the patient home and clinic. The Client (patient home) runs VR simulations and collects real-time patient data. The Server (clinic site) stores patient medical records and runs data analysis and visualization software. The therapist studies the patient data and remotely changes VR simulation parameters (level of difficulty, duration, etc.). This remote action implements the feedback loop mechanism of the rehabilitation process.

“Store and forward” systems are insufficient for implementing the whole range of telerehabilitation services. The second architecture uses a Shared Virtual Environment to enable real-time patient-therapist interaction. Simulated physical interactions between therapist and patient are implemented using force feedback. Data transmitted between the two sites include audio, video, images, scene graph information, force, and control commands. This heterogeneous type of data requires small latency (force data), high bandwidth (video, images) networks. The system can support several ”real-time” telemedicine services, such as telemonitoring, teletherapy, teleconsultation and telediagnosys.

1.3 Thesis Research Contributions

This thesis makes contributions on the use of Virtual Reality techniques for creating advanced telemedicine systems. The main contributions are:

1. A prototype Client/Server architecture for telerehabilitation. This supports VR-based home rehabilitation, automatic patient data collection and offline patient-physician interaction.
2. Virtual Environments enhanced with force feedback for orthopedic rehabilitation. The design follows guidelines and examples extracted from rehabilitation literature. The VEs are prototypical and can be reused for other rehabilitation applications.
3. Prototype of a Shared Virtual Environment for telerehabilitation that includes real-time physical (haptic) interactions between patient and physician, patient data measurement, data sharing and videoconferencing.

The thesis research also makes a number of contributions to haptics for Virtual Environments, such as:

1. The control and communication of hand haptic interface design (RM-II); the communication protocol provides support for haptic Virtual Environment simulations.
2. A haptic rendering technique for hand interactions in VE. This technique extends current models designed for point-based haptic interactions to hand-based interactions. Based on this technique, a software library for hand haptic simulation was developed.
3. A haptic programming interface for virtual hand interactions. The programming interface uses haptic extensions of the scene graph model. These extensions provide modularity in development of haptics and graphics simulations.

1.4 Thesis Outline

The next chapter will review the applications of Virtual Reality in medicine, focusing on telemedicine and telerehabilitation. Of special interest are the haptic interfaces used in medical applications. Related work on computer-based patient monitoring and force feedback for rehabilitation is also summarized.

Chapter three describes the Virtual Reality platform for telerehabilitation used in our experiments. This includes the PC-based systems with videoconferencing and multimodal input capabilities and the haptic interfaces for rehabilitation. Haptic devices for hand and ankle, their control interfaces and performances are described in detail.

Chapter four studies the modeling of hand haptic interactions in Virtual Environments. A short review of haptic rendering literature discusses the current methods and motivates the research on virtual hand haptic interaction modeling. The proposed solution uses a novel *haptic interaction mesh* model evolved from haptic interaction point techniques. A software library using haptic scene graph extensions and the haptic interaction model is presented at the end of the chapter.

Chapter five presents a library of Virtual Reality exercises designed for hand and ankle rehabilitation. The concept of Virtual Rehabilitation Room is introduced first. Then the functional and physical therapy exercises designed for hand and ankle are described.

The Client/Server architecture for offline interaction between patient and therapist is described in Chapter six. This includes the clinical database as well as the tools for data analysis and patient data collection. Videoconsultation and remote control of home rehabilitation exercises complete the patient-therapist interaction model. Database client performance measurements are included.

Chapter seven summarizes the clinical trials of the telerehabilitation system. Experimental results from hand rehabilitation patient trials at Stanford University are presented. Ankle rehabilitation proof-of-concept patient trials of are also described.

Chapter eight presents a new approach for telerehabilitation: the Shared Virtual Rehabilitation Room (SVR^2). This architecture supports real-time physical interactions between the patient and the therapist. Simulated physical interactions between therapist and patient are implemented using hand force feedback. The SVR^2 can be used to implement several telemedicine services: telemetry, telediagnosys, teleconsultation and teletherapy.

Concluding remarks and future research directions are given in Chapter nine.

Chapter 2

Virtual Reality in Telemedicine

2.1 Medical Virtual Reality

Classical medical VR applications such as surgical training systems demonstrated that VR meets a critical need for better education and training of medical students and residents. Other applications such as pre-operative planning, computer-assisted surgery, virtual endoscopy have lead the way of VR into medical practice. In recent years the area of medical VR applications has continuously expanded, addressing home healthcare systems, virtual endoscopy, clinical neuropsychology, rehabilitation, and palpation training. While VR had a considerable impact in developing advanced medical systems, the reverse is also true. Medicine pushes the development of advanced VR interfaces, having special requirements for user interaction and physical modeling. Following is a review of medical VR and telemedicine applications.

2.1.1 Medical VR interface and Simulation Requirements

The most pervasive aspect of VR in medical applications is interactive 3D visualization. At the core of the medical VR interface is the display technology including head-mounted displays (HMD), 3D-video monitors, and room-sized CAVEs. As important as visualization is interaction for medical VR. The use of haptics is mandatory for realistic medical VR simulations. The haptic interfaces used in medical applications have special requirements (Burdea, 1998-a):

- Interfaces need to be compact, light, clean, and safe to use;
- High dynamic range is needed for high fidelity of force feedback sensations.
- Large feedback forces are not necessary for surgery applications but physical rehabilitation requires devices able to match human muscle strength.

Haptic interfaces used in medical VR can be classified in two categories:

- general purpose interfaces, such as Rutgers Master II (Gomez et al., 1995), exoskeletons (Bergamasco, 1993), or robotic arms (e.g. PHANToM (Massie and Salisbury, 1994));
- special purpose interfaces, such as Laparoscopic Impulse Engine (Rosenberg and Stredney, 1996), or HT Medical angioplasty haptic device (Bro-Nielsen, 1997).

Highly realistic simulators (surgical simulators) require in general special purpose haptic interfaces. General-purpose haptic interfaces are suited mostly for education and therapy applications. Designing haptic interfaces that meet the above-mentioned requirements is challenging. Furthermore, the cost of a haptic system is more than an order of magnitude higher than that of a PC. That explains the relative small number of haptic interfaces used in medical VR applications.

Even more challenging is simulation modeling. Typically, medical simulations involve organ geometrical modeling. These models range from generic human anatomy models to anatomically-correct models and patient specific data. Several 3D databases offer detailed generic human anatomy models (Viewpoint, 1999), (3D Café, 2000). The models have three levels of resolution: “low”, “medium” and “high”. Some of these are pre-segmented, a necessary condition for simulating the dynamics of human body. Anatomically-correct models can be obtained from Visible Human Project, available online at (WebVH, 2000). Patient specific data can be obtained through CT/MRI image segmentation and 3D reconstruction.

Realistic simulators require collision detection in order to model interaction with the virtual organs. Simplistic bounding box methods are inadequate; precise collision detection is often required. Very often collision detection is just the first stage of more complex physical simulations involving organ deformation, tissue cutting, etc. The simplest methods used in physically-based simulations employ direct vertex-based manipulation, while more realistic ones may require solving partial differential equations, finite elements modeling, etc.

Collision detection is also the fundamental step in haptic rendering. Here collision is followed by a sequence of force computation steps: penetration distance computation, force smoothing and force mapping. The difference from graphics computations is that these steps have to be executed an order of magnitude faster. While 30 Hz is a typical graphics refresh rate for interactive simulations, haptic simulators need to run hundreds of loops per second in order to produce realistic effects.

At the present time it is difficult to meet haptic interface performance, computational power and simulation modeling requirements for the most demanding medical applications, such as surgical simulators. Therapy and rehabilitation simulations however can use generic VR equipment and simulation techniques. The development of haptic

Virtual Environments for rehabilitation requires advances mostly in the area of haptic rendering. Therefore an important part of our research focused on developing haptic modeling techniques.

2.1.2 Virtual Environments for Therapy and Rehabilitation

An early therapy application of Virtual Reality was phobia treatment (North et al., 2000). The therapy repeatedly exposes the patient to a phobia-generating situation in a controlled environment. The method allows better control of the exposure therapy and more patient privacy and than in vivo techniques. The Virtual Environment is used to implement systematic desensitization for psychological disorders. Several VE applications were develop for specific phobias: fear of flying, fear of highs, acrophobia, fear of spiders - arachnophobia, fear of public speaking, etc. A review of Virtual Reality applications for treatment of psychological disorders is presented in (North et al., 2000).

More recently Virtual Reality technology is becoming a useful tool for the study, assessment and rehabilitation of cognitive processes and functional abilities (Rizzo et al., 2000). Virtual Environments technology offers clinical assessment and rehabilitation options not available in traditional neuropsychological therapy. Several applications were developed for impairments due to Traumatical Brain Injury, neurological disorders, and developmental/learning disabilities. The main VE application areas in clinical neuropsychology are:

- attention process: treatment of attention deficit disorders,

- executive functioning: regaining behavioral competencies related to planning, sequencing, ability to sustain attention, resistance to interference, coordination of multiple activities, etc.,
- memory: use VE training based on skill memory – concerning the capacity to learn rule-based or automatic procedures - in order to restore cognitive processes.
- spatial ability: improving spatial perception, orientation, visualization, etc.,
- functional skills and instrumental activities of daily living (IADL): training for functional skills and IADLs.

VE scenarios were developed to test and teach activities of daily living such as wheelchair navigation, street crossing, automobile driving, meal preparation, supermarket shopping, use of public transportation, etc.

The pilot studies of the applications reviewed in this section were executed typically under direct supervision of a therapist. Once proven effective, many of these treatment systems can be deployed at patient home. A telemedicine system will then allow the therapist to supervise and control the therapy or rehabilitation process. Such a system developed for orthopedic rehabilitation will be described in the next chapters.

2.2 Telemedicine Standards

Telemedicine is broadly defined as the use of medical information (i.e. high-resolution images, sounds, live video, and other patient related data) exchanged from one site to another via electronic communications (WebATA, 2000). The telecommunication media used for data transfer includes: POTS, ISDN, fractional to full T1, ATM, intranets, the Internet, etc. Telemedicine applications span several medical specialties, including dermatology, oncology, radiology, surgery, cardiology, psychiatry, rehabilitation and home health care. Among telemedicine main applications are:

- *Telementoring*: *teleeducation* of primary care physicians, residents, nurses, *teletraining*, *telesuport* of medical equipment and maintenance.
- *Teleconsultation*: consultation among physicians and other health care providers in various medical specialties such as radiology (*teleradiology*), pathology (*telepathology*), endoscopy and dermatology.
- *Telediagnosys*: providing services of specialty practitioners to remote locations, which do not have local expertise.
- *Telemonitoring*: remote monitoring of patient health; *telemetry: r*emote measurements of patient health condition
- *Teletherapy*: remote therapy
- *Telesurgery*: remote surgery

Telemedicine applications typically require exchange of patient specific information as well as multimedia data. Patient data include medical images and text formatted medical information (pulse, temperature, etc.). A prototypical telemedicine architecture for radiology is the Global Picture Archiving and Communication Systems (PACS) (Martinez et al., 1995). The Rural and Global PACS system is a wide area network that interconnects several PACS networks at medical and hospital complexes (Figure 2.1).

Each local PACS has a viewing workstations and database archive systems. The architecture can support Remote Consultation and Diagnosis (RCD) and videoconferencing services. The RCD application allows image annotation and voice communication. T3 connectivity (45 Mbps) is required for videoconferencing at 30 fps.

In order to provide interoperability and portability across heterogeneous platforms, the Global PACS was implemented using OSF/DCE - a Client/Server based distributed computing environment. This early work was continued with the development of a Virtual Radiology Environment (WebVRE, 2000). VRE is a large scale distributed system developed using Java IDLs and CORBA/DCE Middleware protocols. The system is currently tested over the Internet and MEDNET.

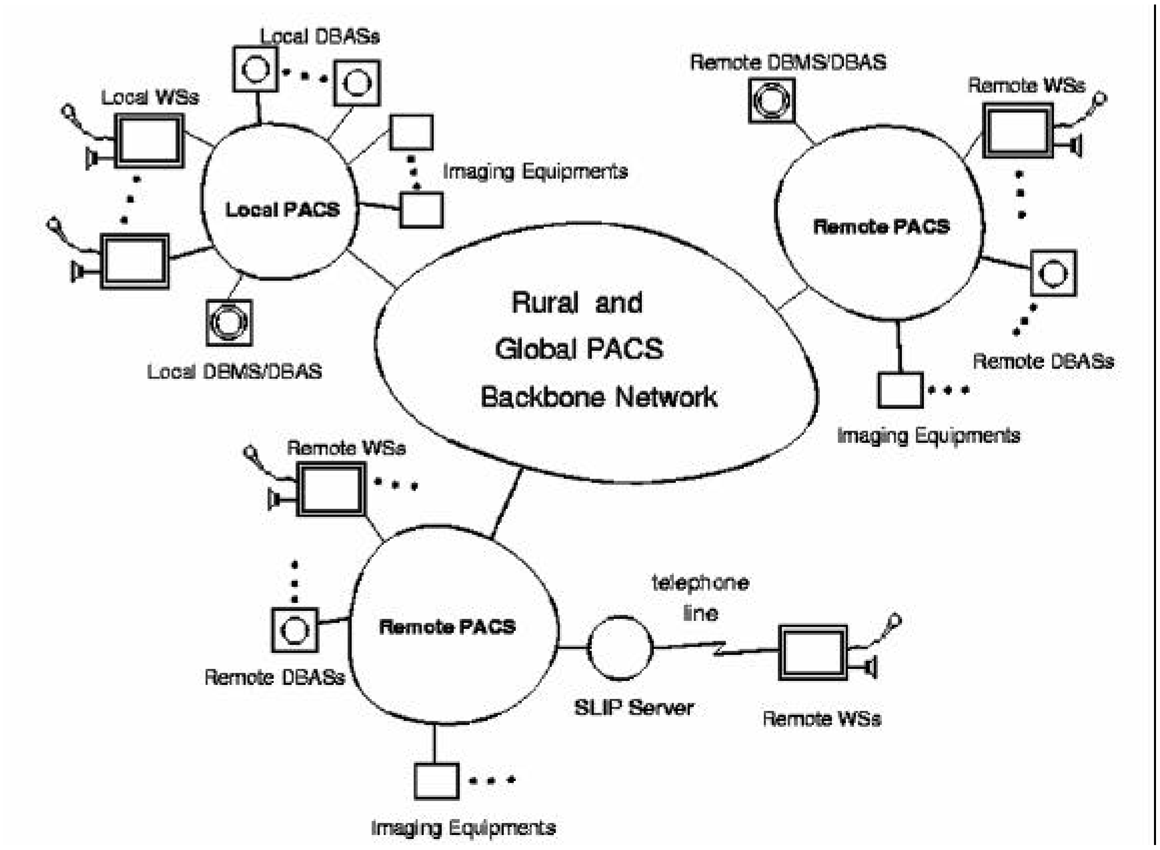
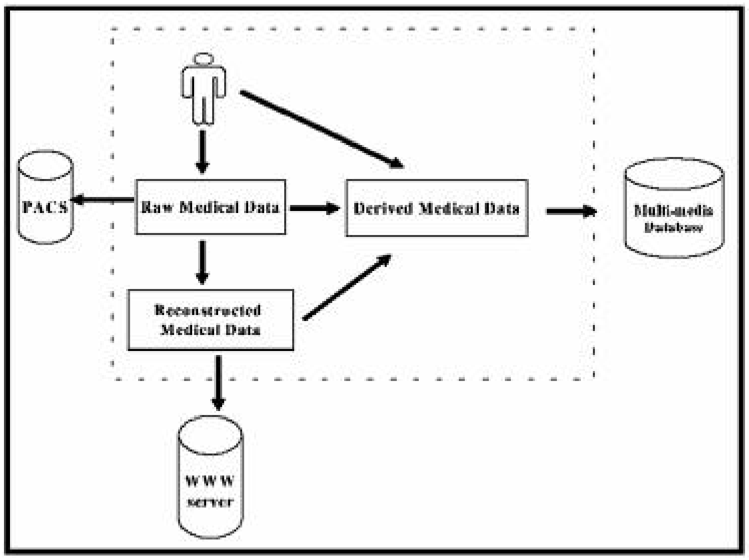


Figure 2.1 Rural and Global PACS

Several standards were developed for medical data interchange. Each of these standards focuses on a specific problem related to exchanging medical information (messaging framework, images transmission, Web accessibility). Health Level 7 (HL7) is an industry standard specification for electronic data exchange in healthcare (HL7, 2000). HL7 standardizes clinical, financial, administrative data interchange among independent healthcare computer systems (hospital information systems, pharmacy systems, etc.). It is widely implemented in the US, as well as in several other countries (Australia, Austria, Germany, Holland, Israel, Japan, New Zealand, and UK). The standard includes an ASCII based description of the syntax and message specifications. The latest version (HL7 3.0) utilizes a formalized methodology (which includes a variety of models) to create messages. The messaging framework uses XML and component technologies such as Corba and ActiveX.

An industry standard for transferal of radiologic images and related medical information is DICOM – "Digital Imaging and Communication in Medicine" (DICOM, 2000). DICOM deals with several aspects of medical imaging: image formats, information model, application services, communication protocol, and conformance specifications of the communication systems. The standard will lead workstations, CT scanners, MR imagers, film digitizers, shared archives, network printers to communicate by the means of an open-system network. An implementation of the standard is currently available in the public domain.

Another recent standard is Virtual Medical Worlds (VMW), (Marsh, 1999) developed by the EUROMED project. This standard is aiming at integrating telemedical services with the WWW. It uses the WWW as a navigational medium to remotely access medical information and to interface to several telemedical services. The telemedical services framework proposed defines 20 building blocks, among those being communication, accessibility, storage, privacy and security, navigation, transmission, telecollaboration and conferencing, visualization and manipulation, interaction and prediction, medical computing, compression, etc. The scope of the standard is limited, as it does not include services related to patient–physician interaction. The raw medical data (X_RAY, CT, MRI) is assumed to be stored in PACS, derived medical data (subsequent diagnosis data) is stored in multimedia databases, while reconstructed medical data (e.g. computer generated 3D models) resides on WWW servers. Data can be accessed through Web interfaces using plug-ins (e.g. VRML plug-in). VRML is used for 3D data visualization and manipulation. The standard is also trying to accommodate Virtual Reality interfaces and techniques (Figure 2.2-b) but the proposed use of VR is limited to data visualization and predicting medical scenarios.



a) b)

Figure 2.2: a) Storage components of WVM standard; b) Advanced interface for telemedicine (Marsh, 1999).

We believe that VR will have a more comprehensive role in telemedicine than to provide high-end interfaces for data visualization. One example is provided by VR-based telerehabilitation. The next section summarizes previous research in telerehabilitation, focusing on force feedback-based rehabilitation systems.

2.3 Virtual Reality for Telerehabilitation

Telerehabilitation is a form of telemedicine that provides remote rehabilitation services. Rosen (1999)-review defined telerehabilitation as the delivery of rehabilitation services at a distance by the means of electronic information and communication technologies. In this review he identifies five categories of telerehabilitation services: *home telerehab*, *home rehab teleguided*, *community telerehabilitation*, and *community rehabilitation teleguided* and *community practitioner teleconsultation*. Our work focuses on *home telerehabilitation*, which is the provision of rehabilitation services when the patient is located at home and the PT is remote. This is a more difficult approach, in view of the current technology, and no such systems were known at the time of the review, according to Rosen.

2.3.1 Telerehabilitation Research Motivation

The Research Planning report of the National Center for Medical Rehabilitation Research (part of NIH) indicates that in 1993 there were approximately 40 million disabled Americans (NCMRR, 1993). This staggering number includes people with restricted mobility, with reduced sensorial capacity, or with communication and intellectual deficits. The aging of the American population, coupled with the negative impact age has on disabilities (including recurrence of previously controlled conditions) has increased the number of disabled in recent years. Societal cost has similarly increased to 300 billion dollars according to a report of the Institute of Medicine (Brandt & Pope, 1997). The above cost does not account for the psychological impact on the disabled, on their family and environment, which further increase societal impact. While the number of patients needing rehabilitation (including long-term therapy) has increased, the resources available to them have unfortunately diminished, in part due to restrictions in managed health care agreements.

The reduction in the covered duration of therapy has thus a negative impact on the patient's condition, and on the recovery process. The duration of the rehabilitation therapy is important, as is timeliness of treatment. Indeed, assessment and therapy have to occur early on, else the same therapy duration will have diminished results. Timeliness and duration of rehabilitative therapy are problematic for those in remote rural locations or living in depressed urban areas. In such instances there are no clinics in the vicinity of the patient's home. Avoiding travel to the clinic altogether would mean that adequate therapeutic intervention could be done at home, after an initial assessment at the clinic. However, therapists may not be able to travel to the patient's remote home, or may be unwilling to do so. Another, more subtle, problem with current rehabilitation therapy is the patient's lack of motivation, stemming sometimes from lack of control over one's life, and over therapeutic choices in a clinic environment.

The leading cause of activity limitations for Americans is orthopedic impairments. Such patients typically follow a regimen of combined clinic and home rehabilitation exercises. Home exercises are done on simple mechanical systems that are loaned to the patient or constructed for them. Since these mechanical devices are not networked, there is no way a therapist can either monitor patient's progress at a distance, or change exercise difficulty levels remotely. There is also no way to verify that the patient has actually done the prescribed home rehabilitation exercises. Therefore, there is a need for a home telerehabilitation system that will record data from patient rehabilitation routines and will allow the therapist to remotely monitor the patient's progress.

Telerehabilitation major benefit is assuring timeliness of therapy for patients with mobility impairments located in remote areas (Burdea, 1998-b). Additional benefits are related to the quality of the rehabilitation therapy, since one therapist could monitor and work with several patients almost at the same time. Providing home-based therapy without home care therapists should reduce health care costs, while maintaining quality.

Telerehabilitation also benefits the patient's motivation, a major component of the recovery process, by increasing his/her sense of empowerment when performing rehabilitation under one's control, and in one's own home. The use of virtual reality simulations is a powerful motivating tool for patients, as shown by earlier pilot trials (Burdea et al., 1997-a). Furthermore, by placing the system in the patient's home we maximize family support, and family/patient education, which are important components of the rehabilitation process. A key component of the patient's education is knowledge about his/her impairment and loss of function and available therapies.

Last, but not least, the Institute of Medicine (Brandt and Pope, 1997) report recognizes that there is little theoretical structure within the rehabilitation science, and empiricism predominates. What results is a lack of standards and objective measures to quantify the enabling process (reversing disabling conditions). There is also a lack of repeatability within the assessment and therapeutic intervention process. Each therapist will read a given instrument differently (in part due to inadequate technology), and each therapist will have a subjective measure of patient's function. Telerehabilitation should respond to these problems by forcing structure and standards. Furthermore, through the use of advance technologies it should improve the lack of repeatability present in today's rehabilitation science.

2.3.2 Computer-Based Patient Monitoring Systems

Historically, computer-based biomechanical evaluation tools were first used for monitoring the rehabilitation process. Greenleaf Medical developed "Eval" and "Orca" systems for orthopedic evaluation (Fox, 1991), (Greenleaf Medical Systems, 1997). The systems offer easy data collection and storage and tools for analyzing the patient information stored in the database. Other companies (Lafayette Instrument Company (WebLafayette, 2000), Electronic Healthcare Systems, Inc., (WebEHS, 2000)) are offering software for patient monitoring and evaluation. Data are stored in custom databases and patient reports can be displayed. The systems described above were designed to be used in the clinic so that they don't include either a networking or a rehabilitation component. No forces are applied on the patient by these devices.

An example of a system for computer-based patient monitoring and remote evaluation is the "Electronic House Call" (EHC) (Peifer et al., 1998) developed by Georgia Institute of Technology in collaboration with the Medical College of Georgia and the Eisenhower Army Medical Center. Six at-home patient measurements were demonstrated, with data stored at the clinic using a Client/Server database architecture. A Web-based platform for patient rehabilitation progress evaluation was recently developed by Grimes et al. (2000). The physician uses NetMeeting in order to instruct the patient during the data collection process. Data collected by a gripmeter attached to a home PC is transmitted remotely to the hospital patient's record. The results show the same quality of the measurements obtained using the remote system as when used in a patient face to face visit. In the area of home rehabilitation Ward and Bullinger patented a system where a remote clinician can monitor and set the range of motion of body joints through a "dual-plane joint monitor" (Ward & Bullinger, 1998). There is no virtual reality component to their proposed system and no forces are measured or applied by the patented apparatus.

Telerehabilitation research was recently boosted by an NIDRR initiative (NCDRR, 1998). The initiative sponsored the Rehabilitation Engineering Research Center, a joint research effort by Catholic University of America, National Rehabilitation Hospital, Sister Kenny Rehabilitation Services, and various other collaborators (WebRERC, 2000). The center conducts research on four priority areas: 1) telecommunication for rehab education, taining & support; 2) telemonitoring & teleassessment tools; 3) teletherapy for rural/remote settings and 4) virtual reality for rehabilitation. Virtual reality reasearch focused on rehabilitation of cognitive and perceptual impairments (Trepagnier, 1999). Current research projects involving VR are: 1) Prototype Personal Augmentation Devices (PADS); 2) Rehabilitation Telemonitoring; 3) Interactive Systems for Provision of Therapy, Assessment, Teaching and Demonstration at Home; 4) Investigation of Face Gaze Behavior in Autism and Right Hemisphere Stroke (Trepagnier et al., 1999).

Preliminary results were encouraging and the projects are expected to move into the patient trial stage.

The telerehabilitation program at the Sheperd Center in Atlanta focuses on augmentative communication devices, computer access equipment, home modifications for telemedicine, and the use of telerehabilitation for the community-based care of patients with Acquired Brain Injury. The main technology being used is videoconferencing, as it is the most accessible to the disabled. Recently the center start developing a specialized computer network for people with disabilities.

The Missouri telerehabilitation training program involves the use of teleconferencing equipment to train community specialists in the follow up treatment of patients with Traumatic Brain Injury. This facilitates continuity of care once the patient return to the community and behavioral health services are needed. Communication is done over T-1, which allows real-time video/audio between several sites on the network.

Despite the ongoing research efforts, telerehabilitation activity represents a very small fraction of currently deployed telemedicine systems. A recent, the ATSP survey (ATSP, 1999) reported that only 1.2% of the listed clinical services were rehabilitation, while other small percentages were obtained for home health, remote site evaluation (rehab.), etc. Only three (2.2%) of the clinical sites listed the rehabilitation as the most active clinical specialty and thirteen programs (out of 1,122) listed telerehabilitation as a provided clinical service. None of the reported systems uses force feedback for orthopedic telerehabilitation.

2.3.3 Force Feedback – Based Rehabilitation

As we have seen in previous sections many VR applications were developed for treatment and rehabilitation of neuropsychological disorders. Much less work was done for orthopedic rehabilitation systems, since this require the addition of haptic interfaces and physical simulation modeling. Previous work related to force feedback for rehabilitation includes prototype systems that provide forces for manual therapy developed by Hogan at MIT (Krebs et al., 1996), Luecke at Iowa State University (Luecke et al., 1996), Takeda and Tsutsul at Nagasaki Institute Applied Science (Takeda & Tsutsul, 1993), and more recently Rovetta at the Milano Politechnic Institute (Rovetta et al., 1998). All these prototypes have certain advantages versus the clinical practice. For example, the MIT system showed faster upper limb motor rehabilitation for stroke patients who exercised with a robot. The Iowa State system allowed independent force control for each finger, while the Nagasaki system was extremely light and powerful through the use of pneumatic “muscle“ actuators. The Milano Politecnic system is portable (uses a laptop), and is intended for patients that need neuromotor rehabilitation (such as those with Parkinson's disease). However, all the systems cited above have also drawbacks, due mainly to their complexity (for example the use of robot manipulators) making them difficult for use at home. In the case of the Milano Politecnic system, forces to only one finger are measured, and only one virtual finger is shown. Furthermore, there is no networking component in either of these systems, such that at-home monitored rehabilitation is not possible.

A VR-based system for hand rehabilitation was also developed by Burdea and colleagues (Burdea et al., 1997-a), (Burdea et al., 1997-b). The system differs from the other prototypes mentioned above as it includes a diagnosis module (with standard diagnosis instruments), a rehabilitation module using VR simulations and the Rutgers Master I haptic glove (Burdea et al. 1992). Proof-of-concept trials done with a small group of patients were promising especially for the subjective evaluation of the system by the patients. Problems remained due to the DataGlove technology used at the time for hand readings, as well as the slow graphics workstation used (Sun 10-Zx). This system, like the ones before, was not networked, as it was intended for clinic, rather than at-home use. The next step in the evolution of the above rehabilitation system was the development of networking architecture to support remote patient data collection and monitoring and therapist-patient interaction, as described in this thesis.

2.4 Conclusions

Medicine is a major application domain for Virtual Reality. VR applications in medical therapy include phobia treatment, and rehabilitation of cognitive processes. Less work was done to apply Virtual Reality to orthopedic telerehabilitation. Early experiments in orthopedic rehabilitation use force feedback systems to train patient muscles. In addition, networking solutions for home telecare were illustrated by several home-based patient monitoring systems. However there is little work on the architectural design of VR-based telemedicine systems. We present in the next chapters the prototype of a VR-based home rehabilitation system networked with the clinic.

Chapter 3

A Virtual Reality Platform for Telerehabilitation

A typical telemedicine system requires networked workstations able to handle video/audio (MPEG) and still image compression (JPEG) (Kim et al., 1995). In addition to audio/video equipment, VR-based home rehabilitation needs a powerful graphics card in order to run 3D applications at interactive frame rates. Haptic interfaces should be added for orthopedic rehabilitation, in order to measure joint motion and provide forces for muscle and task training. In the following sections we will present a generic PC-based platform and haptic interfaces designed for telerehabilitation.

3.1 The PC-based Telerehabilitation System

The prototype of the home rehabilitation system is shown in Figure 3.1 (Popescu et al., 1999-a). It consists of a PC equipped with an InsideTrack 3D tracker (Polhemus, 1993), a FireGL 4000 graphics accelerator, a microphone array developed at the CAIP Center (Lin et al., 1996), and a net camera. A multiplexed haptic control interface is connected to the PC serial port. The Rutgers Master II hand master is used to control a virtual hand and to provide force feedback to the patient's fingers.

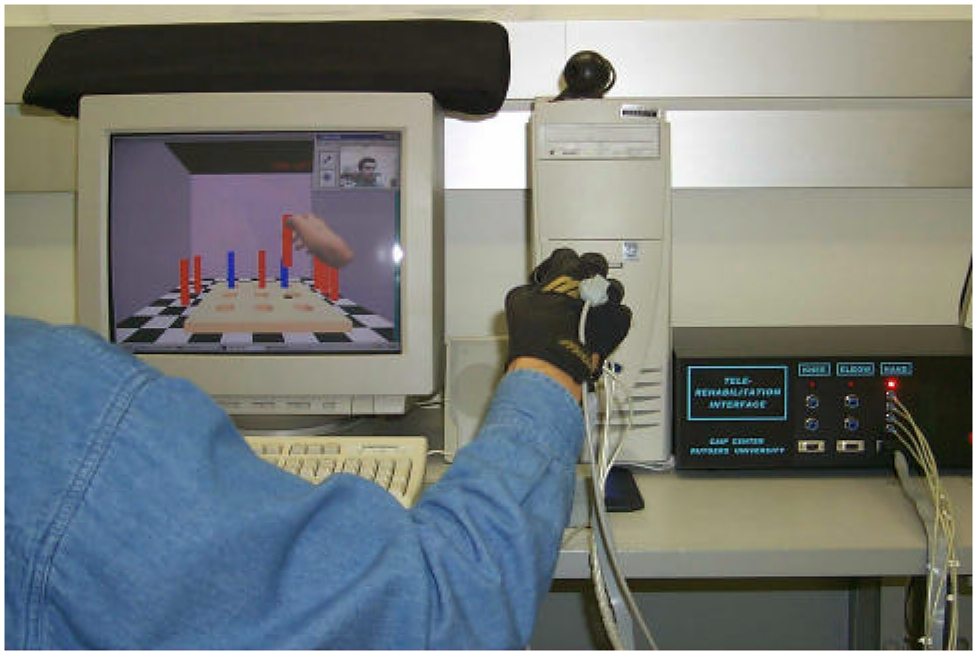


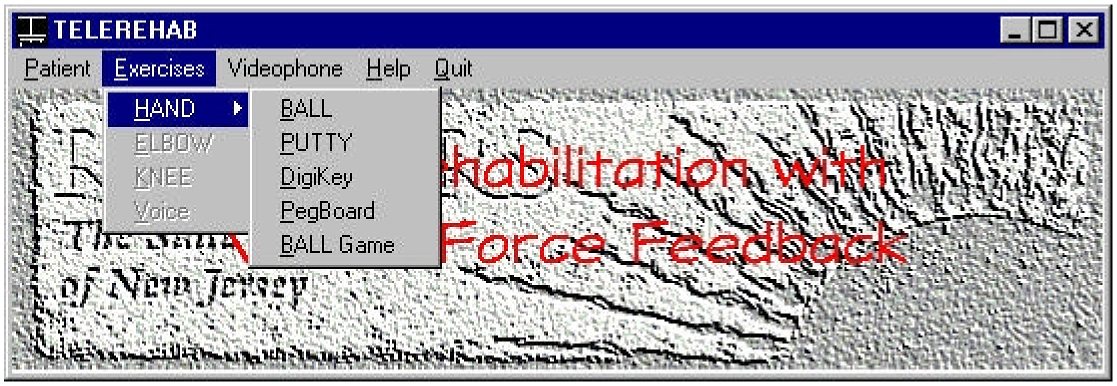
Figure 3.1: Telerehabilitation workstation experimental prototype (Popescu et al., 1999-a)

The microphone array provides hands free voice input by focusing on the patient's head siting approximately three feet in front of the monitor. The spatial localization of the microphone array is useful to filter out noises in the room and reflections from the PC speakers. This filtering is beneficial for speech recognition. The speech input is very useful for orthopedic rehabilitation. During rehabilitation exercises, the patient is wearing haptic devices, which make the operation of a keyboard and mouse awkward.

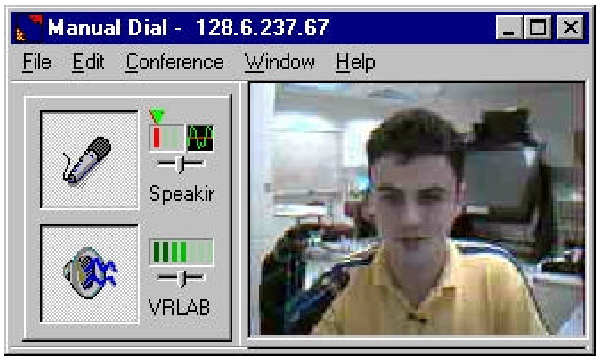
We experimented with a speech recognition system with a small set of commands: {"Show database", "Calibrate glove", "Start EXERCISE", "Start consultation", "Show help", "Start EXERCISE tutorial", where EXERCISE = "Digikey" | "Ball" | "Putty" | "Peg Board" | "Ball Game"}. The speech interface uses Whisper recognition engine

(Microsoft Co., 1998) with a finite-state grammar and a restricted vocabulary. The speech recognition platform with a small vocabulary proved to be robust in previous applications (Popescu & Burdea, 1998-b), (Medl et al., 1998), (Popescu et al., 2000-a). In these applications the recognizer runs on a separate PC and communicates though sockets with the main application. However, a distributed system is not practical for home rehabilitation; all software should run on a single computer. Unfortunately the speech recognition is CPU intensive and cannot run at this time in parallel with an interactive 3D graphics program. Instead a Windows NT menu style GUI was used to control the application. The telerehabilitation graphics interface is shown in Figure 3.2-a.

A net color camera connected to the PC is used for videoconferencing with the clinic. A parallel port camera can provide up to 15 fps QCIF (**Q**uarter **C**ommon **I**ntermediate **F**ormat) images when running on a local machine. A PCI card camera can provide 30 fps QCIF and includes hardware compression for image and audio. A duplex sound card supports I/O audio for videoconferencing and VR simulation sound. The patient-clinician videoconferencing uses "CuSeeMe" software (WhitePine, 1997) with MPEG or H.263 (ITU, 1998-b) video codec and 8Kbps audio (see Figure 3.2-b). The videoconferencing was not integrated at the present time with the Virtual Reality simulation due to the slow PC CPU (reduce by half the frame rate when running simultaneously with the VR exercises). Thus videoconferencing runs on a separate window, before or after the VR exercise. The NetMeeting 3.0 SDK (Microsoft Co., 1999) will be used in the future to allow integration of an H.323-based (ITU, 1998-a) (DataBeam, 1998) videoconferencing system in the VE.



a)



b)

Figure 3.2. a) Telerehabilitation GUI; b) Videoconferencing using "CuSeeMe" (WhitePine, 1997)

The Pentium PC is connected to a novel Haptic Control Interface (R-HCI) which was designed to drive rehabilitation haptic interfaces for the hand, elbow and knee (Figure 3.3). The R-HCI is a newly designed controller interface for pneumatic systems. It contains an upgraded embedded PC, a custom pressure regulator and multiplexing hardware. It can switch between the hand, and future elbow and knee haptic devices seamlessly, as required by the VR exercise routine to be executed. The system is self-configurable, depending on the patient's needs, without any hardware changes (connect, disconnect, etc.). Currently the system is used with the Rutgers Master II haptic glove while the elbow and the knee units are under development.

The haptic devices are mainly designed for orthopedic rehabilitation, even though they may be used for other rehabilitation applications (e.g. treating and diagnosis of neuromotor disorders). The hand haptic device, the RMII Hand Master, can provide 1DOF force per finger (excluding the small finger) opposing flexion motion. It can be used to improve fingers' strength as well as whole arm mobility. The elbow and knee units are 2DOF devices displaying forces opposing flexion-extension motion. The "Rutgers Ankle" is an electro-pneumatically controlled Stewart platform that apply forces and torques to patient's foot. The platform allows the control of forces and torques in six DOFs and movement throughout the ankle's full range of motion.

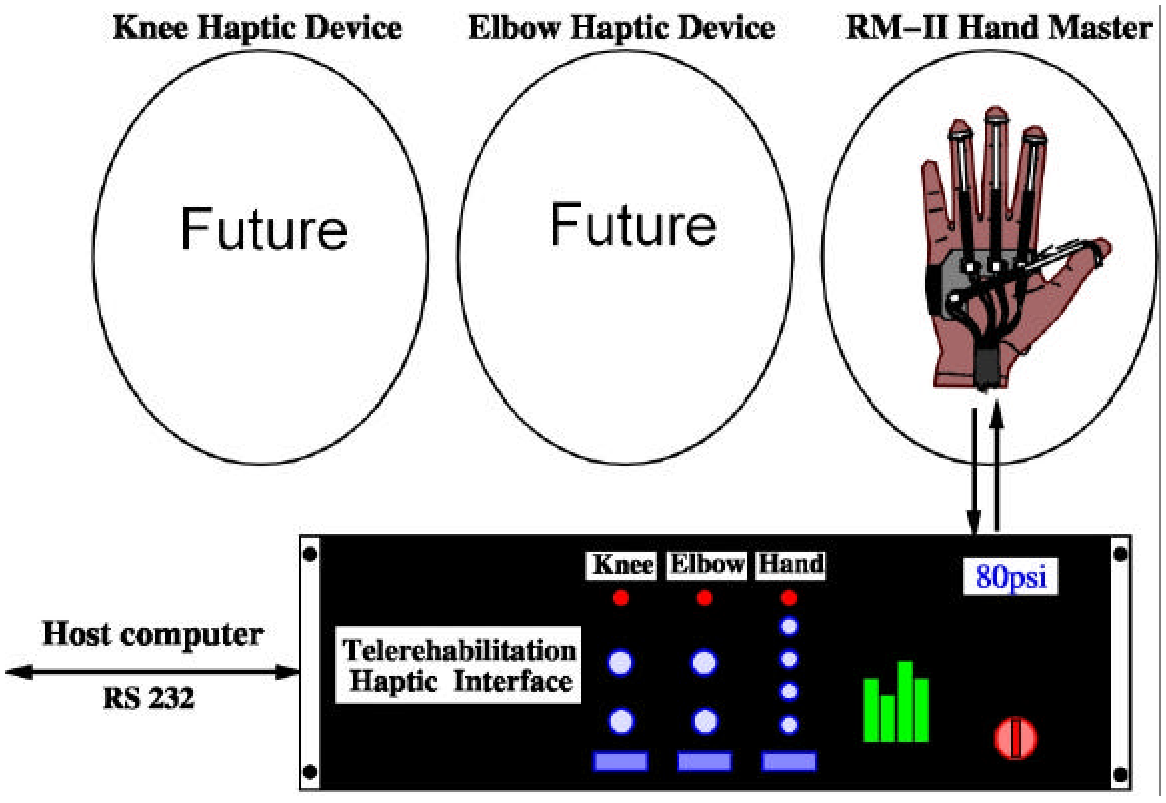


Figure 3.3: The RM II connected to the Haptic Control Interface (Popescu et al., 1999-a).

3.2 Haptic Interfaces

3.2.1 The RMII Haptic Interface

As shown in Figure 3.4, the RMII-ND (Bouzit et al., 2000) glove is an exoskeletal structure that provides forces at the patient's fingertips and contains its own non-contact position sensors. Thus, the system is simplified (no need for a separate sensing glove) and light (about 100 grams). This lightweight makes the master glove as comfortable as a lather glove. It is also beneficial to the patients with hand injury as it reduces fatigue. The shape of the palm-base is optimally designed to fit comfortably in the user's palm. The palm-base is attached behind the “middle-line” of the palm allowing a complete flexion of the metacarpal phalanx. The inside of the palm-base contains four pneumatic tubes and a sensor electronic board. Highly flexible and lightweight PVC plastic tubing is used to provide air pressure to the actuator from the palm-base. The deformation of this pneumatic tube causes a negligible resistive torque at the user’s fingertip.

The feedback actuators have glass/graphite structures with very low static friction. The combination of high, sustained feedback forces (16 N at each fingertip), and low friction provides for a high dynamic range (300). This makes the RMII capable of high sensitivity and resolution in the range of feedback forces it can produce.

The actuator flexion (relative to the palm-base) motion varies from -10 degrees to 110 degrees, which is equivalent to a natural flexion of a proximal finger joint. The actuator abduction/adduction motion is not constrained, following natural finger motion. The linear motion of the piston (stroke) varies from 27 mm to 48 mm depending on finger size. The range of the linear motion allows a maximal flexion angle equal to 45deg for a second and third finger joints, which represents typically 55% of the natural motion. This is a limitation when performing hand rehabilitation. Another design limitation is that the “small” finger is not provided with force feedback/position measurement.

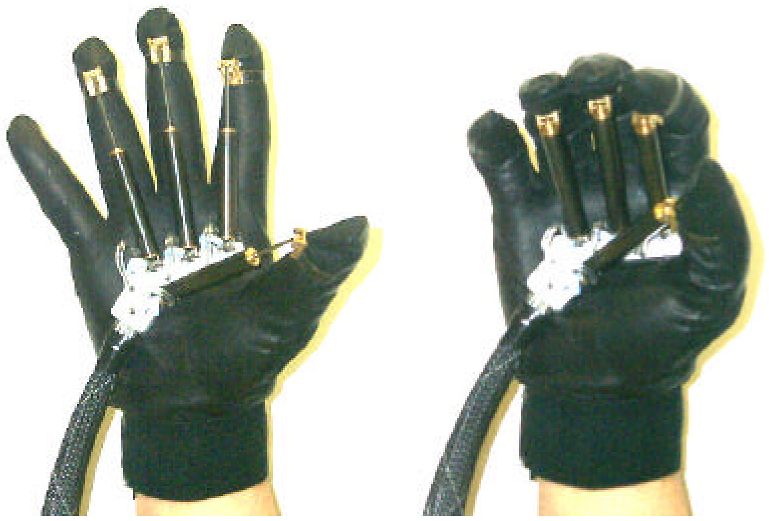


Figure 3.4: The RMII-ND master glove, a) Open hand; b) Closed hand.

3.2.2 The Elbow Interface

Designing a haptic device for the elbow is the next step toward whole arm force feedback for VR simulations. The elbow joint has only one degree of freedom of “hinge” type (the radioulnar motion is not considered here). The hinge joint allows only planar movement (flexion, extension) and is simple to work with. The average range of motion for the male elbow flexion from Airforce Personnel Data (NIST, 2000) is 142 degree, and the standard deviation is 10 degrees. The elbow extension is about 0 degrees. For women the range of motion is increased on average by 8 degrees.

When designing the elbow device we are primarily interested in the characteristics of the flexor muscles. The maximum torque is produced by flexor with the elbow flexion at 90 degrees. For this position, studies show that for this flexion angle the peak pulling force is about 230 N for women and 370 N for men. For rehabilitation purposes one third of peak force should be sufficient for muscle training. Therefore the haptic interface sustained force output should be around 120 N.

The actuator chosen for the elbow interface is the McKibben Artificial muscle. The actuator is composed of a rubber tube covered in tough plastic netting, which shortens in length when the rubber tube is inflated with compressed air at 1 to 6 bar. The muscle is compliant, being easy to use in different configurations. The actuator is light and has similar characteristics with the biological muscle. A detailed analysis of the actuator is presented in (Chou & Hannaford, 1996).

The maximum tension for air muscles reported in (Chou & Hannaford, 1996) was 110N at 5 bar. One muscle is enough to meet minimum requirements, but two pneumatic muscles can simulate peak forces applied to the patient’s elbow. A trade off between maximum applied force and system bandwidth has to be made. Increasing the muscle section we can obtain higher maximum tensions, but we reduce the bandwidth of the system (the valve has to control a higher volume of air). Two groups of air muscles control the elbow as shown in Figure 3.5. The two muscles control the flexion and extension of the elbow. Their control commands are opposite. When one is inflated the other is deflated. This dual control also increases system bandwidth.

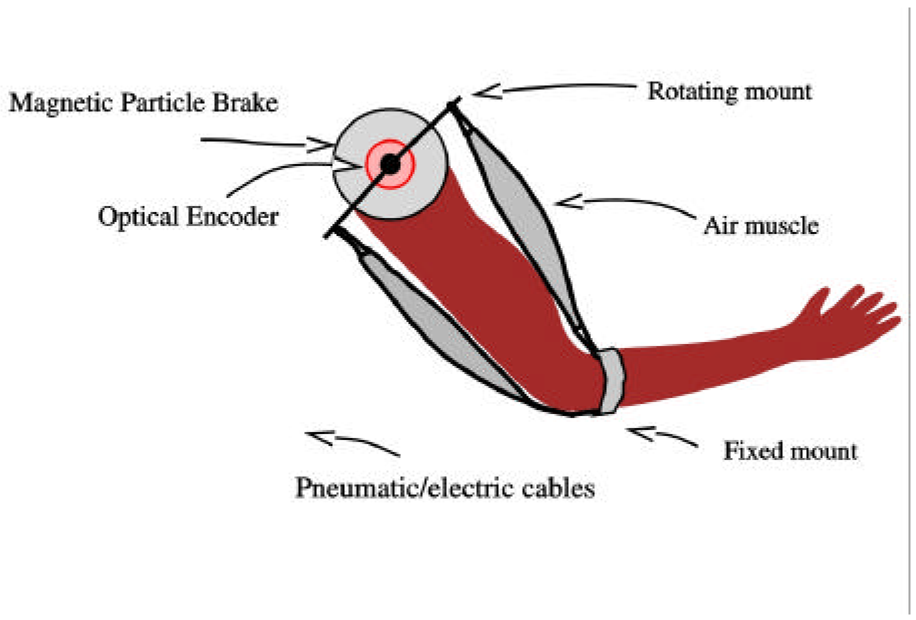


Figure 3.5: The elbow haptic interface

One end of the air muscle is attached to the rotary mount, which includes a magnetic particle brake (Burdea, 1996). A light spring can be used to tension the air muscle. When a force feedback command is issued (the intake valve of the pressure regulator is opened) the brake is applied on the rotating mount. The brake is released when the force feedback is removed. The MPB is a passive element that can be used to simulate stiff objects (walls) and to increase the work envelope of the device. It operates at high frequency and can support well high torques, but it is affected by hysteresis. The air muscle is the dynamic element used to apply force feedback to the elbow.

3.2.3 Why Pneumatic Actuators?

The haptic interfaces presented above use pneumatic actuators controlled by embedded PC and A/D/A cards. The choice of pneumatic actuators for VR-based rehabilitation can be explained by the analysis of three types of actuators presented in (Burdea, 1996).

Hydraulic actuators are too heavy to be used for portable haptic interfaces. In addition they might not comply with safety rules for user interfaces. DC motors are not suited to be used to provide forces of hundreds of newtons because of their low power/weight ratio and overheating. In addition, for orthopedic rehabilitation the high mechanical bandwidth, which is the main advantage of DC motors-based actuators, is not that important. After all, in orthopedic rehabilitation the haptic interfaces are used as body workout devices. Safety and cleanliness are more important. Therefore the DC motors-based systems are at a disadvantage when used as haptic interfaces for rehabilitation. Pneumatic actuators have a high power-to-weight ratio and can be used in clean environments. Due to air compressibility, they are compliant and safe to use in medical equipment.

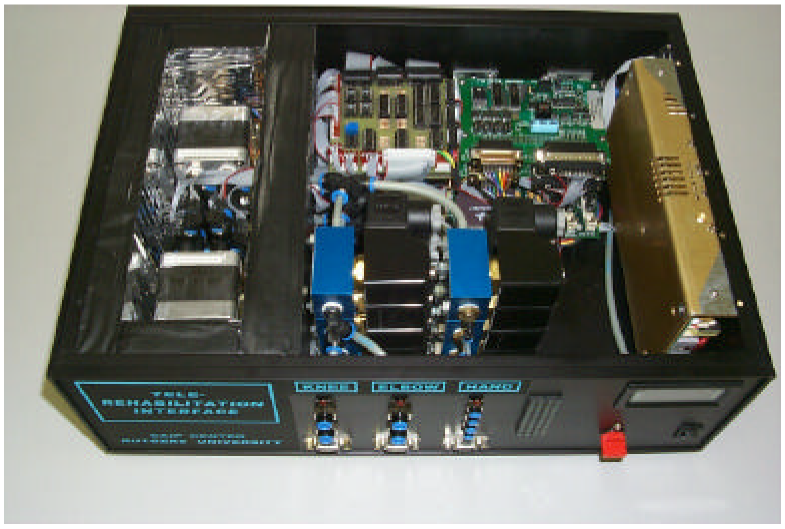
Several types of pneumatic actuators are available. One of the best known pneumatic actuators available is the McKibben Artificial muscle, presented in the previous section. The design of elbow actuator is a variation of this artificial muscle. Another actuator is the pneumatic cylinder. Commercially available actuators from Airpot (Airpot Co., 2000) as well as custom designed ones were used for RMII and Rutgers Ankle interfaces. This type of actuator has a very good dynamic range due to low friction, no hysteresis and higher mechanical bandwidth. Mechanical and dynamic characteristics of the muscles dictate the type of actuators to be used. High force output and low bandwidth artificial muscles are good for elbow and knee interfaces while high bandwidth, low friction pneumatic cylinders are suited for hand force feedback. Combinations of different pneumatic actuators should be used in the design of a future whole body haptic suit for rehabilitation.

3.3 The Haptic Control Interface

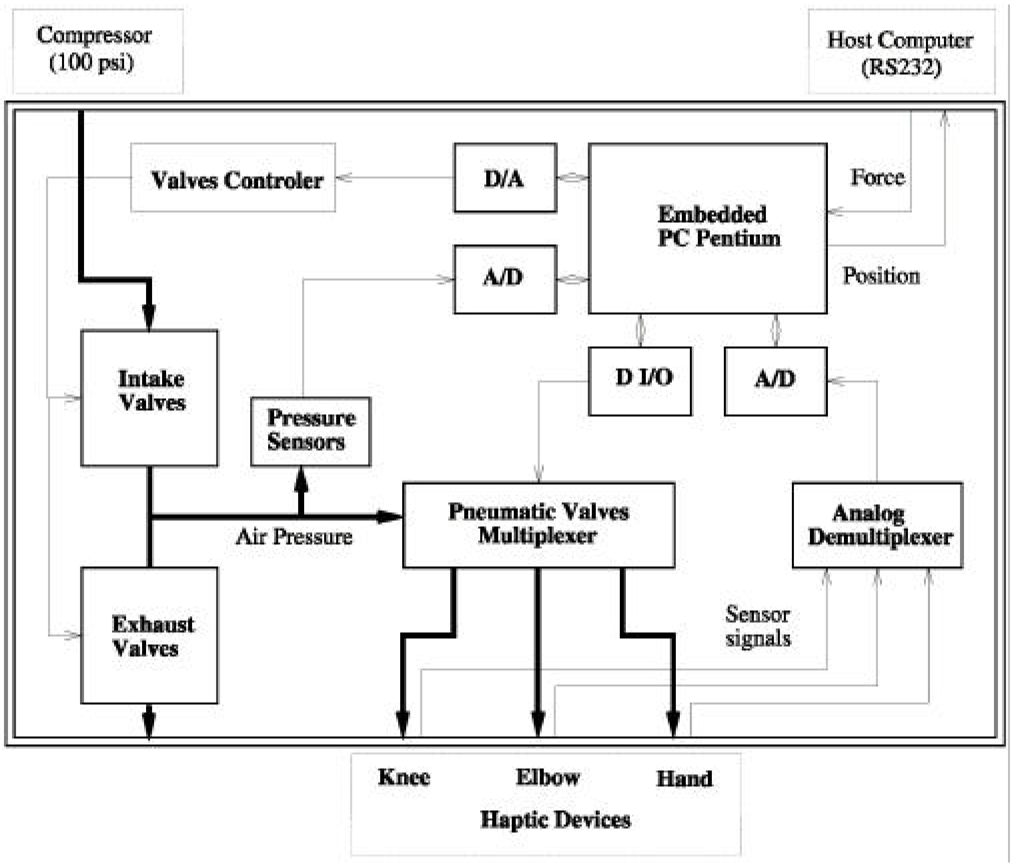
3.3.1 Control Interface Hardware

The hardware of the Electronic Control Interface is composed of an embedded Pentium PC, pneumatic valves and electronic boards for sensors reading and pressure control (see Figure 3.6). The embedded PC is a 233 MHz Pentium board with a PC104 bus. An A/D/A board (MPC550 from Micro/Sys) with sixteen input/eight output channels is mounted on the PC104 bus. Twelve A/D inputs are used for reading sensor data from the haptic glove (three inputs per finger), while four A/D read control loop pressure sensors. Half of the output D/A channels control the intake valves, while the other half control the exhaust ones.

The custom electronic boards perform filtering, amplification and multiplexing of the analog signals. Signals from IR and Hall effect sensors mounted on the haptic glove are amplified and filtered before they are feed into the A/D board. Pressure sensor signals go to a bridge amplifier before the A/D conversion. D/A outputs are amplified before being connected to a valve control board, which uses power amplifiers. In order to switch between knee, elbow and hand units, the Haptic Control Interface needs to multiplex the pressure lines and to demultiplex the input lines from the sensorized interfaces. Signal demultiplexing is handled by an electronic board using PC digital I/O signals. Pressure line multiplexing uses a set of switching valves. The valves are controlled by digital I/O signals from the Pentium board amplified on the same optional electronic board.

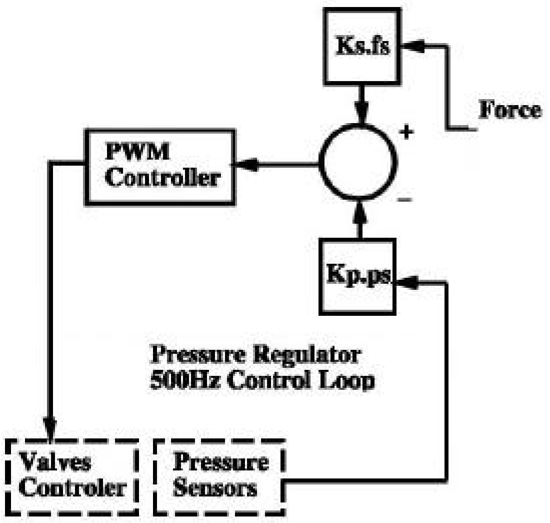


a)

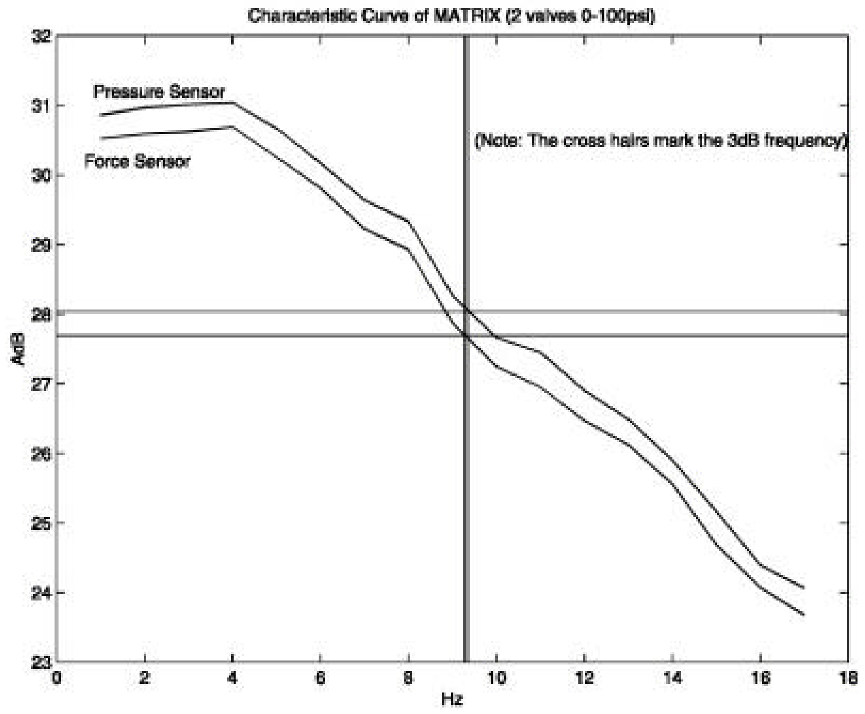


b)

Figure 3.6: The Haptic Control Interface: a) upper view of the interface; b) functional diagram (Popescu et al., 2000-b, © IEEE).



a)



b)

Figure 3.7: a) Pressure regulator control loop; b) characteristic curve of pressure regulator with two valves (adapted from Patounakis et al., 1998).

The pneumatic valves were carefully selected regarding their response time and the airflow on which haptic interface bandwidth is highly dependent. The solenoid valves operate a frequency of 500 Hz with a flow of 200 Nl/min. The intake valves have one input and eight outputs (and eight inputs one output for the exhaust ones). The pressure regulator was built with two of these fast valves (exhaust and intake) equipped with a pressure sensor and software controlled by software running on the Pentium board.

The control loop software of the pressure regulator uses a PWM technique (Figure 3.7-a). Experimental results show that the close loop control using PWM technique achieves a precise and stable output pressure. An experimental setup consisting of a software controlled pressure regulator, an RMII piston and a load cell was used to test the mechanical bandwidth obtained with the solenoid valves. The load cell was mounted at one end of the RMII piston to record the force felt at the fingertip. The characteristic curve of the pressure regulator using two valves per finger is shown in Figure 3.7-b. As can be seen, the mechanical bandwidth when using two valves per finger is about 10 Hz. This bandwidth is sufficient for slow to medium velocity finger movements as encountered in rehabilitation exercises.

3.3.2 Software Driver and Communication Protocol

The imbedded PC runs a continuous loop (Figure 3.8) which executes the following tasks:

- read and filter the data from the sensors and then transform them to a user joint data;

- communicate with the host computer by sending the requested data (joint, forces, device state, etc);
- run the PWM software control loop.

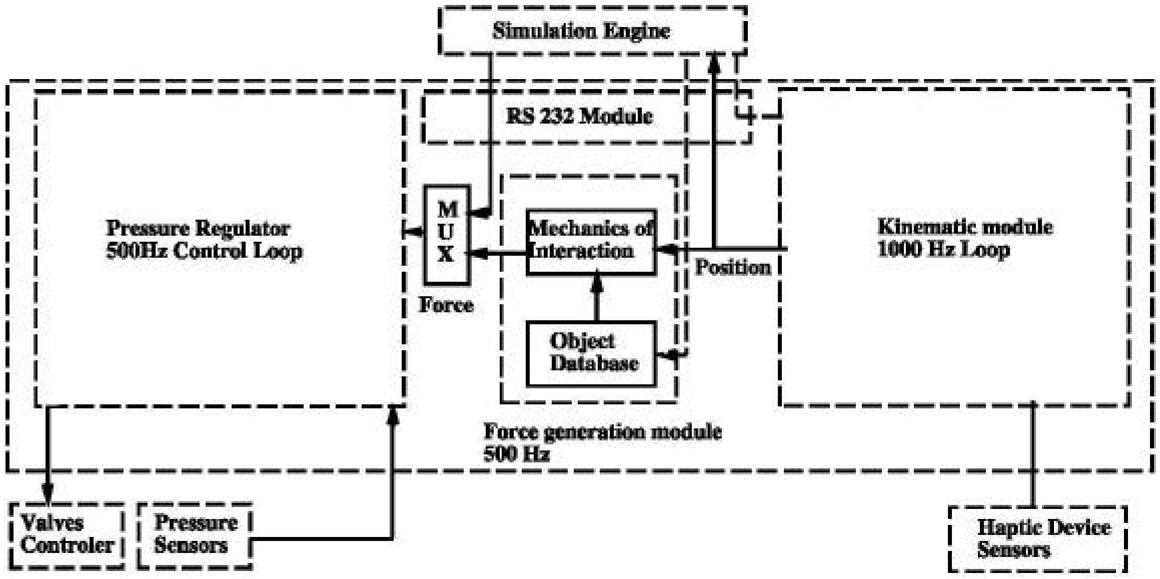


Figure 3.8: Real-time software of the Haptic Control Interface

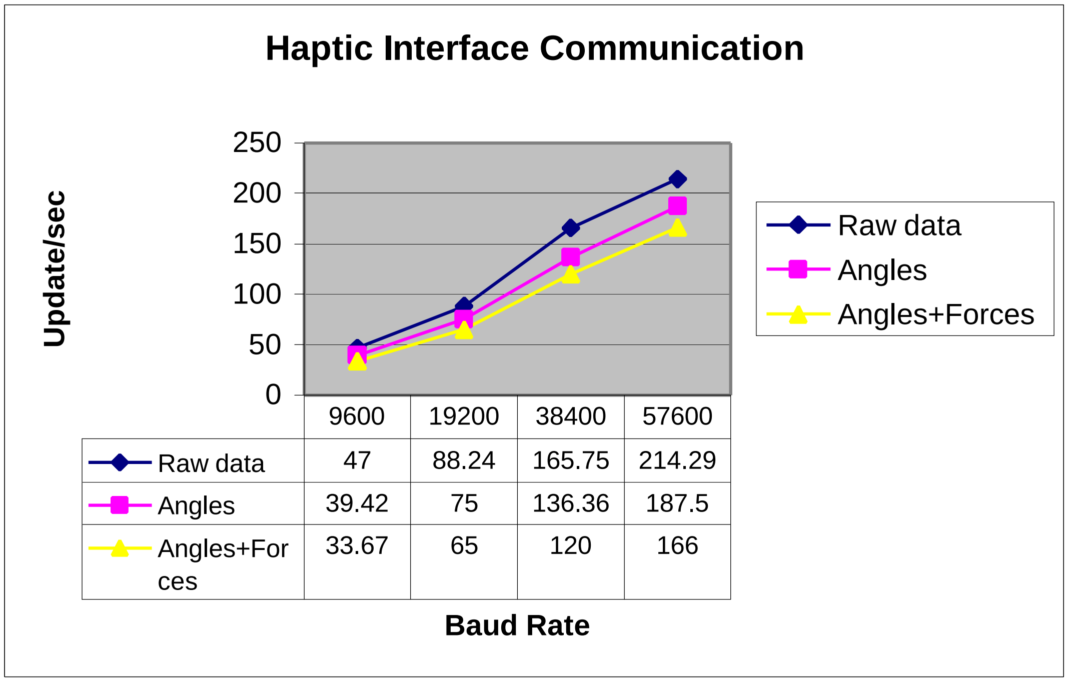
The input data (IR and Hall sensors) are read at a frequency of about 1000 updates per second and filtered with an averaging filter to eliminate electric noise. Input data are sub-sampled since the requests from the host computer come at a frequency of less than 300 updates/second. The communication with the host computer uses an RS232 serial port, with baud-rate settings ranging between 19,200 BPS to 115,200 BPS. The software running on the embedded Pentium transforms the device sensor input data into user's hand joint angles.

The communication between the control interface and the host computer is done through an RS232 port. The data sent to the host computer consists in joint angles, raw sensor measurements (displacement, flexion angle and abduction angle), measured forces or device states. The data received from the host computer contains commands for retrieving data, applying forces and for changing the functioning mode.

The communication is based on a request-answer protocol. The interface waits for a request from the host computer, serves it and then waits for the next one. The most frequent requests are those asking for joint angles, raw sensor readings and force readings. The communication flow is asymmetric. Much more data go to the control interface than to the host computer. In the continuous loop the size of a package sent from the host computer to the interface varies between one and six bytes, while a package sent from the control interface to the host computer has fourteen to twenty bytes. The size of the data packages sent during calibration is fourteen bytes; command requests require between one and fourteen bytes.

The quantity of data sets sent and received per second varies depending on the serial port settings. The speed of the serial communication is settable at run-time, the baudrate ranging from 9,600 to 57,600 BPS. In Figure 3.9 we present the performance obtained for different baud-rates on a single processor Pentium II at 300 MHz.

At a rate of 57,600 BPS the RS232 line can transmit up to 187 RMII position data sets/sec or 166 data sets that contain both finger positions and force readings every second. This is comparable to the data rate of Cyberglove (149 updates/sec at 115200 BPS), while being much smaller that that of a Phantom device (~1000 updates/sec) interfaced with a PCI card.



	9600	19200	38400	57600
Raw data	47	88.24	165.75	214.29
Angles	39.42	75	136.36	187.5
Angles+Forces	33.67	65	120	166

Figure 3.9: RMII serial communication performance

3.4 Conclusions

This chapter presented the components of a Virtual Reality based orthopedic rehabilitation system. The system includes a PC with a graphic accelerator, a videoconferencing system and a haptic interface. Our efforts focused on developing the haptic interface electronics and integrating it with commercial available systems. Two haptic interfaces were tested for our system: a force feedback glove (RMII-ND) and an ankle force feedback system (described later in this thesis). The hand unit is driven by a multiplexed control unit, which can also control elbow and knee haptic interfaces (currently under development). The performance study (mechanical bandwidth, maximum applied force, etc.) of the haptic systems presented above showed that they are suited for rehabilitation purposes.

Chapter 4

Haptic Interaction Modeling

4.1 Haptic Rendering

Haptic (force and touch) feedback is a recent addition to Virtual Reality simulations (Burdea, 1996). This sensorial modality increases the simulation realism during virtual object manipulation. The addition of haptics requires several enhancements to the typical VR simulation. First the system needs a dedicated haptic interface which transmits the simulation feedback to some part of the user's body (usually the hand). Most of today's interfaces are joysticks, or small robotic arms. Second, physical modeling tools need to be developed, containing a library of functions to calculate and replicate contact forces, surface deformation, rugosity, object weight, etc.

One of the most successful haptic interfaces commercially available is the PHANToM (Massie and Salisbury, 1994). This small robotic arm provides forces (but no torques) to the user's index finger. Therefore most of the research on modeling the PHANToM are concerned with only the end effector point of contact (Ziles and Salisbury, 1995) (Ho et al., 1997). The state of the art of haptic interaction modeling is reviewed in (Srinivasan and Basdogan, 1997).

A more complex category of haptic interfaces is haptic gloves. Several haptic gloves were designed for VR simulations: the "Rutgers Master" (Bouzit et al., 2000), the LRP Glove (Bouzit, 1996), the "CyberGrasp" (Turner et al., 1998), etc. Force feedback bandwidth for these devices is in the range of 10 - 50 Hz. These gloves have one or more force degrees of freedom (DOF) per finger with forces grounded in the palm or on the back of the hand. A virtual hand maps the user's hand to the Virtual Environment.

The design of force feedback devices for hand rehabilitation was presented in the previous chapter. We focus in this chapter on haptic modeling methods for force feedback gloves. Glove-based haptic interactions require more elaborate models due to the complex shape of the interface. In addition, efficient development of Haptic Virtual Environments needs advanced haptic programming tools, similar to those available in computer graphics. Next sections analyze the requirements of haptic simulations and propose new concepts necessary to build haptic programming interfaces.

4.1.1 Haptic Programming Interfaces

Most of current haptic simulations were developed with low level programming tools. This is due to difficulties in integrating haptics with currently available graphic APIs. Typically the graphics is developed with OpenGL and the haptics uses custom rendering algorithms. Developing Virtual Environments with low level API is very time consuming. Higher level graphics API for VE development are available. Few haptic programming interfaces are available, most of them binding together haptics and custom developed graphics tools.

The GHOST™ SDK is a programming toolkit developed by SensAble Technologies for the PHANToM™ haptic interface (Sensable Technologies, 1998). This toolkit implements automatic computation of the interaction forces between a haptic point (mapped to the user's index fingertip), and objects or effects within the virtual environment. It can also simulate object compliance and friction, as well as springs, impulses and vibrations. GHOST™ allows programmers to specify object geometry, properties, and global haptic effects, using a custom scene graph. The scene graph includes most of the VRML nodes plus some additional ones necessary for haptic rendering. Among these new nodes are:

- *dynamic nodes:* to create dynamic behaviors for virtual objects;
- *effect nodes:* to generate haptic spatial effects;
- *manipulator nodes:* allow manipulation of hapticly rendered objects;
- *haptic interface device node:* a special node to represent the haptic interface point.

The library contains functions to import VRML files, but all haptic properties have to be added using GHOST functions. The graphics tools provided by the library allow development of simple graphics environments. For more complex Virtual Environments, the developer has to integrate Ghost with a dedicated graphic toolkit. The complexity of integrating such an SDK with a graphic one is illustrated by a WorldToolKit-Phantom example application. In this example the application needs to maintain two separate scenegraphs, one for graphics and the other for haptic rendering. The developer has to manipulate numerous pointers and to maintain consistency between the two separate scenegraphs. Development is difficult and the simulation is inefficient.

A higher level haptics programming interface for the PHANToM is "Magma SDK" (Reachin Technologies, 2000). This development toolkit uses a scene graph manager based on VRML specifications and extensions and has separate *graphics rendering* (OpenGL-based) and *haptic rendering* modules. The haptic SDK offers advanced modeling tools such as surface bump mapping, haptic texturing algorithms, rigid body dynamics (mass inertia), etc. The graphics SDK offers advance features like interpolators, behaviors, and NURBS rendering. This toolkit gives developers more options but still binds together the graphics and haptics programming tools.

A voxel-based approach to 6-DOF haptic rendering was developed by McNeely et al. (1999) at Boeing Co. This original approach extracts a point shell model from the voxel map of the dynamic objects. This corresponds to one point for each surface voxel of that object. In this way both speed and accuracy are maintained as long as the voxel has the appropriate size, and only one dynamic object exists. The VPS Library built using this algorithm contains programming tools for collision/proximity detection and force modeling based on a spring-damper model. Haptic simulations are integrated with the proprietary FlyThru© visualization system.

A haptic toolkit for game developers is DirectInput from Microsoft (Microsoft Co., 1998). This is a component of the DirectX programming interface. It supports several haptic profiles (square, triangle, sawtooth, sine wave) and force feedback models using spring, damper, inertia, and friction. The toolkit functionality is very limited, targeting only force feedback joysticks and similar game devices (steering wheels, etc.). The easiest integration is with Direct3D graphic programming interface, also part of DirectX.

The software toolkits presented in this section bind together haptic with proprietary graphics development tools. A flexible graphics & haptics development platform needs to allow programmers to add haptic rendering to a currently running graphics simulation. A look at the haptic rendering pipeline will give us some insight on the design of haptic programming tools.

4.1.2 Haptic vs. Graphics Pipeline

The discipline studying the generation and rendering of haptic stimuli was coined "Computer Haptics" by Srinivasan and Basdogan (1997). From the beginning, its development was influenced by the more mature computer graphics research. There are many examples were haptic rendering algorithms were inspired by graphics rendering. But there are also significant differences related to the way the information is rendered and displayed. Therefore a graphics vs. haptic rendering comparison is relevant. This comparison will help us understand the best way to integrate haptic and graphics feedback in Virtual Reality simulations.

The basics of haptic rendering theory are presented in (Srinivasan and Basdogan, 1997). While the discussion there is in many cases limited to point-based haptic interactions, most of the concepts can be generalized to other types of interaction (hand based, etc.). Starting from this theory, we envisioned a haptic rendering pipeline similar to the graphics one. The five stages of the graphics pipeline are scene generation, scene graph traversal, transformation, rasterization and display (Cok et al., 1999). Similarly, the haptic rendering pipeline can be segmented in five basic stages: scene generation, collision detection, force rendering, tactile rendering and haptic display. These stages are further detailed in Fig. 4.1. Force rendering can be divided in three sub-stages: force calculation, force smoothing and force mapping. When touch feedback is superimposed over force rendering, a touch rendering stage is needed. Haptic texturing algorithms will generate the localized feedback displayed on the touch sensors. Here are the definitions of the haptic rendering stages:

- *Scene Generation:* performs the loading or generation of the haptic data. The data are organized in a suitable structure for haptic rendering. This work is done prior to entering the rendering loop.
- *Collision Detection*: check collisions between the virtual avatar of the haptic interface (*H-object*) and other virtual objects. Collision checking involves scene graph traversal, and is similar in this respect with the *Traversal* stage of the graphics pipeline. However scene traversal has different goals in graphics than in haptics. In the graphics pipeline, all connected nodes (or internal data structures) are passed to the rendering stage. In the haptic pipeline, only the object colliding with the haptic avatar should be rendered. Therefore only the colliding structure is passed down to the haptic rendering stage.
- *Force calculation*: calculate the interaction force between the *H-object* and the colliding virtual object; typically uses Hooke's law or spring-damper model.
- *Force Smoothing*: adjust the direction of force vector in order to avoid sharp transition between polygonal surfaces.
- *Force Mapping*: project the calculated force on the haptic display system.
- *Haptic Texturing*: perform rendering of touch feedback.

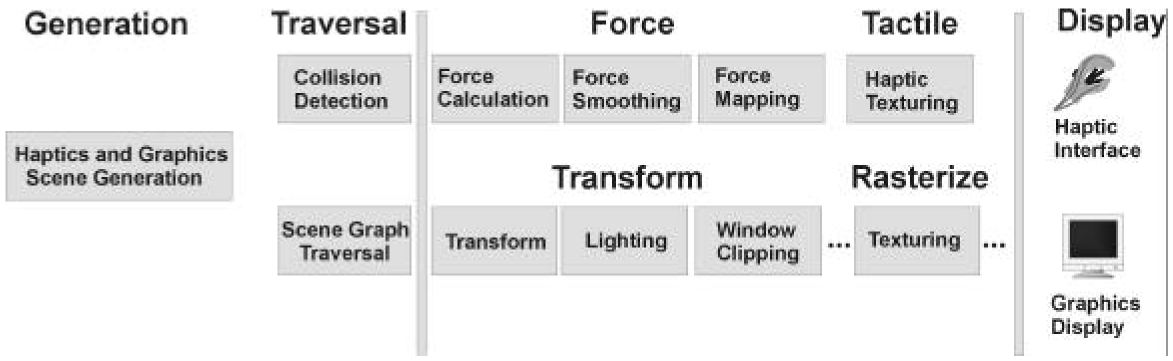


Figure 4.1: Haptic vs. Graphics pipelines

Some interesting analogies can be drawn between the operations performed by the two pipelines. For example, some *force smoothing* (haptic pipeline) algorithms were derived from Phong shading, a *lighting transform* algorithm (graphic pipeline) (Foley et al., 1990). Other analogies are related to haptic and graphic texturing. Analogous to the bump mapping technique used in computer graphics, a force perturbation method was proposed for haptic texturing.

Despite these analogies, the implementation of most of the stages is quite different. A major difference is in the speed of the graphics and haptic loops. Haptic rendering runs an order of magnitude faster and uses simplified rendering methods rather than physically-correct ones. Providing adequate rendering and mechanical bandwidth is currently the major challenge of haptic display systems. Haptic computation performance is limited by the number of objects in the scene graph. This limitation appears at the collision detection stage, which involves processing the whole scene graph. The rest of the haptic pipeline performs only local computations on the colliding object. In contrast, graphics pipeline performance scales down with the total number of polygons in the scene graph. Graphics performance can be improved by optimizing the rendering algorithms at each stage in the pipeline. Performance improvement in haptics rendering is currently dependent mostly on designing faster collision detection algorithms.

Another important aspect that can be observed from the above comparison is that the two pipelines only overlap in scene generation and traversal stages. Similarly to graphic APIs, haptic programming interfaces should provide development tools for data generation, traversal and manipulation. Benefits related to easy software integration will come from using the same architecture. Since the graphics research has standardized the scene graph architecture, the haptic software should use the same model. One of the solutions is to extend the scene graph concept with nodes carrying haptic information, as described in the next section.

4.1.3 Scene Graph Haptic Extension: The Haptic Node

A straightforward way to add haptic information to a scene graph is to extend it with a haptic node. This node should be attached to the group containing the transforms and geometry of the virtual object. The necessary fields of the haptic node are: *stiffness*, *viscosity*, *friction* and *haptic effect*. The general force model considered is:

$$F = k * x + b * x' + u \tag{4.1}$$

Three parameters – s*tiffness (k), viscosity (b) and friction force (u)* - are used to implement this spring-damper-friction force model. A simpler model is Hooke's law $F = k * x$, largely used by current haptic simulations. However this model will exhibit an unstable contact behavior for high stiffness objects. Spring and damping parameters require fine-tuning in order to eliminate the vibration caused by control instabilities. Friction is rarely used at the present time, as it requires additional gravity force calculations. Portable interfaces such as haptic gloves cannot simulate gravity and friction forces. *Friction* can be simulated with desk/floor grounded haptic displays like the PHANToM.

Haptic effect field is a one-byte code indicating the force feedback effect to be rendered on the interface. When this is zero, no effect is required and the force is calculated according to the above model. All other codes represent some predefined force profiles. Examples of haptic effects are: square, sine wave, constant force, spring effect, etc. Effects can be rendered directly on the controlling interface, thus reducing the communication bandwidth required between the controller and the host computer.

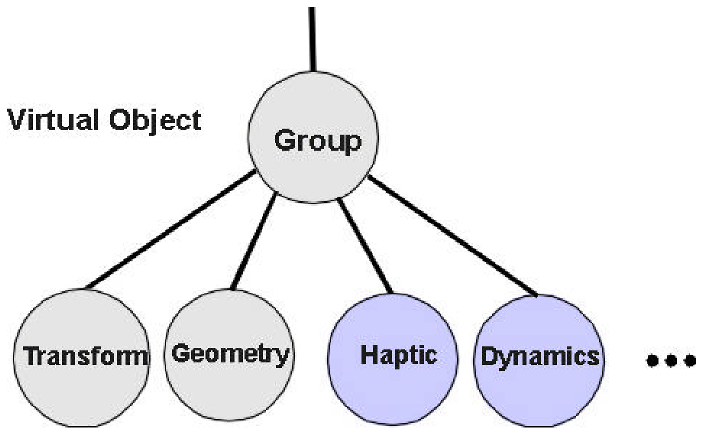


Figure 4.2: The Haptic Node

Using a haptic node is important for compatibility with VRML scene graph. However, this might not provide the best implementation of the haptic data structures. Depending on collision detection algorithm of choice, an optimized internal data structure can be used (see for example the haptic data structure proposed by Cai et al. (1999)). The simulation will then load the extended VRML (or compatible file formats) scene graph and convert input data to a custom haptic data structure - a hierarchical structure optimized for collision detection. This solution combines the advantage of optimized collision search with an intuitive haptic programming interface. Being based on VRML, such a programming interface will allow deployment of haptic simulations on the WWW.

4.2 Haptic Interaction Modeling

As mentioned above, previous work on haptic rendering concentrated mostly on point-based haptic interactions. This type of model is simple and efficient (can achieve high update rates), but has some limitations. For example it does not take into account the shape of the haptic instrument. A step upward in haptic tool modeling complexity is ray-based haptic rendering (Basdogan et al., 1997). This method, designed for a pen shape haptic instrument, approximates the end effector with a line. However, in many cases the haptic instrument has a more complex shape that needs a better representation. An example is a virtual hand for real-time interaction with the VE. Fingers of a virtual hand (which map the user's fingers) have a complex shape that cannot be reduced to simple point or line geometrical primitives.

To account for the haptic instrument shape, we propose a *haptic interface mesh (HIM)* – a set of points used for haptic rendering and deformation. The *HIM* is a simplified geometrical shape that captures the essential features of the haptic interface interactions. The method evolved from the popular haptic interface point algorithm. The algorithm, developed at MIT Touch Lab, is used for modeling PHANToM haptic interactions. The basics of the algorithm and some enhancements for the case of dynamic virtual environments are presented in the following two sections.

4.2.1 The Haptic Interface Point Method

The method uses a virtual representation of the haptic probe end point, called the "Haptic Interface Point" (HIP). HIP is not constrained and always follows the movements of the haptic device. In addition to the HIP, the algorithm defines a pair point called IHIP. IHIP follows HIP outside of virtual objects, but is constrained to not penetrate surfaces. Whenever HIP penetrates a virtual object, IHIP stays at the surface in a position closest to the HIP (see Fig. 4.3). The algorithm has two stages, defined by the position of the HIP relative to the virtual objects: a) outside of virtual object; b) inside of virtual object. In the first case, the HIP path is traced and checked for collisions with the virtual objects. A polygon-level collision detection algorithm is used to detect the polygon penetrated by the HIP path. This polygon is labeled *contacted primitive*. Once collision is detected, the algorithm enters the second stage. Now for each new HIP position an IHIP position (representing the closest point on the object surface) is calculated. This position is determined using a local search algorithm on the virtual object surface. The surface element corresponding to the final IHIP position becomes the new *contacted primitive*. Next HIP-IHIP vector is used to calculate the penetration distance and to check if collision is still on. When the dot product of this vector and the normal of the *contacted primitive* is negative the collision is off. The HIP is outside of the virtual object and we are in first stage again, searching for possible collisions. A detailed presentation of the algorithm can be found in (Ho et al., 1997).

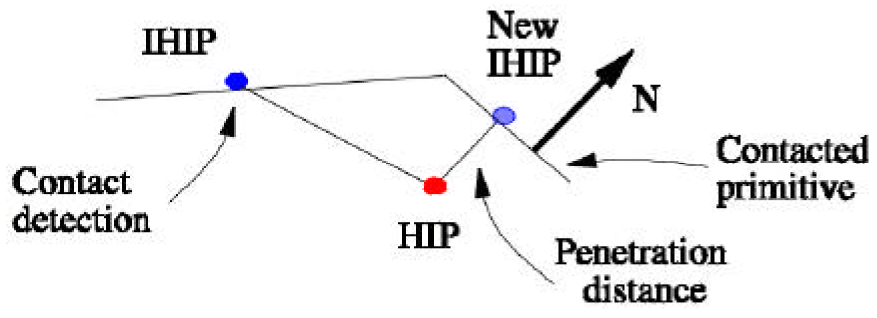


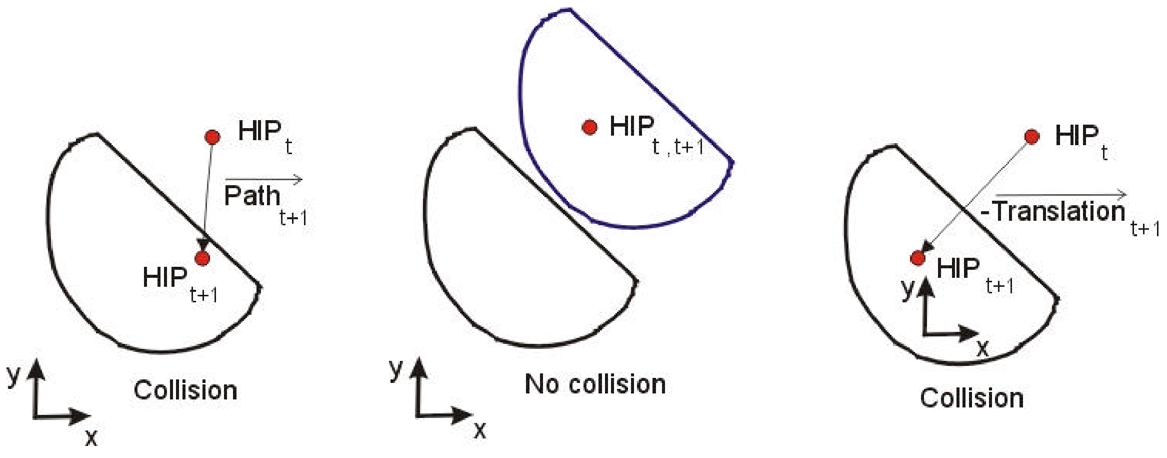
Figure 4.3: The HIP method

4.2.2 The Case of Dynamic VEs

The algorithms presented above handle successfully interactions with static objects but fail for dynamic ones. The reason can be easily seen in Fig. 4.4. In the first stage of the HIP algorithm the collision detection is calculated in a fixed reference frame. When *path* vector is tested against the polygonal surface of the fixed virtual object (Fig. 4.4-a), collision detection is reported correctly. The opposite case corresponds to a fixed user input (*path* is a null vector), and a dynamic virtual object. Here the polygonal surface of the dynamic object can “skip” intersection with the *path* if a fixed reference frame is used (Fig. 4.4-b). However, when collision detection is checked in the dynamic object reference frame, collision is reported correctly (Fig. 4.4-c). The easiest way to modify the previous algorithm is to compute the path vector as:

$$\overrightarrow{Path_{t+1}} = \overrightarrow{HIP_t\ HIP_{t+1}} - \overrightarrow{Translation_{t+1}} \qquad (4.2)$$

for collision detection purposes. No modification is needed for the second stage of the above algorithm.



a) b) c)

Figure 4.4: HIP collision detection: a) for static objects; b) for dynamic objects; c) the modified path vector.

4.2.3 The Haptic Interface Mesh

Instead of a single point, a collection of haptic interface points can be used for a more accurate description of the haptic instrument. We call this collection of points a “Haptic Interface Mesh” (HIM). The HIM is obtained by sub-sampling the haptic instrument geometry following rendering device constraints. This creates a mesh representation of the haptic instrument separate from the graphical one. Independent mesh representations of the haptic interface instrument help separate the haptic and graphics rendering loops. The level of detail of graphics and haptic models (mesh) can be varied independently according to simulation parameters (haptic interface resolution, real-time simulation constraints, etc.). Generally, the graphics model has a higher resolution due to better graphics display tools currently available.

The *HIM* representation was applied for modeling the haptic interactions of the RMII force feedback glove (Popescu et al., 1999-b). We assumed that the haptic interaction takes place at the fingertips of the virtual hand. Haptic meshes were therefore mapped to each fingertip, on the interior side. The central point of the *HIM* corresponds to the "center" of the fingertip. The point is obtained as a projection of the fingertip center of gravity along a normal pointing towards its interior side. The rest of the mesh points sit on the polygonal surface of the interior side of the fingertip, in an equally spaced grid. The selection of mesh points follows an adhoc procedure. The automatic generation of the *haptic interface mesh* starting from the graphic model of the haptic interface instrument requires further investigation.

Representing the fingertip with only one HIP is inadequate for force calculation purposes. For example, this representation will not account for the orientation of the virtual fingertip: for the same penetration distance but different fingertip orientations, the same force is displayed at the user's fingertip. The *haptic interface mesh* method will account for different fingertip orientation. The method is able to provide a better representation of finger haptic interactions provided that the surface of the interacting virtual object has an adequate level of detail.

Each point of the haptic mesh is an HIP and interacts with the virtual objects following a spring-damper model (see Fig. 4.5). As in the HIP method, each haptic point of the *haptic interface mesh* has a corresponding "ideal haptic interface point" (IHIP), or surface point, on the original (undeformed) surface. Force vectors are first calculated in each point of the haptic interface mesh - *HM_m* - with the formula:

$$d_m = u\left(< \vec{P}_{surface\,point\_m} - \vec{P}_{HM\_m}, \vec{N}_{surface\_m} > \right) \tag{4.3}$$

$$\vec{F}_{HM\_m} = k * d_m * \vec{N}_{surface\_m} \tag{4.4}$$

where k represents the object stiffness, d_m is the distance between the surface point and the haptic point along the normal defined at the surface point, and u is the unity step function (forces are applied only when the surface is being deformed).

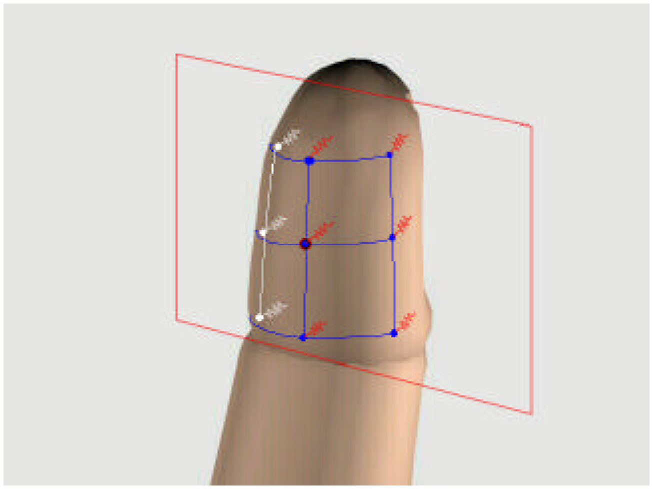
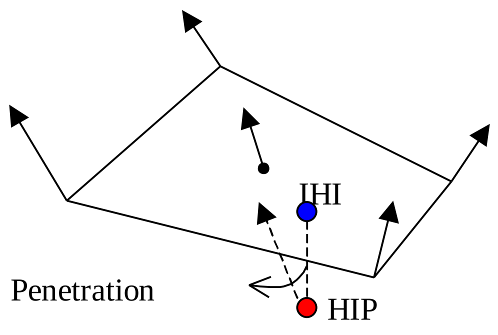


Figure 4.5: Haptic interface mesh mapped on a virtual fingertip

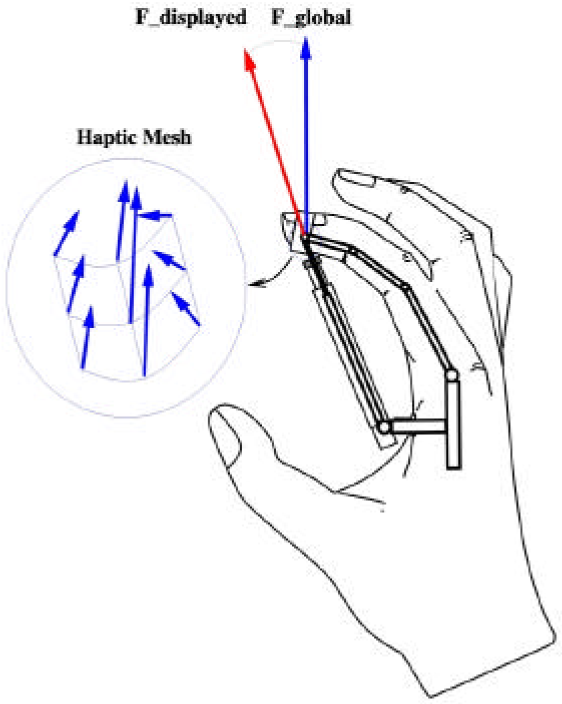
The next two steps after force calculations are force smoothing (shading) and force mapping. Force smoothing prevent large variation in the force from being displayed when interaction points are crossing the edges of polygonal surfaces. The algorithm is similar to Phong shading in graphics (Foley et al., 1990). To obtain the normal vector ($N_{surface\_m}$) a weighted average of the vertex normals closest to the point of contact is calculated. The weights are set based on the distance from the point of contact (the closest vertex receives the largest weight). The weighted average normal is then normalized. The normal vector obtained from this calculation is subsequently used to determine the direction of the force the surface exerts on the finger. This normal is then used as the projection vector for the calculation of the finger penetration distance. (see Fig. 4.6-a)

The force vectors computed using this method are then mapped to the force displayed to the user’s fingertip, as illustrated in Figure 4.6-b. In our case, we compute a global interaction force, which represents the magnitude of the force displayed to the fingertip:

$$\vec{F}_{displayed} = \left\| \sum_{m} \vec{F}_{HM\_m} \right\| \tag{4.5}$$



a)



b)

Figure 4.6: a) Force smoothing; b) Force mapping for the RM-II glove

4.3 Modeling Deformation

In realistic VR simulations object deformation accompanies the force feedback rendering. Our VR simulation assumes that the virtual finger is rigid, while the other objects in the

VE can be deformed. The deformation model implemented uses the displacement vector of the mesh points (see (Burdea, 1996) for a review of physical deformation methods). The method is simple, can be executed in real-time and fits well with the haptic rendering techniques previously described.

The model can be applied to both elastic and plastic deformations. The elastic deformation model uses a non-deformed reference object, while the plastic deformation model updates the reference object after each simulation frame. The elastic deformation is implemented in two steps: a global and a local deformation. The global deformation model uses a morphing technique. The position, normal and color of all object vertices are interpolated linearly between a normal state (corresponding to an opened hand) and a maximum deformation state (corresponding to fully closed hand). The interpolation parameter α is the normalized mean of finger joint angles:

$$a = \sum_{i=0}^{joints\,\#} q_i \Bigg/ \max\left(\sum_{i=0}^{joints\,\#} q_i \right) \tag{4.6}$$

The local deformation model is controlled by the fingertip haptic interface mesh. Contact distances are calculated between the points of the haptic interface mesh on the surface of the fingertip and the closest vertices of the intersecting object within a certain radius of influence (typically of the size of the largest dimension of the fingertip bounding box). Deformations are calculated as a function of the distance from the mesh points to the vertices of the deformed object. A second-degree polynomial is currently used to compute the magnitude of these deformations:

$$D_{m,i} = k_i * u\left(< \vec{P}_{vertex\_i} - \vec{P}_{HM\_m}, \vec{N}_{vertex\_i} >\right) \quad \text{where}$$
$$k_i = 1 - (d_{m,i} / radius\ of\ influence)^2, \qquad (4.7)$$
$$d_{m,i} = \left\| \vec{P}_{HM\_m} - \vec{P}_{vertex\_i} \right\|.$$

and u is the unity step function (deformations are calculated only for positive penetration distances). The deformations from different points of the *haptic interface mesh* are not summed up but a maximum value is calculated to obtain vertex displacement:

$$D_i = \max_m \left(D_{m,i}\right) \qquad (4.8)$$

Figure 4.7 presents a sequence of deformations while squeezing a virtual ball. The screen shoots were taken while the hand was moving in real time.

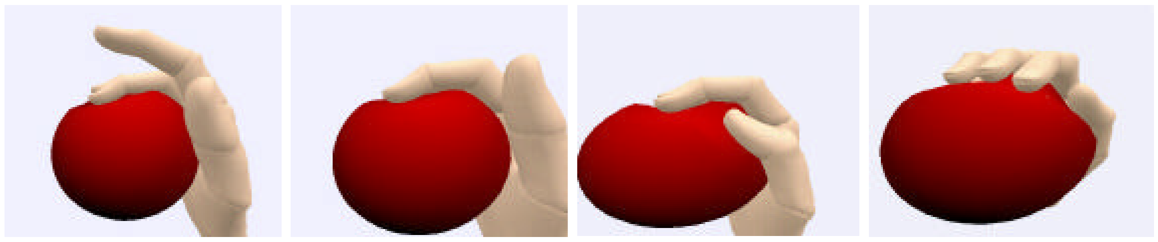


Figure 4.7: Rubber ball squeezing sequence.

4.4 The Rutgers Haptic Library

Efficient development of haptic virtual environments requires an advanced haptic programming interface, which gives the developer the ability to manipulate haptic objects without concern for haptic rendering details. Based on the previously described algorithms and rendering methods, we designed a haptic library for RMII virtual reality simulations (Popescu, 2000). The design of the library addresses the main requirements of a haptic programming interface:

- allows easy integration with the graphics API;
- hides haptic interface driver programming details;
- provides haptic modeling tool for contact detection, force calculation, smoothing and mapping, and haptic effects (sine wave, spring, etc.);
- allows further extensions.

The haptic modeling is based on the haptic interface mesh method presented in the previous section. The use of a unified scenegraph extended with a haptic node allows easy integration with the graphics API. The library sits on top of the haptic interface driver, presented in the next section.

4.4.1 The Haptic Interface Driver

The low-level component of the haptic rendering engine is the haptic interface driver. For the RMII system, the driver contains a component running on the electronic control interface and one on the host computer. The design and performance of the driver was presented in the previous chapter. As shown there, the driver provides methods for reading and writing data to the haptic control interface, setting reading/writing modes and calibration. In this section we are analyzing its functionality as part of the haptic rendering pipeline. The driver supports two rendering modes:

1. local haptic rendering – the forces are all calculated and displayed locally on the control interface, using parameters downloaded from the host computer; this mode is useful for implementing the haptic effects mentioned in the previous section.

2. remote rendering on the host computer – the forces are calculated on the host computer and transmitted to the haptic interface.

In the first model the host computer only commands the beginning and the end of haptic feedback loop. Haptic interaction parameters (spring constant, friction and dumping constants, position parameters) are transmitted by the host PC when the force feedback loop is activated. Forces are subsequently calculated locally on the haptic control interface based on glove sensor readings (finger position) and haptic effect model. In the second case the haptic rendering model resides entirely on the PC workstation. Here the host PC calculates and sends force targets to be displayed by the HCI to the user's fingers.

The first method is suited for grasp-release type of interactions as it assumes that the relative position of the hand and grasped object does not change. The second method is suited for haptic interaction with static objects. The limitation of this method is related to the communication bandwidth that can be achieved over a serial line. In our experiments the average force refresh rate to be displayed at the fingertip was around 100 Hz (vs. 500 Hz achieved by the local rendering loop).

4.4.2 Haptic Library Implementation

The Rutgers haptic library was implemented in C, as an extension to WTK library, a multi-platform VR programming tool (Sense8, 1997). Its modular architecture overlaps with the stages of haptic rendering pipeline, described in a previous section. The WTK scene graph representation was enhanced with a haptic node attached to the object group node. The haptic node contains stiffness, damping and haptic effect fields. Friction cannot be implemented with the current design of the RMII haptic glove. The Rutgers haptic library performs the following operations:

- traverse the scene graph and check for collision;
- calculate the contact detection between the fingertip (last segment of the finger model) and the virtual object;
- highlight the contacted polygon for each finger during the simulation;
- calculate the penetration distance for each fingertip;
- smooth the normal of the contacted polygon;
- calculate the force applied for each fingertip based on the penetration distance and polygon normal;
- map the forces calculated at the previous step on the RMII hand master.

The functions in the library are organized in several components. The first component manages the internal nodes and parameters of the haptic rendering algorithm, generates haptic points and lines, turns on and off their visibility, sets the haptic mode and the node haptic properties, etc. The second component implements collision detection and contact polygon selection. The third component does the force calculation. The last component contains functions used for managing the haptic simulation. Each of these components contains several methods accessible to the developer. Library functions are described in Appendix A.

The structure of the haptic simulation loop is presented in Fig. 4.8. All algorithms are executed in sequence for all four fingers provided with force feedback by the RMII interface. The loop is derived from the HIP algorithm, modified to use a haptic mesh at the force calculation stage. The haptic loop has two stages: collision and rendering.

Collision detection is used only for the central point of the haptic mesh (one point per finger). Collision detection is executed in several steps. First a bounding box collision is checked for the virtual fingertips. If this step indicates a possible collision, a polygon level collision detection is executed. The detection of a colliding polygon switches the loop in the rendering stage. Before the switch, the colliding object haptic data are loaded in the force calculation model. The second stage implements the haptic interface mesh method. A force vector is calculated for each point of the mesh and then smoothed. The resultant force of the finger mesh is sent to the haptic control interface and rendered on the fingertip. The penetration distance is checked each loop in order to determine if collision is still present. When the HIM center steps outside of the colliding object, the loop gets back to the collision stage.

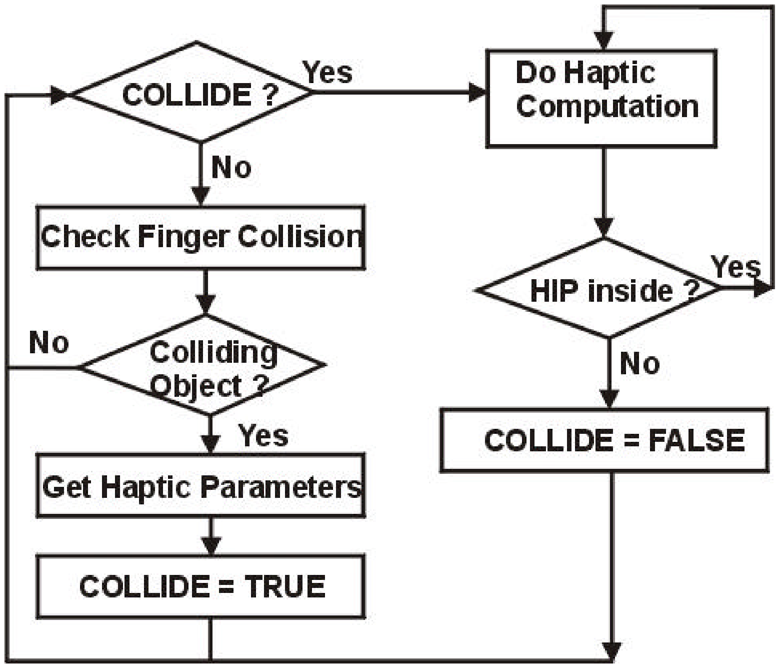


Figure 4.8: Haptic simulation loop

4.4.3 The Test Application

The performance of the Rutgers haptic library was tested on two PC platforms. The Virtual Environment of the test application contains about 2,500 polygons. The application implements the haptic rendering of two static objects: a hemisphere and a cube. Both objects have the vertex normals defined. The haptic properties of the two objects are described in the attached haptic scene graph nodes. The application displays the contact polygon in green. When not in contact, polygon color returns to blue. The polygons resulted from a local search are colored red. In the next frame following the search, these polygons are turned to their initial color (blue). In this way the user can see how the collision and polygon tracking mechanism work in the haptic library. Next figure shows the virtual hand interacting with the hemisphere.

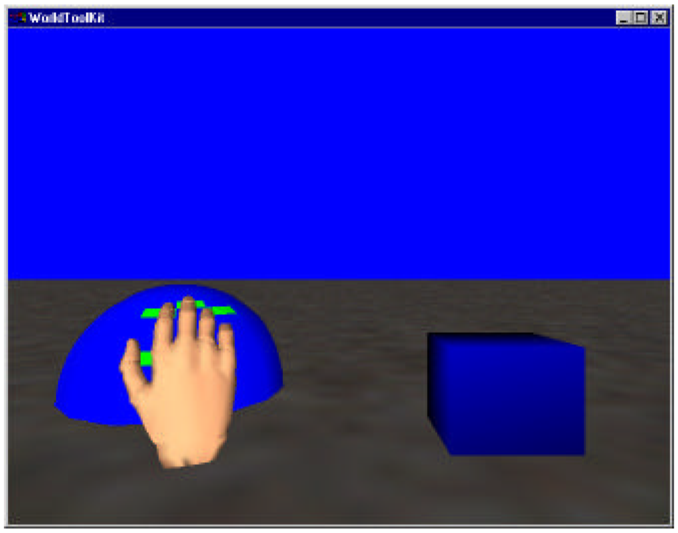


Figure 4.9: Haptic library test simulation

The haptic rendering functions are called in a separate thread, which runs several times faster than the graphic rendering loop. The force update rate depends on the characteristics of the machine running the simulation. The graphics and haptic performance is shown in table 4.1. As can be seen there, simulation performance is CPU bounded. Therefore a dual processor machine is the preferred configuration.

Platform	**Haptic Thread** (frames/second)	**RMII driver** (updates/second)	**Graphics** (frames/second)
Dual Pentium 450 MHz with AccelGalaxy	300	110	40
Pentium 300 MHz with FireGL 4000	50	58	21

Table 4.1: Haptic Rendering Performance

4.5 Conclusions

Modeling hand haptic interactions needs more elaborate methods than previous point-based interaction models. A new method for modeling virtual hand haptic interactions was proposed. The model uses a *haptic interface mesh* to calculate haptic interactions and object deformations. This model allows better quantification of local haptic interactions. The method is not dependent on the haptic interface hardware used. However, the mesh parameters need to be customized according to the geometry and rendering capabilities of the haptic device.

The rendering method was applied for modeling the haptic interactions of the RM-II interface. Haptic interface meshes were defined at each fingertip and used in the force calculation and virtual object deformation. A programming toolkit – the Rutgers haptic library – was developed for modeling hand-based force feedback. The toolkit was used to develop real-time simulations that involve elastic and plastic deformations and physical modeling.

Chapter 5

Design of a Rehabilitation Virtual Environment

The most notable application of Virtual Environments for rehabilitation is the treatment of psychological disorders. Several research projects and clinical studies have used Virtual Reality Therapy (VRT) for treatment of specific phobias (agoraphobia, acrophobia, claustrophobia, etc.) (North et al., 2000). These applications require "exposure therapy" simulations, and therefore focus on visual and auditory feedback. Other user-VE interface components (user input, force feedback) were generally overlooked since they were not crucial for the therapy. In contrast to VE treatment of psychological disorder, orthopedic rehabilitation requires the use of force feedback and VE interaction techniques. Our design focused on developing interactive Virtual Environments enhanced with force feedback. In the previous chapter we presented the details of haptic modeling for a virtual hand. The interaction techniques needed to control the VE rehabilitation simulations are presented in the following section.

We implemented natural mapping techniques for selection and manipulation in order to provide an intuitive user interface. A general-purpose design configures the Virtual Room with the visual elements needed to simulate rehabilitation routines and to display relevant therapy information. All these elements were used for implementing a library of VR exercises for hand and ankle rehabilitation.

5.1 User-VE Interaction

The focus of Virtual Environments for orthopedic rehabilitation is haptic manipulation of virtual objects. In a typical scenario the user (patient) selects the virtual object, controls it and releases it. Controlling the VE is important in order to engage the user in the simulation. Furthermore, a good control system allows efficient patient training for lost skills. The control of virtual objects should simulate natural interactions, to allow the transfer of skills from patient training to activities of daily living. The two interaction techniques implemented for our system are selection and manipulation.

Traditionally human input in VE is achieved through gesture-controlled input devices (joysticks, trackballs, etc.). Therefore many interaction techniques were designed for simple point-and-click devices. Our application uses a sensing glove, which is a more complex input device. The glove allows dexterous interactions, overcoming the limitations of simpler input devices. Natural mapping techniques can be used in this case for user-VE interaction. These techniques replicate the physical world, but are not powerful enough for many VE applications. Other interaction techniques considered for our interface were arm-extension, ray-casting, “go-go” and “homer” techniques. A detailed review of selection and manipulation techniques is presented in (Bowman, 1999). We implemented natural mapping techniques since these are intuitive and provide good user control over a limited workspace volume. In our application, the interaction takes place in a virtual room (described in the next section) which was scaled to cover the volume accessible to user hand wearing the RMII glove. The sensing part of the glove allows user fingers to control virtual finger motions. A Polhemus sensor attached to the palm allows the control of the hand position and orientation. Virtual hand mapping is shown in Figure 5.1.

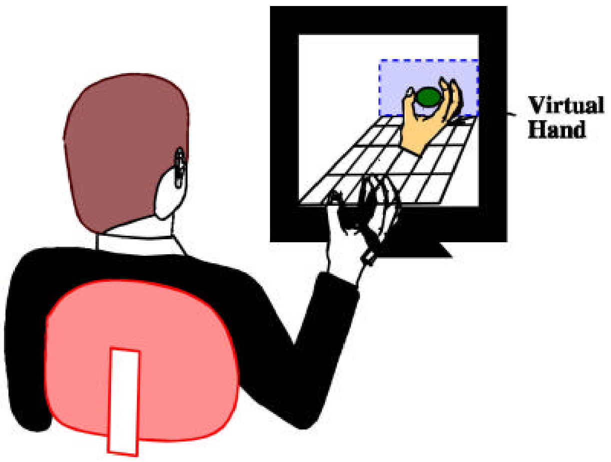


Figure 5.1: Virtual hand mapping

User-VE interaction is implemented on two components: collision detection and gesture recognition. Collision detection uses a bounding box collision between hand segments (palm, index proximal, etc.) and virtual objects. The bounding box fits tight on the cylindrical shape of the finger segments, providing a good approximation of hand - virtual object collision. Polygon level collision would be more accurate but also more expensive to compute, since the hand uses about 2000 polygons. Hand collisions trigger the gesture recognition module. This is a finite state machine where each state is a gesture. The gesture vocabulary implemented for user–VE interaction includes whole hand grasping, two finger grasping (lateral pinch), selecting (pointing), pushing, throwing and releasing. A description of user-VE interaction implemented for our VR simulation follows.

a. Grasping:

When the hand closes while being free of “attached” objects contact detection between hand segments and a virtual object triggers a grasping gesture. The “closed hand” condition is defined by finger positions. The simplest condition is shown below:

$$\bigcap_{finger \in \{thumb, index middle, ring\}} \left(\left(\sum_{i}^{I\_\max} q_{finger\_i} \right) \Big/ I\_\max \quad \geq \quad q_{finger\_threshold\_grasp} \right) \tag{5.1}$$

where *I_max* is the number of joints measured per finger: 2 for the thumb, 3 for the other fingers.

Angles are measured from the zero position corresponding to a flat (open) hand. When all average (per finger) joint angles are larger than a preset threshold and contact is detected, grasping is enabled. Similarly, two-finger grasping (thumb-index grasping) is defined by the above condition applied only to thumb and index joints. During grasping, force feedback and deformation engines calculate virtual object deformation and the forces to be applied to the user’s fingertips. The virtual object is “attached” to the hand until a release gesture is detected.

Threshold based grasping is too rigid and may be difficult to execute by the user in some cases. For example grasping moving objects is a difficult task when the above closed hand condition is used. Sometimes the user cannot estimate the speed and the position of the virtual object his is trying to grasp. Therefore the closed hand and contact detection condition may not occur simultaneously, preventing the user from grasping the object. A prediction based grasping mechanism would improve this situation. The idea here is to capture user intention to grasp an object rather than require him/her to learn and execute a preprogrammed grasping gesture.

Grasping is the final result of a user motion sequence which starts from the moment the grasping decision is made. Analysis of user motion prior to grasping should reveal his intention. Therefore hand position and finger motions are buffered over a sequence of *K* frames. At the time of collision, the output of the prediction filter is fed into the threshold based decision module:

$$\bigcap_{finger \in \{thumb, index middle, ring\}} \left(\sum_{k}^{K} a_k x_{finger\_k} \geq q_{finger\_threshold} \right), \quad \text{where K is filter size}$$
$$x_{finger\_k} = \sum_{i}^{I\_\max} q_{finger\_i}(t_k), \quad \text{where } \mathrm{t_k} \text{ indicate the k} - \text{th past frame} \tag{5.2}$$

The parameters and size of the filter were determined experimentally.

b. Release:

The release gesture is triggered when the user's hand opens while holding a virtual object. The open hand condition is shown below:

$$\bigcap_{finger \in \{thumb, index middle, ring\}} \left(\left(\sum_{i}^{I\_\max} q_{finger\_i} \right) \Big/ I\_\max \leq q_{finger\_threshold\_release} \right) \tag{5.3}$$

The hand is considered opened when all average joint angles are less than some predefined threshold. When that happens, the virtual object is detached from the hand and it resumes its behavior. A particular case is throwing, where the motion of the hand influences the object's behavior after the release. The virtual object position and speed at the time of the release (before been detached from the virtual hand) are the initial conditions for its flight dynamics.

c. Hitting:

Hitting is defined as the intersection of the virtual hand with a dynamic virtual object. The hit object bounces off the virtual hand (considered a plane aligned with the palm). A short force feedback pulse is applied to the fingers to signal the collision with a virtual object. The condition that triggers hitting is open hand collision with the object.

d. Pushing:

In contrast with hitting which involves a very short (one frame) interaction between the hand and the virtual object, pushing can span over a larger period of time. The virtual object is not attached to the hand, as in grasping, but it moves freely following physical laws. As long as the virtual hand is in collision with the virtual object, forces are applied to the user's hand. The virtual object is accelerated in the opposite direction of force vector proportional to the overall interaction force. When the contact between hand and virtual object is lost, the object decelerates due to the friction until it stops.

e. Selection:

The "select" gesture is executed with the index finger touching a virtual object. Selection is triggered by collision detection of index distal link with the virtual object surface. No hand configuration condition is imposed. This gesture typically used at the beginning of each exercise to interactively set exercise parameters such as the rehabilitation routine level of difficulty, object stiffness, etc.

A VR simulation was subsequently developed to test the hand gesture and haptic feedback modalities (Figure 5.2). The hand force feedback simulation was developed using the RMII haptic library presented in the previous chapter. The user can hapticly interact with two virtual objects. Each object has its own haptic parameters and behavior. On collision with the hand, the behaviors are stopped and the force calculation module loads stiffness, damping and friction parameters. The user can choose between squeezing and throwing the ball, or pushing the cube. Pushing the cube is based on collision detection and force calculation for each finger. The interaction force is the sum of all forces applied at the fingertips. The cube is restricted to move along its rail, until it stops due to the friction. When hitting the wall, the cube bounces back and loses energy. Acceleration is calculated as the ratio of interaction force along the rail over the mass of the cube.

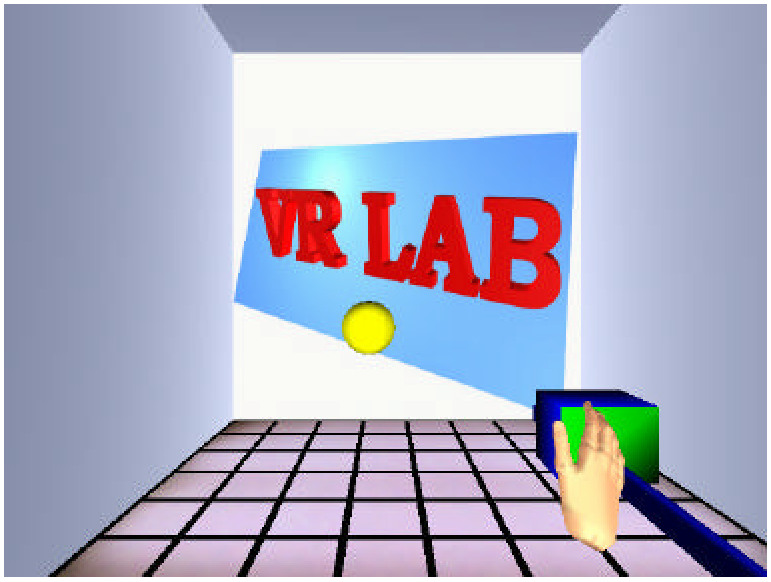


Figure 5.2: Pushing-throwing exercise

The test application showed that the hand interactions we implemented empower the user with intuitive VE controls. Force feedback contributed decisively to user engagement in the simulation. The application was demonstrated at VR 2000 Conference exhibit booth. Some or all of the hand gestures were implemented in the rehabilitation exercise library presented next. The testing of the library during Rutgers-Stanford pilot trials involved repeated fine-tuning of the interaction module in order to create easy to use interaction metaphors.

5.2 The Virtual Rehabilitation Room (VR^2)

The VR^2 is a Virtual Environment configured to simulate rehabilitation routines and to display to the relevant information related to the therapy. Some elements of VR^2 can be configured for a particular exercise while others can be common to a group of exercises. As an example, we will present the VR^2 for orthopedic rehabilitation exercises. In this case, the rehabilitation routines were broadly classified in two categories: physical therapy (PT) and functional therapy (FT). In both cases, the walls of the virtual room are used to display feedback information to the user and to support remote consultation. The essential information for physical therapy exercises is the force information. Therefore the virtual room will include a graphic display for finger forces and one for effort level per finger during the rehabilitation routine. Exercise remaining time is displayed in the upper right corner. The effort level and time display have a motivational effect, pushing the user to perform better against performance standards set by the therapist. The force display reinforces the haptic feedback with visual information, keeping the user focused on the rehabilitation exercise. Figure 5.3 shows a Virtual Rehabilitation Room configured for the DigiKey exercise.

The VR^2 for functional rehabilitation exercises has a different configuration. Here the force is used as an additional signaling channel to reinforce the visual information during grasping and object manipulation. Improving the hand eye coordination is the goal of currently implemented exercises. Therefore the VR^2 will display the number of errors made in addition to the remaining exercise time.

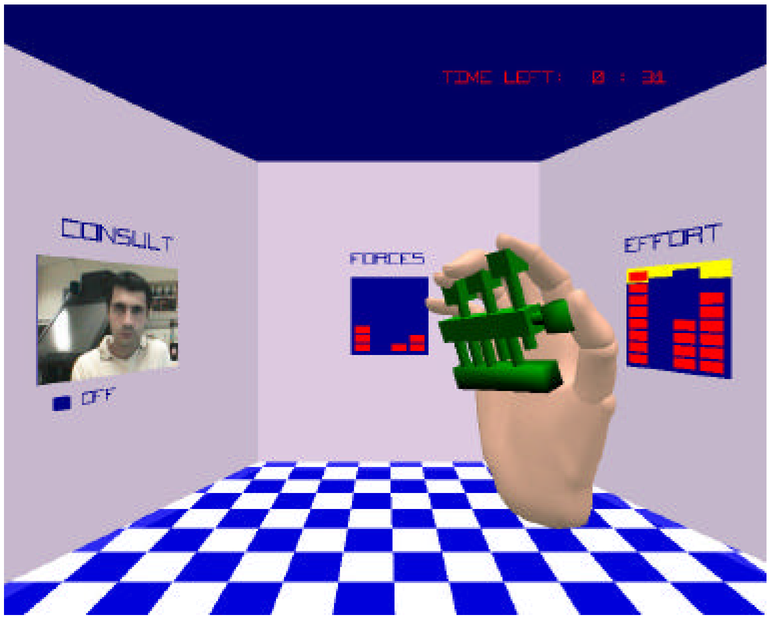


Figure 5.3: Virtual Rehabilitation Room VR^2 (Patient site)

Another component of VR^2 is the videoconferencing window. Videoconferencing was not integrated in the Virtual Environment. CuSeeMe videoconferencing software (WhitePine Software, 1997) is installed at the clinic and patient sites and runs as a separate program. The virtual environment only allows a patient to open a video channel for consultation with the therapist. This is the preferred solution for a monoscopic display. For a stereoscopic display, the videoconferencing software will need to be integrated in the VE and displayed as a texture since the stereo mode will prevent proper display of a separate application window.

5.3 The Virtual Reality Exercise Library

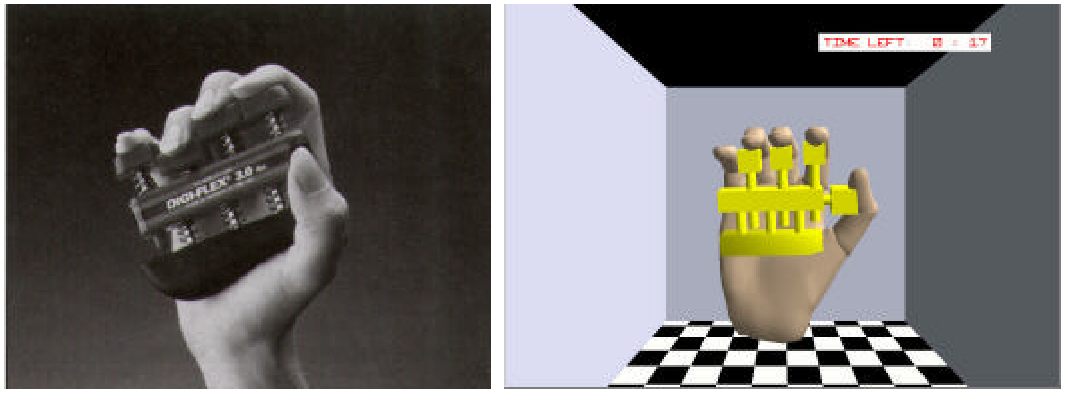
The rehabilitation exercises were developed using the commercial WorldToolKit graphics library (Sense8 Co., 1997), with a simple Virtual Environment in order to keep the patient focused on the rehabilitation procedure. All exercises contain a high-resolution virtual hand from Viewpoint DataLabs (Viewpoint, 1999) and several objects (DigiKey, Peg board, rubber ball, power putty) created with AutoCAD (Autodesk, 1994), or WTK Modeler.

Several Virtual Reality exercises were developed for both physical and functional therapy. Physical therapy exercises use force feedback to improve the patient's motor skills (exercise muscles and joints). Functional therapy is done to regain lost skills (such as those needed in activities of daily living or job related skills). Functional rehabilitation exercises have therefore much greater diversity and their output depends on each exercise design. The essential feature of these exercises is the patient's interactivity with the VE. Each therapy exercise has several levels of difficulty corresponding to the maximum force that can be applied, the time allowed, and several other parameters.

5.3.1 Hand Physical Therapy Exercises

The first PT exercise implements a virtual version of the DigiKey (North Coast Medical Inc., 1994), which is an individual finger exerciser, illustrated in Figure 5.4-a. The model was modified to include the thumb instead of the pinky due to the RM-II kinematics configuration (Figure 5.4-b (Popescu et al., 2000-b)). The DigiKey maximum force levels were color coded to match the commercially available set. After grasping the selected

DigiKey, contact detection is checked between fingers and the corresponding cylinder ends. While in contact, the virtual cylinders are driven by the patient's finger movements. Forces proportional to the displacement of the DigiKey cylinders are fed back to the patient and stored transparently and simultaneously in the database.

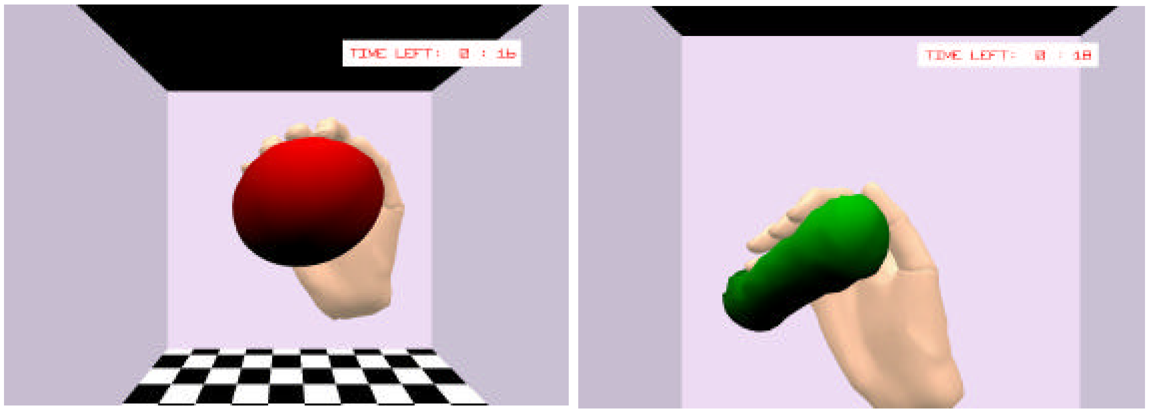


a) b)

Figure 5.4: a) DigiKey model; b) Virtual DigiKey.

The second PT exercise models a rubber ball squeezing routine, as illustrated in Figure 5.5-a. The ball stiffness is color-coded and can be selected by the patient at the beginning of the exercise. Ball dynamics simulate gravity and Newtonian laws. Once it is grasped, the ball deforms in contact with the virtual hand while force feedback is displayed to the patient and recorded in the database. The exercise terminates when either the patient presses an exit key, or the allowed time was exhausted.

The third PT exercise is a molding of virtual “power putty,” as illustrates in Figure 5.5-b. The patient selects between an ellipsoid or sphere unmolded putty shapes, each with three selectable hardness levels. Plastic deformation and haptic rendering models were used for this simulation. The ellipsoidal premolded putty is used for full grip, fingers only, thumb press only or wrist rotation exercises. The spherical premolded putty is used for finger pinch, where the putty is squeezed between the thumb and fingers. A “reshape” button allows the patient to reset the putty to its premolded shape before repeating the exercise.



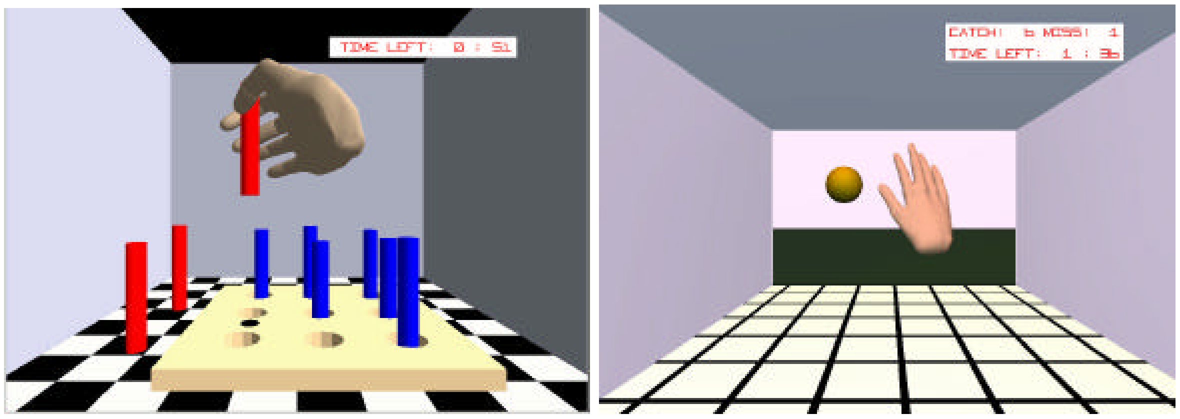
a) b)

Figure 5.5: Virtual PT exercises: a) rubber ball squeezing; b) power putty molding.

5.3.2 Functional Rehabilitation Exercises

The first functional rehabilitation exercise is a pegboard insertion task, illustrated in Figure 5.6-a. The simulation uses a virtual pegboard with nine holes and corresponding number of pegs. The exercise has three levels of difficulty: “Novice,” “Intermediate” and “Expert,” each with a different clearance between the peg and hole (smallest for the “Expert” level). The therapist sets the amount of time allowed to complete the exercise. Visual and auditory cues increase the simulation realism and help the patient overcome visual distortions. Pegs are grasped with a lateral pinch gesture and change color when in a correct insertion position. Exercise results are stored in the form of number of holes filled, time spent to perform the exercise, and number of errors made (missed hole or an attempt to put two pegs in one hole).

The second functional rehabilitation exercise is the Ball Game shown in Figure 5.6-b. The patient has to throw the ball so that it hits the target wall above a marked area (black). When the ball bounces back the patient has to catch it after at most one bounce off the floor. The ball speed parameter ("fast" or "slow" ball) is selected at the beginning of the exercise. Any correct catch increases the patient "catch" counter, while a miss will increase the "miss" counter. The ball deforms when caught by the patient and loses energy while bouncing. This exercise is useful to train feed-forward ballistic type movements and hand-eye coordination. Throwing and catching movements help improve accuracy and speed control.



a) b)

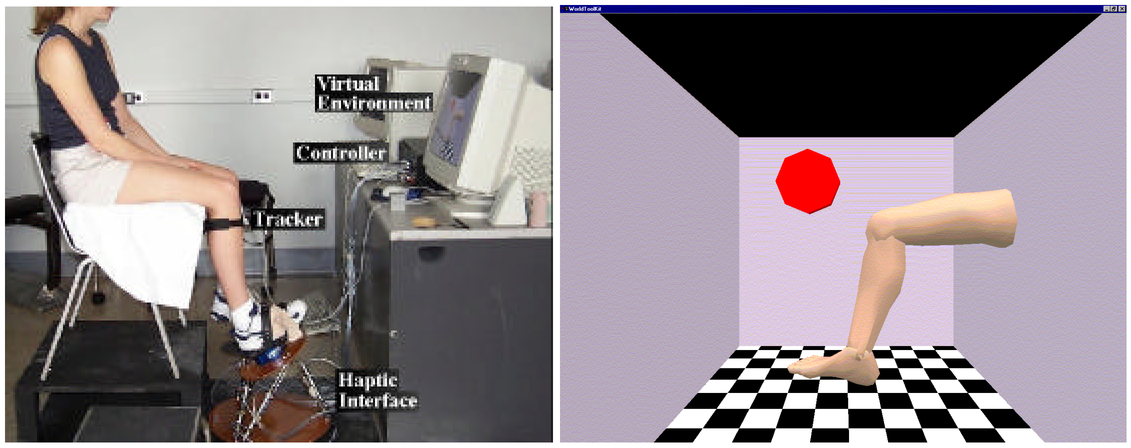
Figure 5.6: Functional rehabilitation exercises: a) Peg board; b) Ball Game.

5.3.3 Ankle Exercises

These exercises were developed for the “Rutgers Ankle” rehabilitation interface (Girone et al., 1999). The “Rutgers Ankle” is a Stewart platform that supplies forces to the patient’s foot during the rehabilitation exercises. The Stewart platform design allows the control of forces and torques in six DOFs and movement throughout the ankle’s full range of motion (ROM). The haptic device is designed to be inherently safe, as the patient’s shin is free to move. This ensures that the device cannot push the ankle beyond its normal ROM. A detailed description of the “Rutgers Ankle” can be found in (Girone et al., 2001).

The ankle exercises use the same VR^2 configured with new elements. The simulation displays now a virtual leg controlled by the user. User interaction is limited, early experiments focusing on visual, audio and force feedback. User input consists of a 6DOF platform and a magnetic tracker. The 6DOF of the ankle device are mapped into direct control of the virtual foot. Shin movement is controlled with an attached 6 DOF Polhemus sensor (see Figure 5.7-a). The rest of the body is fixed as the patient is sitting on a chair. The simulation can be configured for left or right foot and calibrated for different leg positions.

The ankle VR exercises focus on improving ROM, strength, coordination, and balance as well improving lower extremity function. Flexibility exercises improve the patient’s ROM by performing repetitive movements near their ankle limit of motion. At this position there are little or no opposing forces. While patients exercise, their foot position, orientation, and output forces become the inputs and outputs to/from the VE. The patients are required to execute planarflexion, dorsiflexion or eversion inversion movements. While performing the exercises, the display shows a moving foot controlled by the user (Figure 5.7-b). Audio feedback is used to signal the user that he is exercising in the required motion range.



a) b)

Figure 5.7. a) The “Rutgers Ankle” prototype; b) the VR exercise (Girone et al., 2000).

Strength exercises are similar to conventional weight-training exercises. By rotating their ankles, they move the virtual leg and lift a virtual weight. As patients move their feet, the “Rutgers Ankle” device applies programmed resistive forces. Scores are kept to motivate patients to apply more effort and exercise more frequently. In striving to break their records patients continually perform rehabilitative motions. During the exercise the host PC records exercise frequency, position, orientation, and force information. This data are later transferred into patient database.

Recently more interactive ankle VE simulations were developed by Deutsch et al. (2000). The visual feedback displays engaging game-like scenarios. The “airplane flight” simulation consists of a patient piloting a virtual airplane with the foot. The goal is to control the airplane so that it flies through a series of virtual loops placed on a programmed trajectory. Engaging simulations increase the motivation of the patient, which led to faster recovery.

5.4 Conclusions

The library of virtual reality exercises for rehabilitation has a unified design based on a virtual rehabilitation room model and natural interaction techniques. Haptic feedback is the key element of the simulations, providing patient muscle training and increasing VE control. Currently patient re-training of lost skills is implemented using selection and manipulation techniques. Navigation techniques (walking, flying, portal navigation) should be considered in future exercises in order to create more engaging simulations. This will allow the user to explore larger VEs and motivate its quest for performance improvement.

Chapter 6

A Client-Server Architecture for VR-Based Telerehabilitation

The previous chapters described the components necessary to build a VR-based telerehabilitation system. The home rehabilitation platform was shown in Chapter 3. The design of the haptic Virtual Environment was outlined in Chapters 4 and 5. This chapter presents the prototype of a "store and forward" system for home rehabilitation. The system uses a home PC-based rehabilitation station and a server allowing remote data collection and analysis, and modification of patient routines from the clinic. The VR-based rehabilitation data are collected at the patient's home during rehabilitation routines and is transferred to the clinic to be analyzed by the therapist. The system was tested as part of the NSF-sponsored Rutgers-Stanford telerehabilitation project (WebTelerehab, 2000). The project connects a single client site (at Stanford Medical School) with a server site (Rutgers VRLAB) (see Figure 6.1). In this pilot study Stanford Medical School plays the role of a home-based telerehabilitation site, while the Rutgers VRLAB represents a central clinical site. The project used in the beginning a T-1 (1.5 Mbps) connection which was subsequently upgraded to an OC-3 (155 Mbps). The pilot study started in October 1998 and is ongoing. The following section describes the telerehabilitation system architecture.

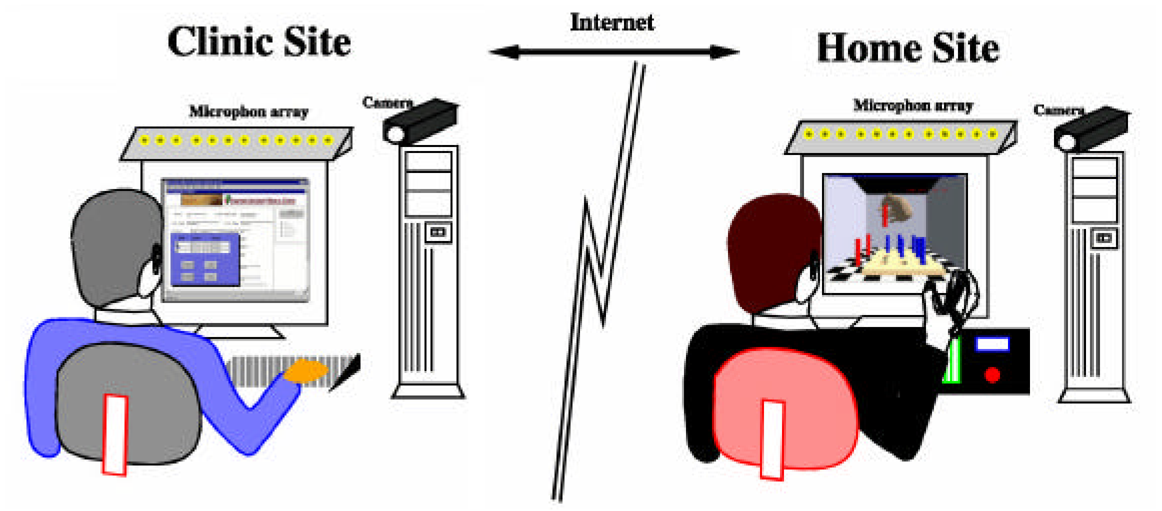


Figure 6.1: Clinic and Patient sites exchanging data over the Internet

6.1 System Architecture

This architecture implements a "store and forward" application where data collected from the patient are transferred at a later time to the clinic database (Figure 6.2). The Client (patient home) runs VR simulations and collects real-time patient data. These data are then forwarded to the server. The Server (clinic site) stores patient medical records and runs data processing software. After processing and formatting, the patient data are displayed to the rehabilitation therapist. The therapist evaluates patient progress and makes decisions regarding the evolution of the rehabilitation process. He is provided with online and off-line tools. Using the video conferencing system he can communicate in real-time with the patient. Off-line, the therapist can remotely set the VR simulation parameters based on progress. On the client side, the patient can access its database record and request video-consultation with the therapist when needed.

The application was designed with the T-1 bandwidth in mind. Upgrading the connection to OC-3 improved the videoconferencing quality and allowed patient data files to be transferred faster to the server. The “store-and-forward” system can also be used over a low bandwidth connection (modem). System performance over several network connection types will be evaluated in the next section.

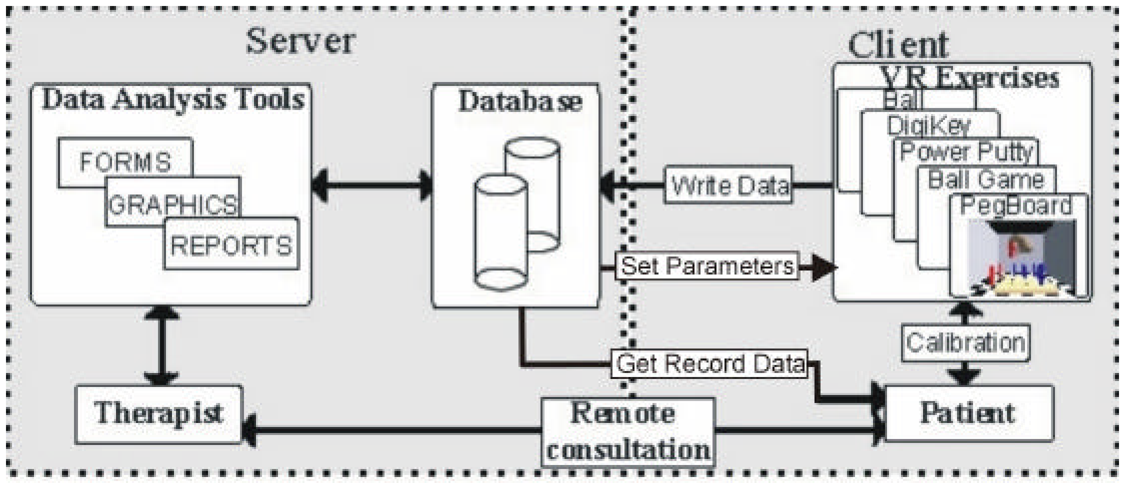


Figure 6.2: Client-Server Architecture for Telerehabilitation (Popescu et al., 1999-a)

6.1.1 The Clinical Database

The patient record is stored at the server site (clinic) in an Oracle database (Popescu et al., 1999-a), as illustrated in Figure 6.2. Patient data are organized in several tables: patient table (contains patient personal data), index table (contains exercise index, type and date), and exercise tables. The database schema is shown in Figure 6.3. Three types of exercise tables were built so far. The force table stores finger forces along with the *time* and an index identifying exercise, patient and date. One row of this table contains the finger forces sampled at the *time* instance during the VR rehabilitation exercise. The functional rehabilitation table stores in each row all rehabilitation measurements collected from the patient at the end of a rehabilitation session. The ankle table stores three joint angles and six forces and torques at the patient's foot.

PATIENT

Patient_ID	Last_Name	First_Name	Address	City	State	Zip	Area	Phone

…

Gender	Injured	Dominance	First_Ex_Date	Physician	Therapist	Comments

EXERCISE

Exercise_ID	DATE	Exercise_Type	Patient_ID

FUNCTION_R

Exercise_ID	Parameter_1	Parameter_2	Parameter_3	Parameter_4

FORCE

Exercise_ID	TIME	Force_1	Force_2	Force_3	Force_4

ANKLE

Exercise_ID	TIME	Position_1	Position_2	Position_3

…

Force_1	Force_2	Force_3	Torque_1	Torque_2	Torque_3

TARGET

Patient_ID	Exercise_Type	Date	Target_1	Target_2	Target_3	Target_4

Figure 6.3: Database schema: PATIENT, EXERCISE, FUNCTION_R, FORCE, ANKLE and TARGET tables.

The TIME column stores the elapsed time (in seconds) from the beginning of the current exercise. The DATE column contains the date when the exercise was performed. The underlined field in the database schema indicates the primary key. Other dependency and data integrity constraints were applied on the columns of the database. Several nested queries allow access to exercise data. The most frequent type of query retrieves the exercise data for a specific patient exercise type and session date. The queries are embedded in database reports described in a follow up section.

6.1.2 Store-and-Forward Data Collection

Patient data are collected during VR exercises as mentioned previously. The data are stored in a local file and transferred into the database at the completion of the exercise. The amount of data collected from the exercise depends on its type and duration. For physical therapy exercises we are recording sampled forces of patient interactions. In order to find an appropriate sampling rate, we started with the Nyquist criteria. Human muscle bandwidth is in the range of tens of Hz. Since our pneumatic based mechanical device limits the interaction bandwidth to about 10 Hz, the Nyquist sampling rate of forces will be 20 Hz. From the recorded data we found out that much less data is actually needed for the physical therapy exercises. Hand impaired patients have a smaller finger motion bandwidth. Additionally, physical therapy exercises assume slow motion while controlling the forces exerted by the haptic interface. We finally choose to sample the forces five times per second.

The data storage format is specific to each exercise. The data format of physical therapy exercises (such as DigiKey) is <time> <finger force>…<finger force>. The first field is the time the force sample was obtained. The other four <force> fields (in case of RMII exercises) correspond to thumb, index, middle and ring forces. For a one-minute exercise, sampling the four finger forces five times/sec results in about 10KB of data.

Functional rehabilitation hand exercises record only a few parameters: exercise difficulty, number of errors, number of correct actions, etc. These exercises produce very small files, typically of only tens of bytes.

Ankle exercise store more information per session. In addition to force data (three forces and three torque), it stores roll, pitch and yaw angles for each record. The sampling rate of ankle data is the same as for hand exercises: five samples per second. For a one-minute of exercise this produce about 30KB of data.

At the end of the rehabilitation exercises the data contained in the temporary files are automatically forwarded to the clinical database. The database update module was written using *ProC,* an Oracle programming interface for development of database applications. The database update application uses a TCP/IP connection. The transferred data file contains exercise type, patient ID, execution time and exercise raw data. The data transfer follows this sequence:

1. execute a login in the database;
2. insert a new exercise row in the exercise table; the row contains the patient ID number and exercise type from the transferred file;
3. transfer data sequentially and insert records in the database; the records contain exercise ID obtained at the previous step and the patient data from the transferred file.

A timer started at the beginning of transfer measures the total transfer time. This parameter, along with database insert failures is written in a log file. The log file can be later analyzed in order to assess the performance of the database update application, and for quality of service measures.

6.1.3 Patient - Therapist Interaction Model

The off-line patient-therapist interaction is implemented through a feedback loop, which allows remote control of VR routines at the patient site. After looking at the patient graphs, the therapist can judge whether the routine was performed in a satisfactory fashion or not, and evaluate the patient’s progress. He will then feedback control information into the system by modifying the target levels and level of difficulty parameters for the VR exercises. Next time the exercise is executed, the simulation runs with the new parameters. This off-line control mechanism enhances the quality of the telerehabilitation service offered at home, making the patient evolution more stable.

The real-time interaction between patient and therapist is supported using the videoconferencing tools described in the previous chapters. Videoconferencing hardware and software is installed at both clinic and patient sites. The graphical interface allows both the therapist and the patient to open a video channel for consultation. The patient can “call” the therapist on a need basis in order to ask questions related to system functionality or to the rehabilitation process. The therapist can discuss with the patient results and progress during the rehabilitation, and give him instructions related to the therapy exercises to be executed. Videoconferencing was intended as a support system, and it is not used as a therapy tool. However, it has an important complementary function to a store and forward system, enhancing patient-therapist communication and increasing patient control of the rehabilitation process. Other types of rehabilitation (neuropsychological therapy for example) will extensively use this patient-therapist communication modality.

6.2 Clinical Database Tools

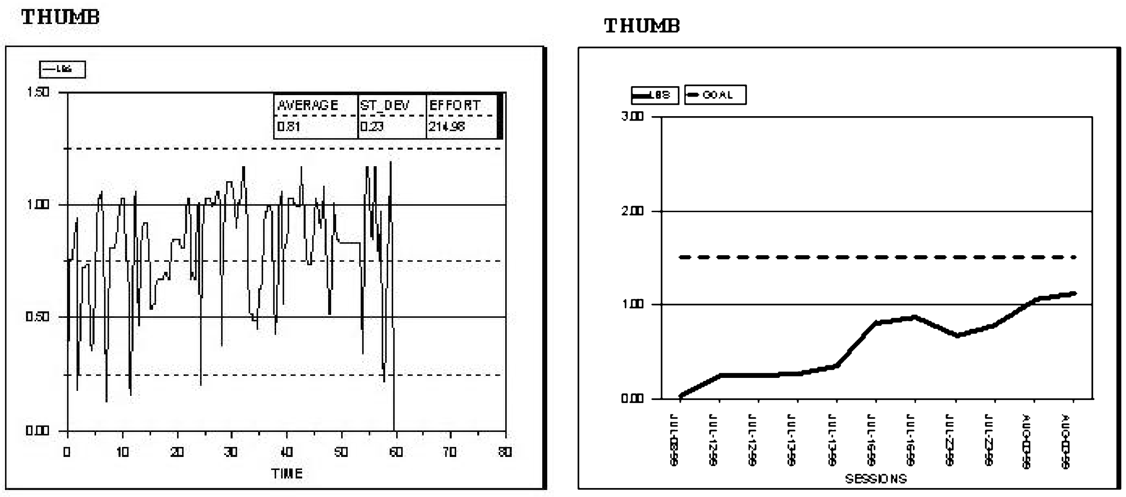
In addition to the database server, the clinical site includes a client developed for the therapist’s use. No VR component is implemented at the clinic site as all rehabilitation takes place at home. The client retrieves patient’s records and formats the data before displaying it as reports and graphs. Data processing tools calculate meaningful parameters, which help the therapist analyze the patient record. The therapist controls the formatting and display of the data through a graphical interface. The Graphical User Interface (GUI) was designed using Oracle Forms, Reports, and Graphics (Oracle, 1995).

6.2.1 Tools for Patient Data Analysis

As described in the previous section, the patient data stored in the database contains forces, 3D positions, and other patient performance parameters collected during the rehabilitation exercises. We call this “raw data”, since it contains the information collected directly from the exercise, without any processing. These data can be displayed as a time history of measurements collected during the exercise. This representation is useful to indicate patient’s focus on the exercise, resting periods, and frequency of motions.

Simply displaying the “raw data” is however of little use for analyzing patient performance and rehabilitation evolution. The data need to be processed further in order to extract meaningful information for patient remote assessment. We build these tools using PL/SQL procedures embedded in the graphics and report tools.

The first category of parameters extracted from the hand rehabilitation raw data contain the average, standard deviation and mechanical effort (force integral). These parameters are calculated for each exercise session and displayed along with the raw data (Figure 6.4-a). They are used to process the force data recorded during rehabilitation exercises. The “average” parameter indicates the average force the patient was able to exercise during the rehabilitation exercise. The standard deviation is a measure of force range displayed to patient’s fingers. The “effort” parameter is calculated as the integral of the force data. This parameter indicates the total effort exercised by the patient’s finger (thumb in the case of Figure 6.4-a) during the rehabilitation session. The three parameters described here offer an intuitive evaluation of patient performance during the rehabilitation session.



a) b)

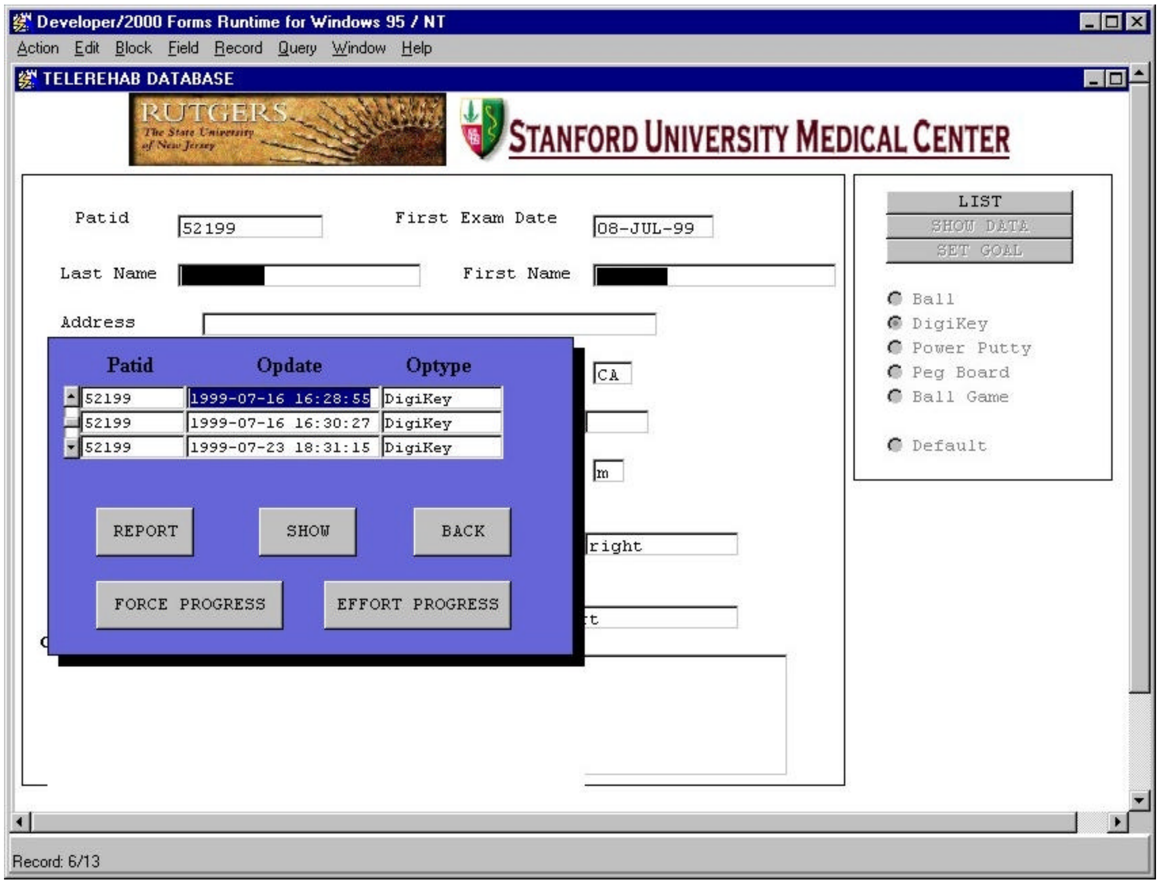
Figure 6.4: Clinical Database: a) graph containing force (in lbs) “raw data” vs. time (in seconds) for a given session; b) a patient's progress history vs. the set target for the thumb.

A higher level of data abstraction is the time history of patient performance over a series of rehabilitation sessions. In addition to calculating average, standard deviation and effort parameters for each finger during a rehabilitation session it is important to represent their variation over time. Time here is represented by the dates of the rehabilitation sessions the patient executed. Using Oracle Graphics and PL/SQL we created a time view of these parameters over several rehabilitation sessions (Figure 6.4-b.). The graph shows a target (goal) parameter, which is set by the therapist at the start of the therapy. The patient has to achieve this target over a specified number of sessions. A therapist accessing the database can easily see how the patient progresses compared to the current target. This target can be remotely modified by the clinician after assessing patient progress, in order to fine-tune the treatment to a particular patient's speed of recovery.

The ankle database reports display similar graphs of angles, forces, and torques along with the standard deviation and average for each data set. The reports allow therapists to assess the patient’s ankle capabilities. By observing the extreme values of the angle and torque graphs, therapists are able to appraise the patient’s ROM and maximum torque output around each ankle axis. By comparing the angle graphs and prescribed ankle motions, therapists can also evaluate patients’ coordination. Deficits and progress in ROM, maximum torque, and coordination can be measured by comparing data from a patient’s injured ankle with that of his or her uninjured ankle. Such comparisons using patient trial data will be shown in the next chapter. To facilitate patient progress assessment, future reports will compare patients’ data sets over time. Extreme ROM and maximum force output values for each exercise will be plotted with respect to time. As patients improve, the therapist will be able to modify exercise parameters including required duration, maximum-opposing forces, allowed ROM, and VE complexity.

6.2.2 The Graphic User Interface

Several graphical forms allow easy access to database records. The ***patient entry form*** provides the graphical interface for data input, query, update, browse, or delete of patient data. It displays a screen with all patient attributes (found in patient table) and allows the clinician to add a new patient to the database, query and update data on an existing patient and modify exercise parameters. Clicking on the ***list*** button can access a list of all the patients currently in the database. Navigation to the next database form is possible after selecting the exercise type from the list displayed on the entry form. A click on ***show data*** button pops up the ***exercise form***. This form displays a listing of sessions of specified type performed by the selected patient. It contains several buttons, which allows the user to access exercises data from clinical database. The ***show*** button displays the exercise data corresponding to the selected patient and exercise (see Figure 6.5). ***Report***, ***force progress*** and ***effort progress*** generate Oracle reports described next. The last graphical form, the ***exercise target form,*** can be accessed from the main form with the ***set goal*** button. The form displays several fields to enter the target for the selected patient/exercise. Here the therapist can enter per finger targets for each patient. At any point in this sequence of forms, the navigation to the parent form can be achieved using the ***back*** button.


Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Block Field Record Query Window Help
TELEREHAB DATABASE
RUTGERS The State University of New Jersey
STANFORD UNIVERSITY MEDICAL CENTER
LIST
SHOW DATA
SET GOAL
REPORT
SHOW
BACK
FORCE PROGRESS
EFFORT PROGRESS

Patid	52199	First Exam Date	08-JUL-99
Last Name	[redacted]	First Name	[redacted]
Address			

CA

m

right

t

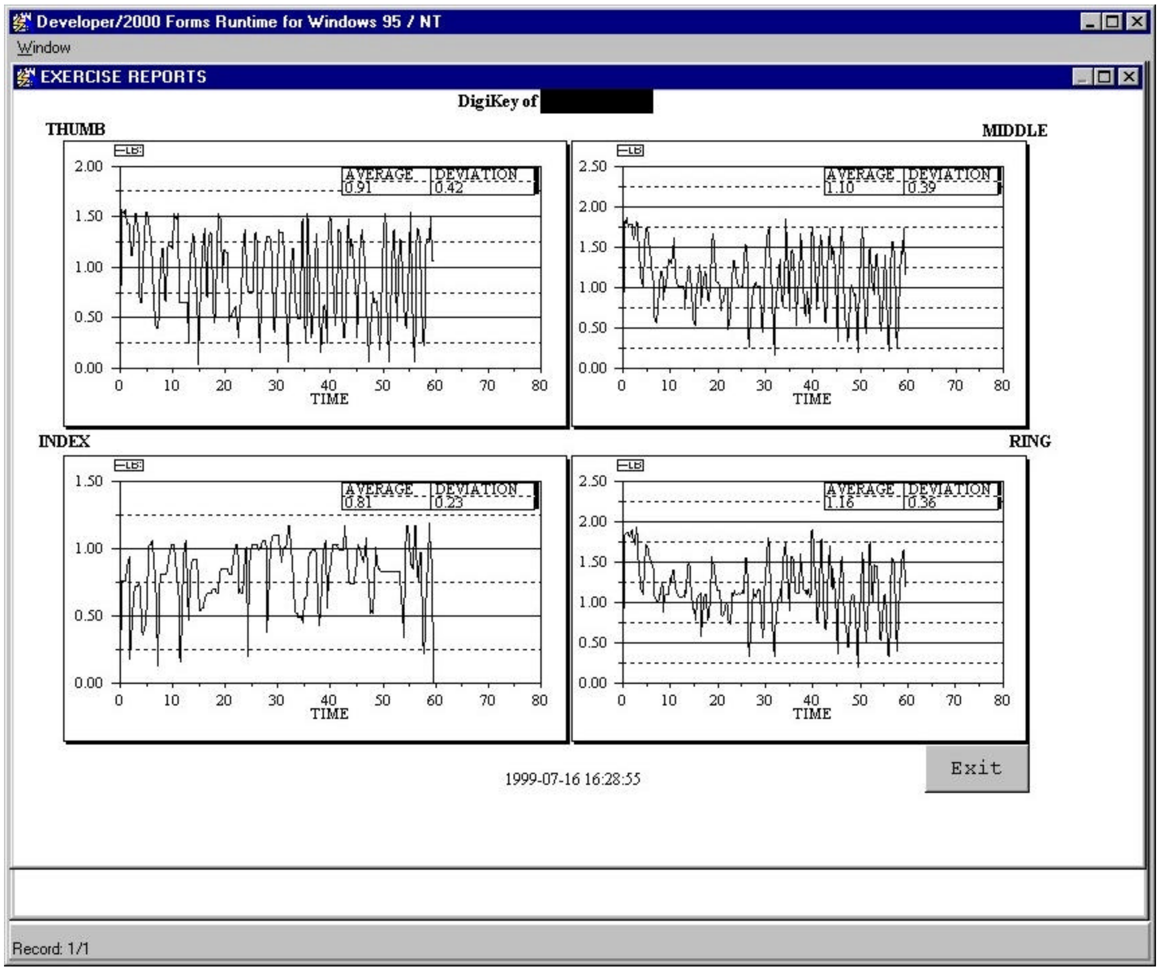
Patid	Opdate	Optype
52199	1999-07-16 16:28:55	DigiKey
52199	1999-07-16 16:30:27	DigiKey
52199	1999-07-23 18:31:15	DigiKey

Ball [✓ DigiKey] Power Putty Peg Board Ball Game Default

Record: 6/13

Figure 6.5: Database GUI - patient and exercise forms

The database reports provide the therapist with a graphical representation of the patient's data. The ***report*** button on the ***exercise form*** generates an Oracle Report for the current patient and procedure. The physical therapy reports contain graphs representing force envelopes, each with average, standard deviation and total effort parameters. The graphs were implemented using Oracle Graphics calls embedded in the report. The report is similar to the one shown in Figure 6.6, but has a more elaborate layout. The functional therapy reports are simpler as they only display rows of exercise data. Data are organized as a table containing the date of the exercises and patient performance parameters (Figure 6.7).



Developer/2000 Forms Runtime for Windows 95 / NT

Window

EXERCISE REPORTS

DigiKey of [redacted]

THUMB

AVERAGE	DEVIATION
0.91	0.42

MIDDLE

AVERAGE	DEVIATION
1.10	0.39

INDEX

AVERAGE	DEVIATION
0.81	0.23

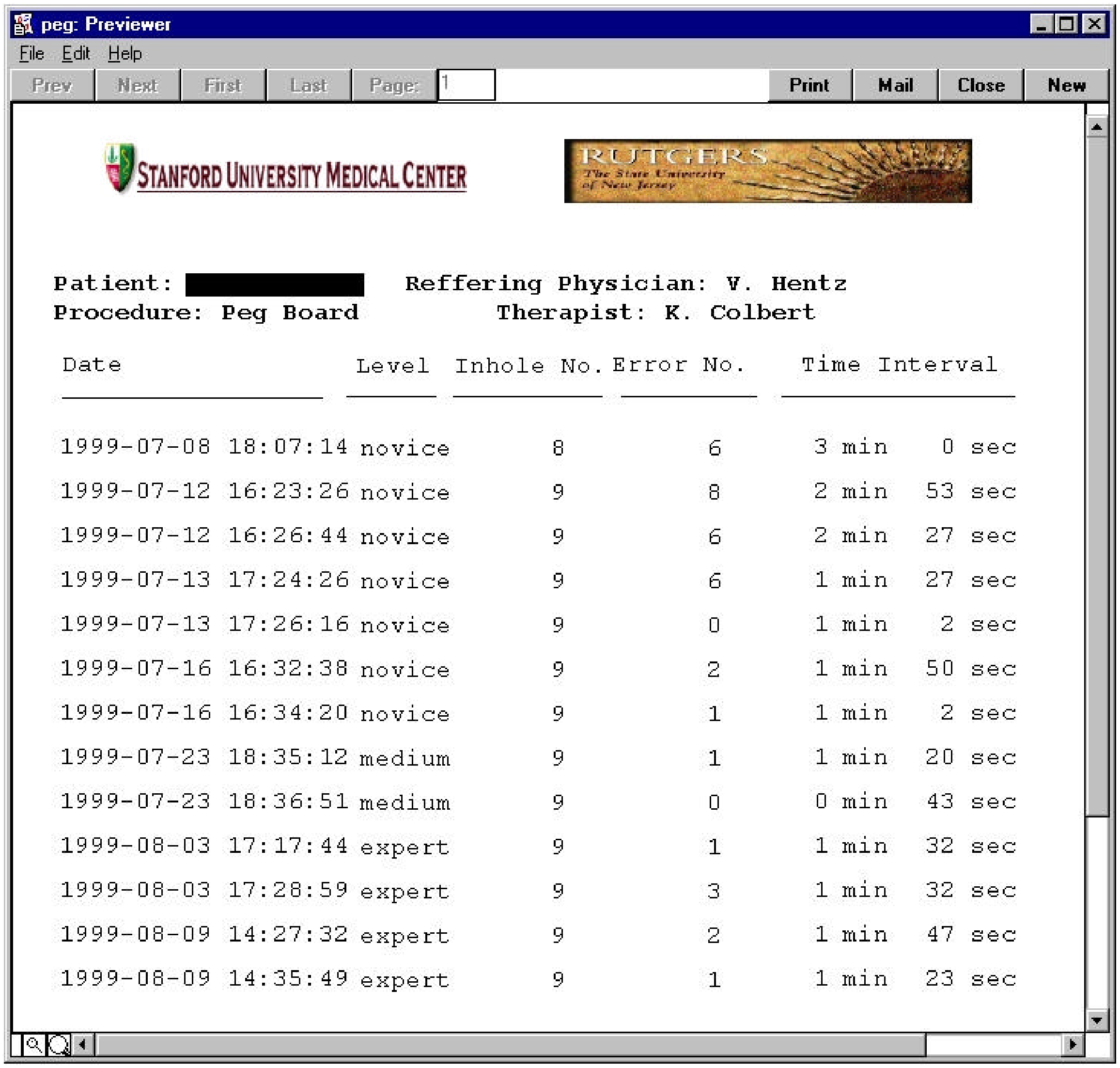
RING

AVERAGE	DEVIATION
1.16	0.36

1999-07-16 16:28:55

Record: 1/1

Figure 6.6: DigiKey exercise report (force in lbs. vs. time in sec.)

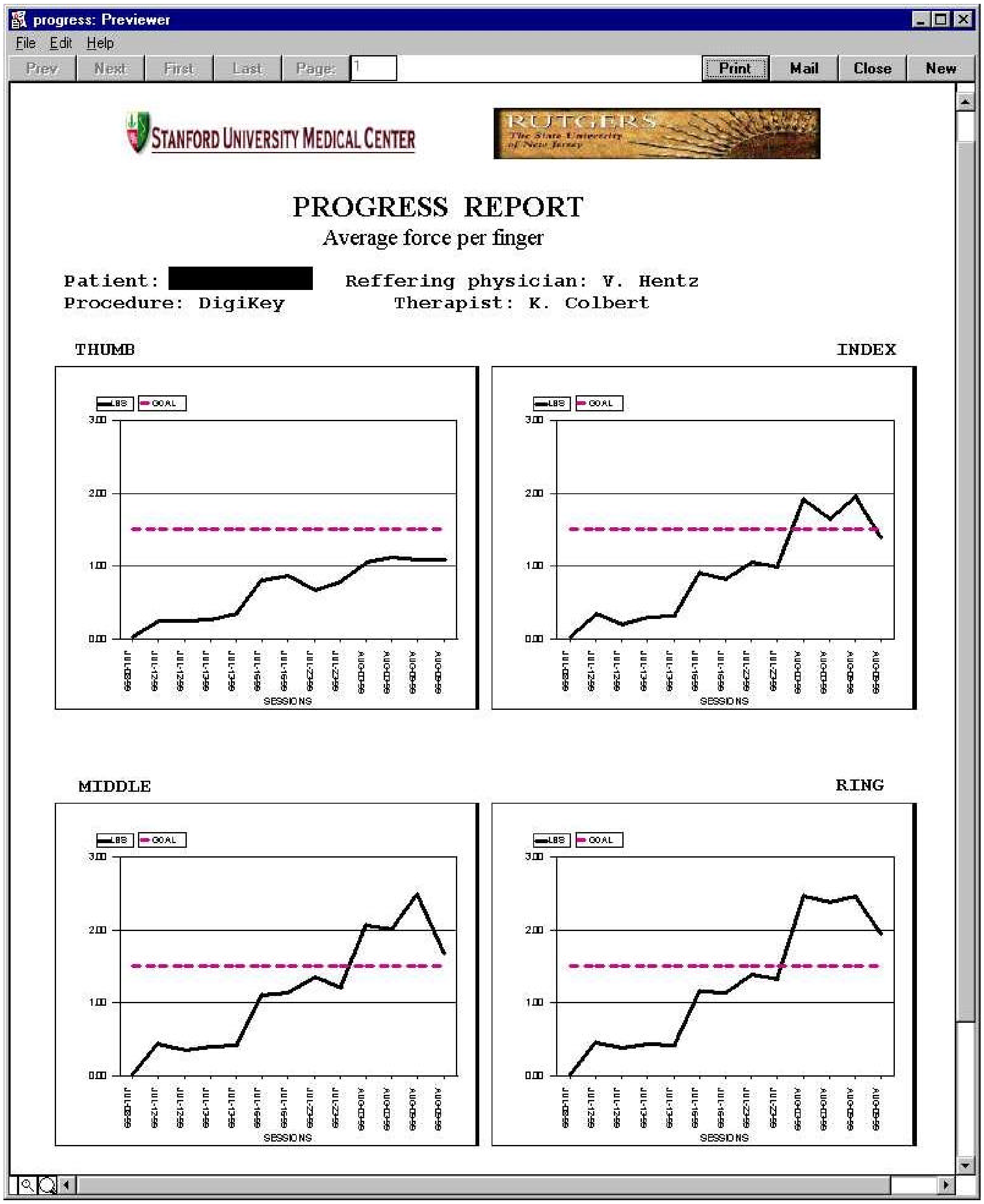

peg: Previewer
File Edit Help
Prev Next First Last Page: 1 Print Mail Close New
STANFORD UNIVERSITY MEDICAL CENTER
RUTGERS The State University of New Jersey

Patient: [redacted] Reffering Physician: V. Hentz
Procedure: Peg Board Therapist: K. Colbert

Date	Level	Inhole No.	Error No.	Time Interval
1999-07-08 18:07:14	novice	8	6	3 min 0 sec
1999-07-12 16:23:26	novice	9	8	2 min 53 sec
1999-07-12 16:26:44	novice	9	6	2 min 27 sec
1999-07-13 17:24:26	novice	9	6	1 min 27 sec
1999-07-13 17:26:16	novice	9	0	1 min 2 sec
1999-07-16 16:32:38	novice	9	2	1 min 50 sec
1999-07-16 16:34:20	novice	9	1	1 min 2 sec
1999-07-23 18:35:12	medium	9	1	1 min 20 sec
1999-07-23 18:36:51	medium	9	0	0 min 43 sec
1999-08-03 17:17:44	expert	9	1	1 min 32 sec
1999-08-03 17:28:59	expert	9	3	1 min 32 sec
1999-08-09 14:27:32	expert	9	2	1 min 47 sec
1999-08-09 14:35:49	expert	9	1	1 min 23 sec

Figure 6.7: PegBoard exercise report

The reports created with ***force progress*** and ***effort progress*** buttons use additional data processing before displaying the graphs. These reports get all the data available per exercise session and calculate a filtering expression (average, sum, etc.). The result is displayed as a function of session number (Figure 6.8). This way the evolution of the specified parameter (average force, average effort, etc.) can be tracked in time.



PROGRESS REPORT

Average force per finger

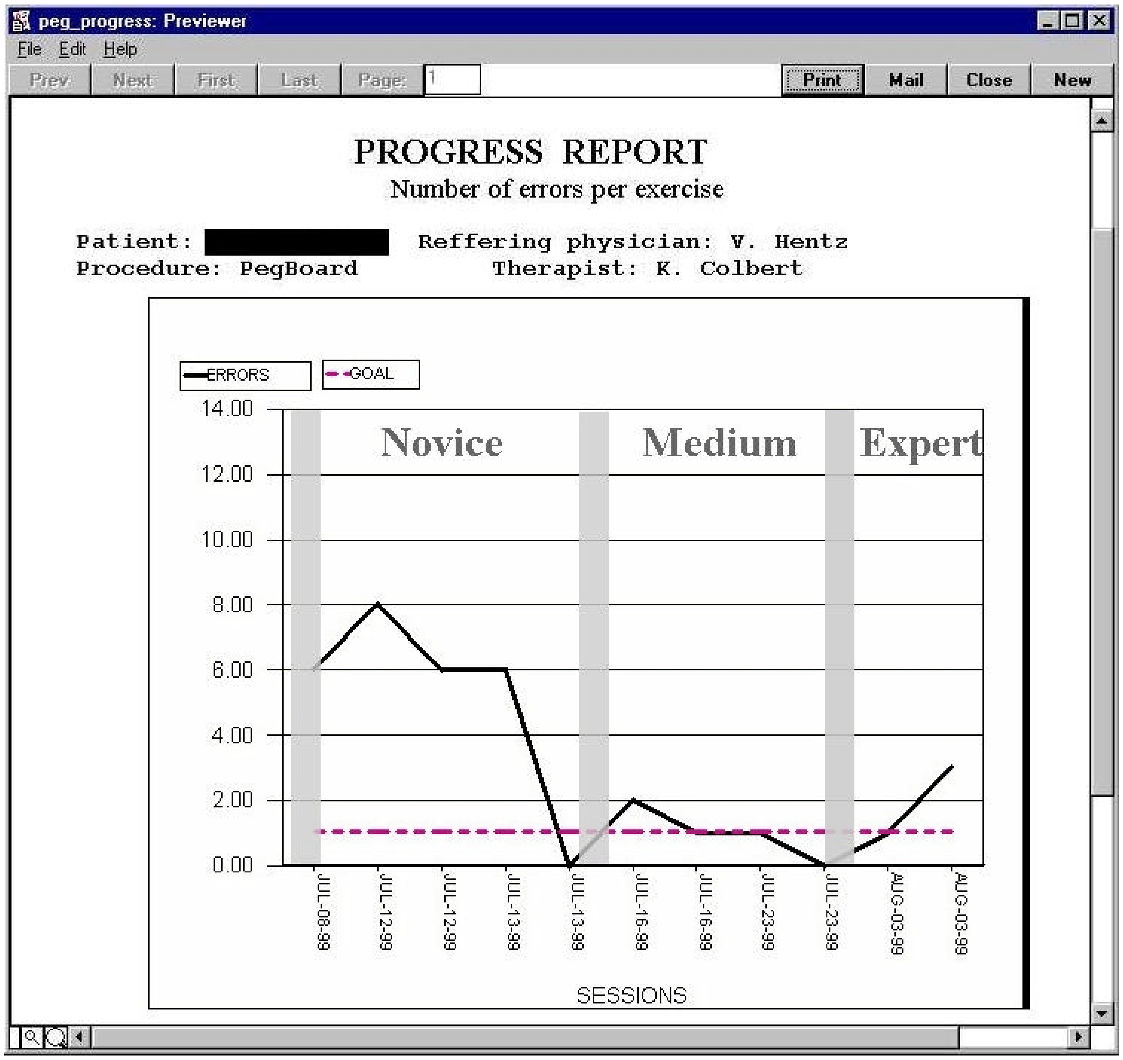
Patient: [redacted] Reffering physician: V. Hentz

Procedure: DigiKey Therapist: K. Colbert

Figure 6.8: DigiKey progress report (Burdea et al., 2000)

The graphs also show the target level that needs to be achieved by the patient after a specified number of sessions. This target can be modified by the clinician using the

exercise target form as explained before. The progress of the functional rehabilitation exercises can be displayed in database reports including graphs with the evolution of the tracked parameter. For the Peg exercise the tracked parameter is the number of errors made by the patient during the exercise as a function of level of difficulty (Figure 6.9). The target here indicates the maximum number of errors allowed per session.



PROGRESS REPORT

Number of errors per exercise

Patient: [redacted] Reffering physician: V. Hentz

Procedure: PegBoard Therapist: K. Colbert

Figure 6.9: PegBoard progress report (Burdea et al., 2000)

6.3 Client/Server Performance

The main function of the store-and-forward architecture is transferring the patient data into the database server. We analyzed the performance of the client program performing the database update over several types of network connection. The experiments were performed on modem (56 Kbps), T-1 (1.5 Mbps), 100 Mbps LAN and OC-3 (155 Mbps) connections. Additional data were collected from the Stanford – Rutgers pilot study.

6.3.1 Test Configuration

The first step of the client performance experiments was the assessment of network connection performance. Except the PC with the modem, all other computers had 100 Mbps Ethernet cards. The CAIP network is a 100 MBps Ethernet. The following connection were tested:

1. Modem to VRLAB (**M-VR**): this used a 56 Kbps modem from Rutgers campus to CAIP VRLAB network.
2. Stanford to VRLAB (**S-VR**): The Stanford PC was initially located on a network with access to T-1 (**S-VR-T1**); later it was moved to a network connected to Internet2 (**S-VR-I2**), which consists of 100 MBps local, OC-3 connection to vBNS backbone and OC-12 vBNS network segments.
3. Newark to VRLAB (**N-VR**): the Newark computer was located in Rutgers-Newark Campus.
4. **LAN**: These PC were connected in CAIP LAN at 100 MBps.
5. The number of network hops and average round trip time is shown below:

6. **M-VR**: 7 hops, RTT=108 ms; **S-VR-I2**: 14 hops, RTT=80 ms; **N-VR**: 5 hops, RTT=1.78 ms.

The TCP bandwidth was measured with the *iperf* (NLANR, 2000) monitoring tool. Several 1000 sec. tests were performed in order to calculate the average bandwidth. The average bandwidth of the connections was: **S-VR-I2**=6.28 Mbps; **N-VR**=17.27Mbps; **LAN**=79.5 Mbps. The *iperf* traffic snapshot for the **N-VR** connection is shown in Figure 6.10.

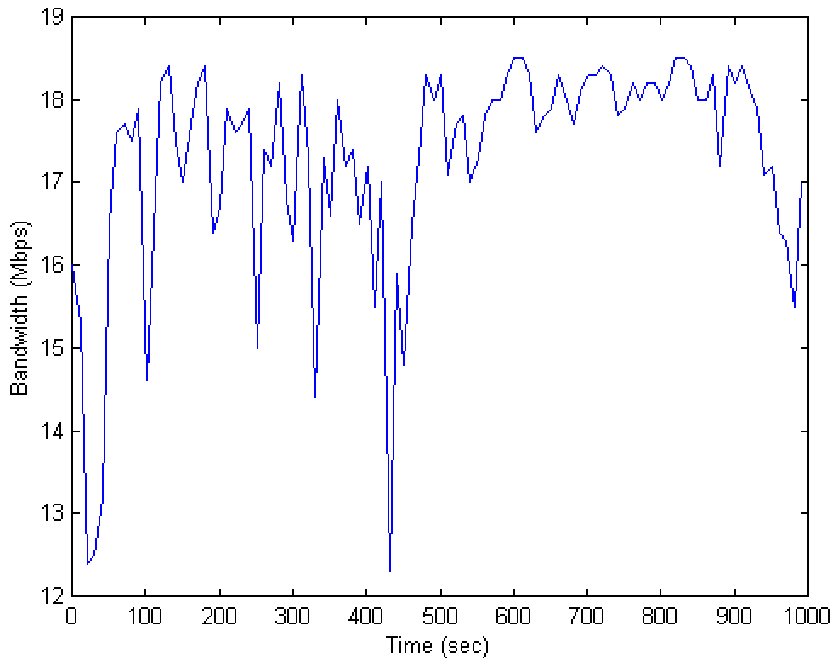


Figure 6.10: TCP measurements for the **N-VR** connection

6.3.2 Experimental Results

Two types of experiments were conducted for the database client: data transfer time vs. file size and average transfer time vs. number of simultaneous clients. Each measurement was averaged over ten database transfers. Figure 6.11 shows the results of the first experiment for **M-VR**, **N-VR** and **LAN** connections. The graphs use logarithmic axes. We observe that for small data files, the transfer time is bounded to a minim value of 1.5 seconds. This minimum is given by time needed for the database connect and login operations. For larger data files, the transfer time varies quite linear with the file size. For large files, the transfer time per 10 KB of data is: 36 seconds for **M-VR**; 0.97 seconds for **N-VR** and 0.57 seconds for **LAN**. Since in our case the database records are small, this number is dependent on the round trip time for large bandwidth connections. When the bandwidth is small (in the **M-VR** case) both RTT and bandwidth influence the total transfer time.

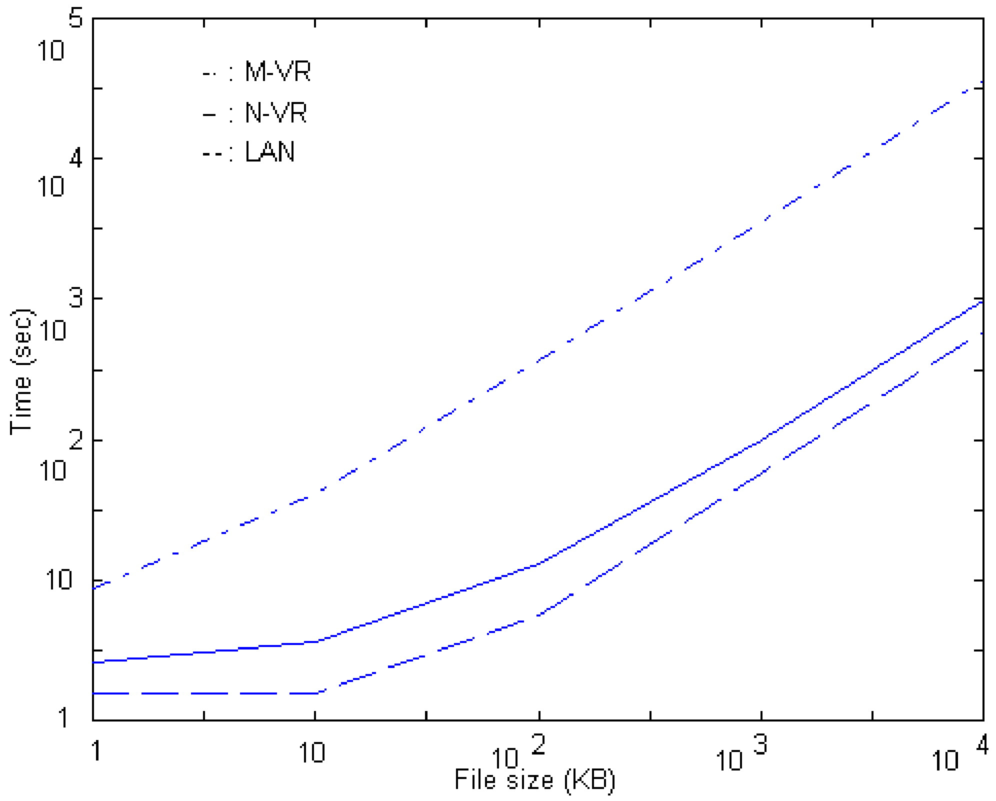
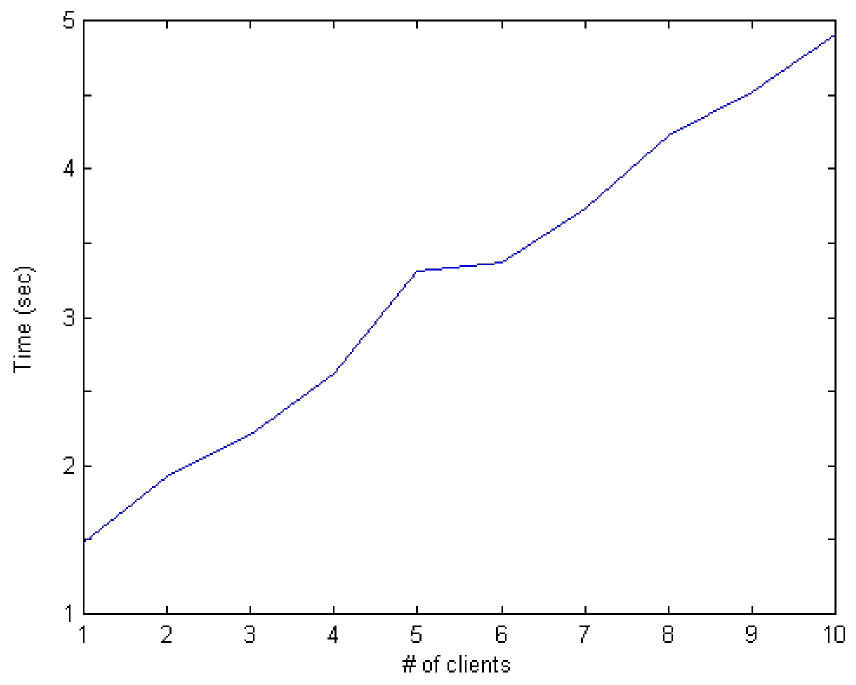


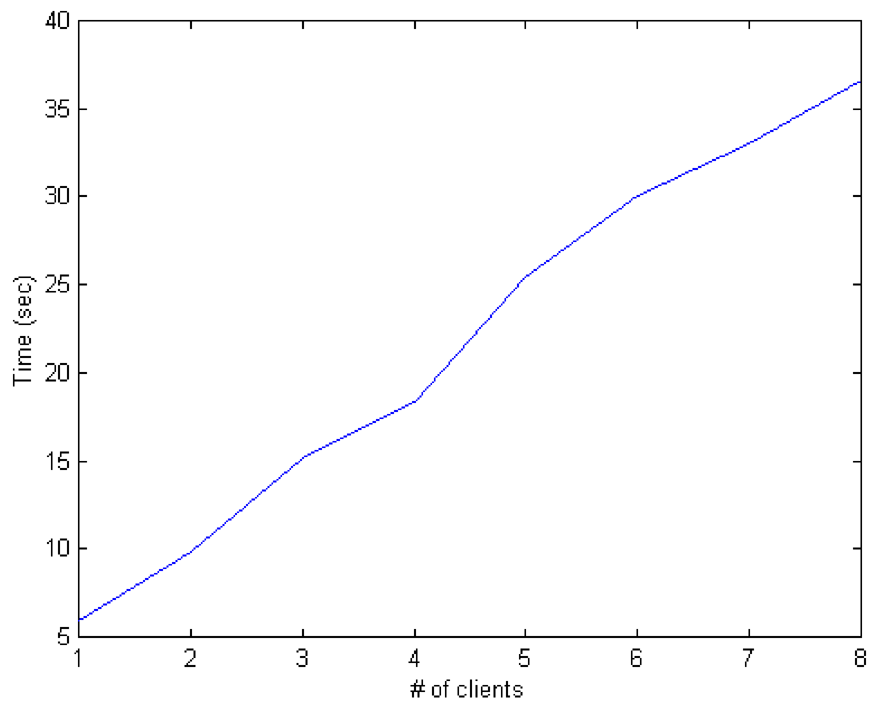
Figure 6.11: Variation of transfer time with exercise file size for the modem (M-VR), OC-3 (N-VR) and LAN connections.

Next experiment measured the variation of data transfer time with the number of concurrent database clients. The experiments were performed over the CAIP LAN. All clients use the same type of network connections. As shown in Figure 6.12 the average transfer time varies linearly with the number of clients. For 10 KB data files, the addition of a client increases the transfer time by about 25%. For the 100 KB file the addition of a client increases the transfer time by 72%. The 10KB experiment, representative of small data files, shows good behavior with concurrent users. The 100 KB case, representative for large data files, shows a sequential data forward pattern of the database client.

Experimental data collected from the Stanford - Rutgers pilot study (**S-VR-T1** connection) is shown in Fig 6.13. The patient exercises produced 9 KB (physical rehab) and 0.5 KB (functional rehab) of data. The average transfer time was 31.85 seconds for the physical rehabilitation exercises (DigiKey, Ball, Putty) and 6.56 seconds for functional rehabilitation (Ball Game, Peg Board). The standard deviation was quite large in both cases: 11.82 sec. for physical rehabilitation (about one third of the mean) and 3.75 for functional rehabilitation (about half of the mean). This could be explained by the network variability, but also by the server CPU load during the experiment. The database server was installed on a computer used also for demonstration and development that produced at times significant CPU load.

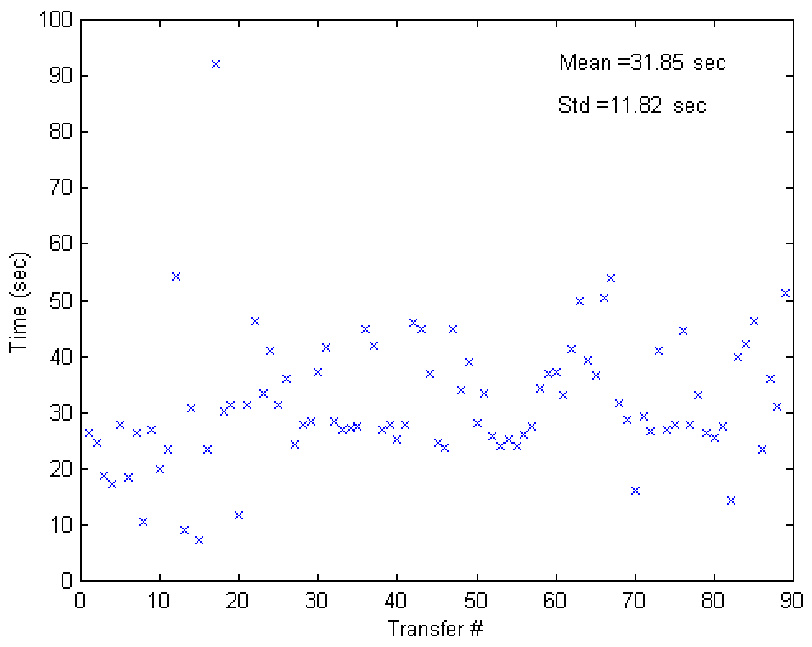


a)

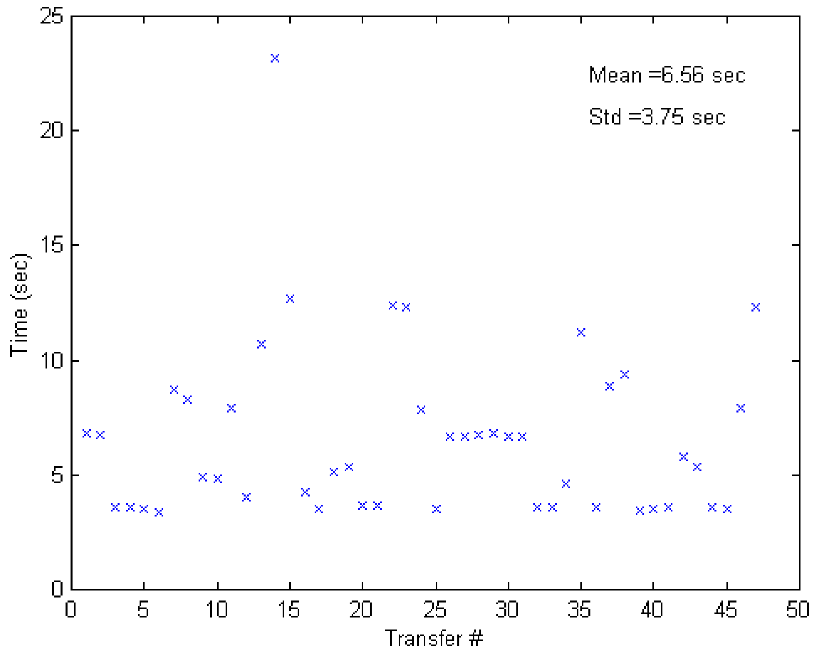


b)

Figure 6.12: Variation of transfer time with the number of clients on a LAN: a) 10 KB file; b) 100 KB file.



a)



b)

Figure 6.13: Stanford clinical trials: a) transfer time for physical rehab exercises; b) transfer time for functional rehab exercises.

The above experiments aimed at characterizing database client performance over different network connections. The studies performed analyzed the client behavior with the patient data collection requirements (collected file size) and the overall size of the system (number of clients). The results of these experiments will be useful for deploying such a “store-and-forward” system in future projects.

6.4 Conclusions

The PC-based rehabilitation system described in the previous chapters was used in a “store-and-forward” telerehabilitation architecture. Data collected during the exercises was stored remotely at the server site (clinic). Here the therapist can analyze it, evaluate the patient's progress and modify VR exercise parameters or rehabilitation goals over the network. Remote consultation is also supported using a videoconferencing system. Client/server performance was studied over several types of connections. The analyzed performance indicator was data transfer time. Transfer time depends on several system parameters such as data file size and the number of concurrent users. File size, in turn, depends on other parameters such as data sampling rate, exercise duration, type of stored data, etc. The designer needs to select these parameters following system-sizing guidelines. Our experiments led to the following recommendations:

- calculate the sampling rate based on human subject performance measurements: estimate from experiments the maximum frequency of the collected data and then apply Nyquist criterion;
- add data compression when using large data files over slow networks;

- extract and store, when possible, higher level parameters (e.g. average, number of errors, etc.) instead of raw data, as this decrease transferred file size while providing a better representation of the medical data;
- keep the transfer time per exercise in the order of tens of minutes, so that several exercises from a single patient can be forwarded in less than an hour;
- for large exercise data files, calculate *load* (in seconds) as the number of clients multiplied by the average transfer time per day per client;
- keep the *load* defined as above under system cycle time (e.g. 24 hours if exercises are performed daily).

Chapter 7

Clinical Trials

The VR-based rehabilitation system was tested in several clinical trials. The hand rehabilitation trials were performed at Stanford Medical School as part of the NSF-sponsored telerehabilitation project. Proof-of-concept ankle rehabilitation trials were completed in collaboration with UMDNJ. Several other clinical trials are currently being performed for post-stroke patients using redesigned VR exercises (Jack et al., 2000), (Deutsch et al, 2000). We present in this chapter the collected results of hand and ankle clinical trials.

7.1 The Stanford - Rutgers Pilot Study

One of the goals for Stanford-Rutgers pilot study was the testing of medical efficacy of the hand rehabilitation system. The rehabilitation unit (patient site) was installed at Stanford Medical School in Fall 1998. At that time the unit was demonstrated to the therapist in charge of supervising the patient trials. A short training session was also given to the medical personnel in order to explain different functions of the system (database, videoconferencing, on-line tutorial, etc.). From October 1998 to October 2000, Rutgers VRLAB provided the software and hardware support for the rehabilitation unit.

The VR exercises were modified as requested by the therapist to better fit patient hand rehabilitation. During the patient trials the force feedback glove was replaced several times. The gloves were more stressed and had a shorter life span than originally estimated. This lead to the redesign of the force feedback glove in a more rugged version (RMII-ND). Several glove sizes needed to be built to better fit patient hand sizes. In the summer of 1999 the computer installed in Stanford was replaced with a newer one with better graphic capabilities. Due to this trial and error process, the patient trial time frame was extended beyond the scheduled end date of the project. Until October 2000, three patients were referred for VR-based hand rehabilitation. The experimental protocol and the results collected from these patients are presented in the following sections. Data were generated on the West Coast (Stanford) and uploaded remotely over the Internet to the East Coast server (Rutgers).

7.1.1 The Experimental Protocol

The therapist collaborating on the project at Stanford Medical School developed the hand rehabilitation protocol. The protocol was designed for patients that undergo a surgical procedure for carpal tunnel syndrome. The protocol recommends the patient standard rehabilitation exercises combined with virtual therapy exercises without force feedback during the first two weeks after the surgical operation. These virtual reality therapy should be performed two-three times per week, one set of exercises per session. In the third week post operative the patient is ready for the first level of resistance. The patient will not undergo any other classical clinic-based rehabilitation for his hand impairments. He executes the Virtual Ball, Virtual Putty with the soft settings and Virtual DigiKey on the second level of resistance. The Virtual Pegboard is executed at the beginner level. Each exercise is executed twice per session. After 5 weeks post operative, the patient increases the level of difficulty for each exercise and begins the Ball Game exercise. Starting with the sixth week, the patient is executing three exercises per session. In the seventh week the exercise difficulty increase to the maximum level. In the eight week the patient returns to normal activities. Virtual therapy exercises are discontinued on an individual basis, following evaluation by the therapist or physician.

7.1.2 Patient Trial Results

The patients performed the VR exercises following the above protocol, under the supervision of the therapist. Figure 7.1 shows data from a DigiKey exercise performed by a patient at the medium exercise difficulty. The amplitude of the graph shows that the patient is fully exercising the fingers, going from open to closed positions. The graphs also show that the thumb is moved less than the rest of the fingers. Thumb and index force average and total effort are smaller than those corresponding to middle and ring fingers. The progress reports capture a better the patient's rehabilitation process. The DigiKey average force per finger, showed in Figure 7.2, increased significantly during a period of two months. After about a month the average finger force surpassed the target set at half the finger force range. The thumb and the index were slightly behind the other two fingers, needing longer time to recover. The total effort per finger (Figure 7.3) follows the same ascendant curve, showing increased finger endurance to effort. The patient progress during Ball exercise is not as steady as for the DigiKey. The average finger force shows progress over an extended period of time, but its evolution is less constant (Figure 7.4). The same pattern applies to total finger effort during Ball exercise (Figure 7.5). The Putty exercise presents a large zero region, as the patient exercised without forces in the first rehabilitation sessions. Later the average force and total effort increases abruptly, as the finger strength increased due to the other physical exercises (Figure 7.6, 7.7).

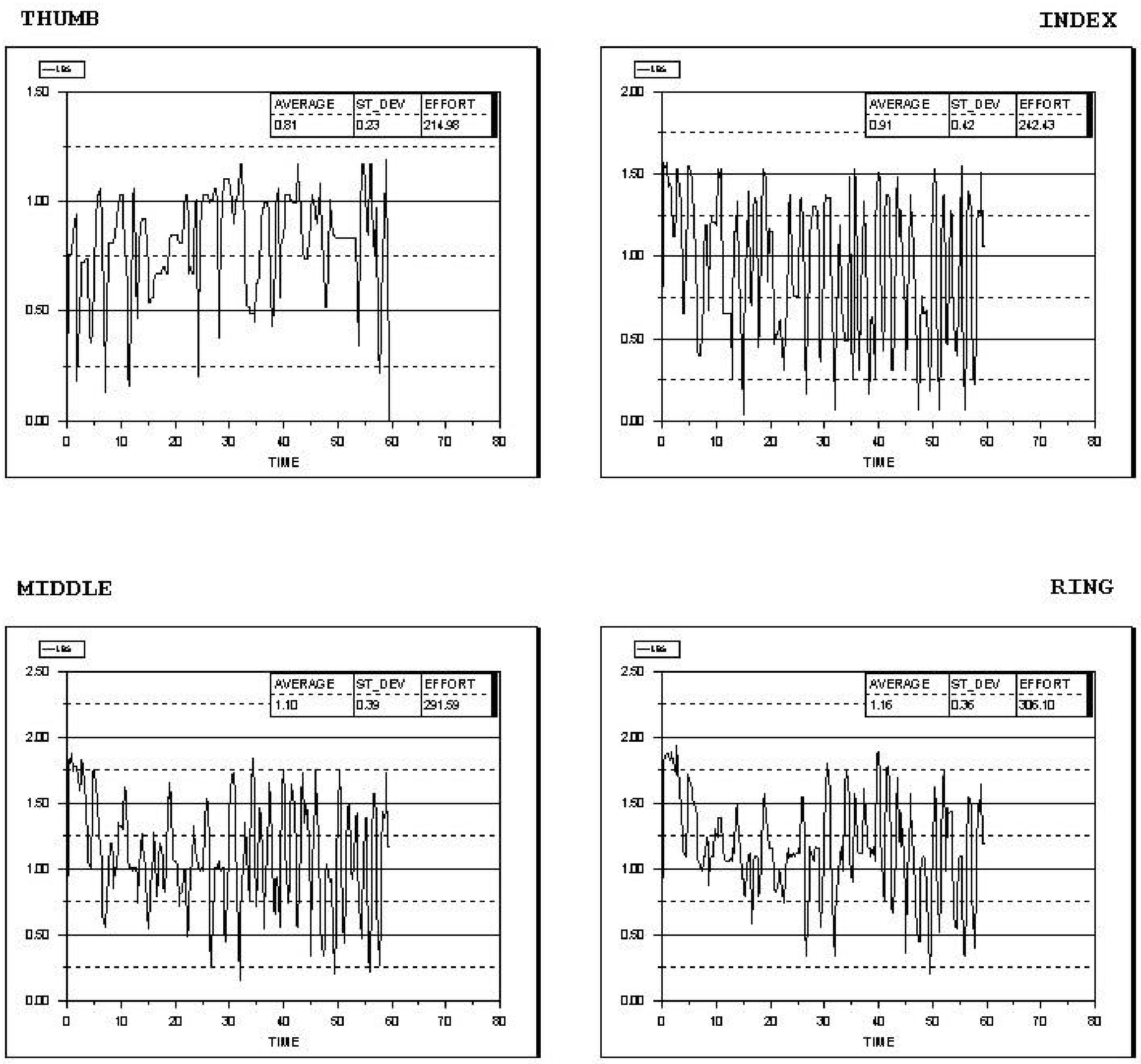


Figure 7.1: Clinical Database: Graphs containing force data for DigiKey exercise (force in lbs. vs. time in sec.).

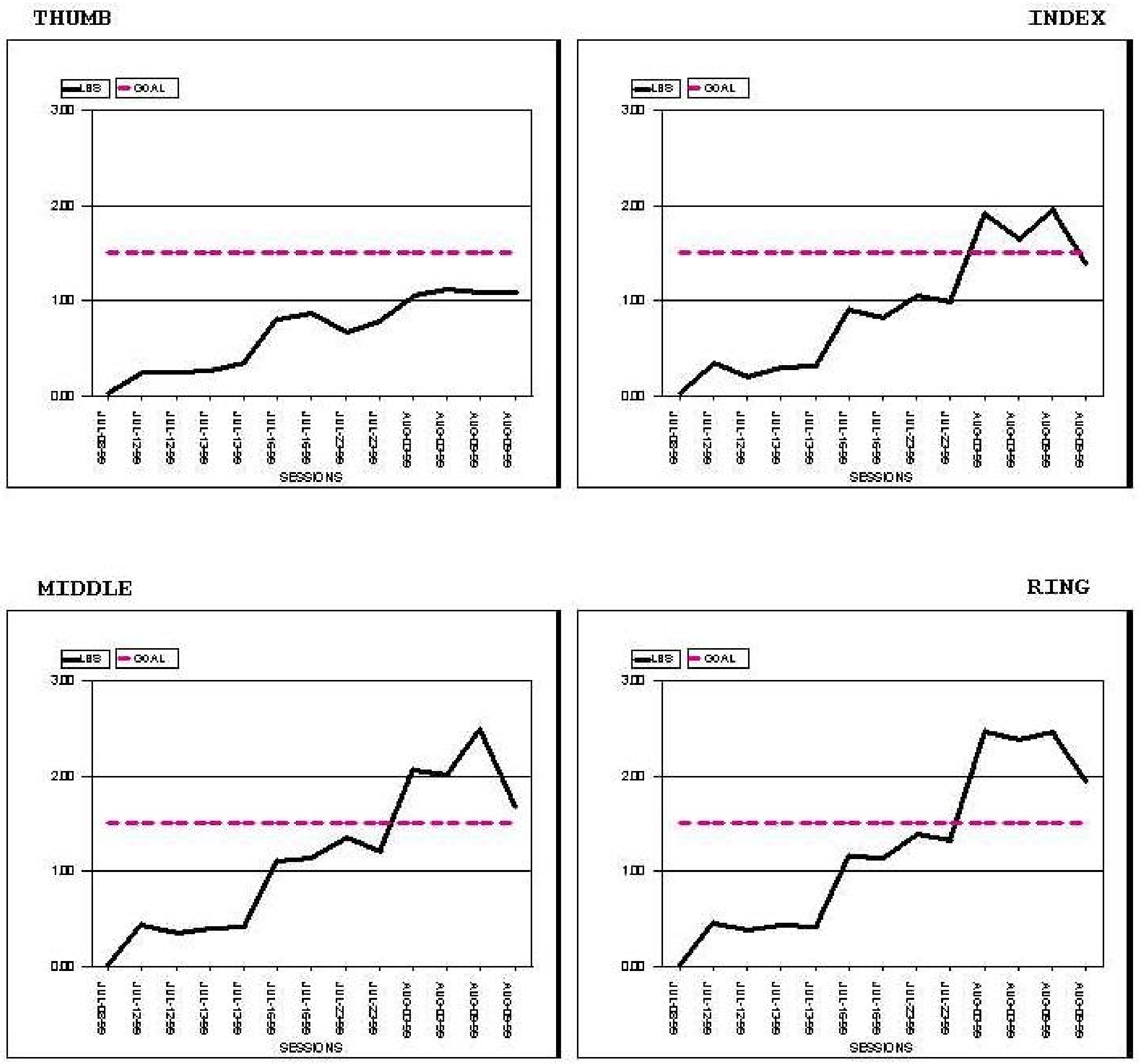


Figure7.2: Clinical Database – DigiKey exercise: patient's progress history vs. the set target; tracked parameter: average force per finger (force in lbs. vs. sessions).

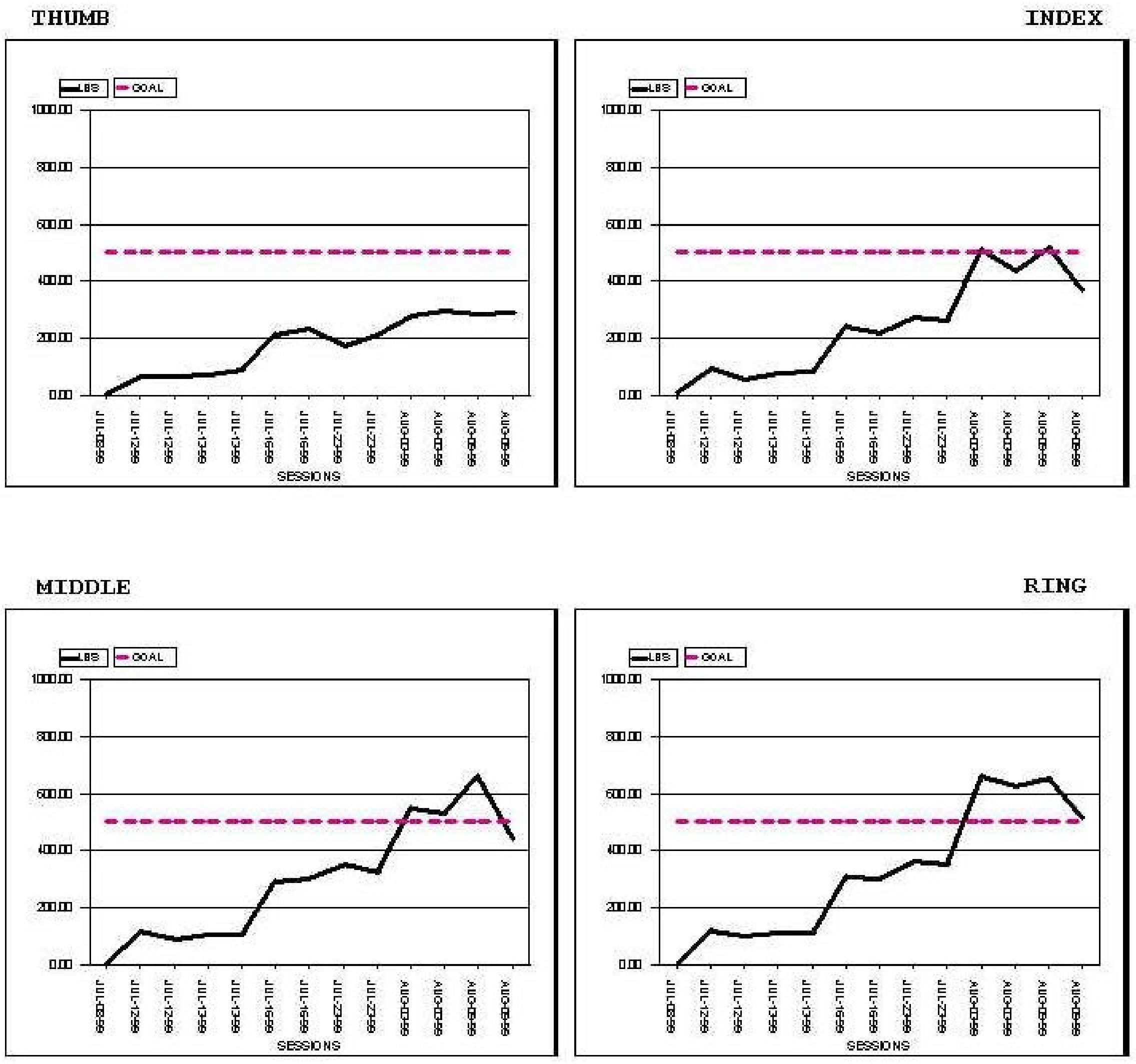


Figure 7.3: Clinical Database – DigiKey exercise: patient's progress history vs. the set target; tracked parameter: total effort.

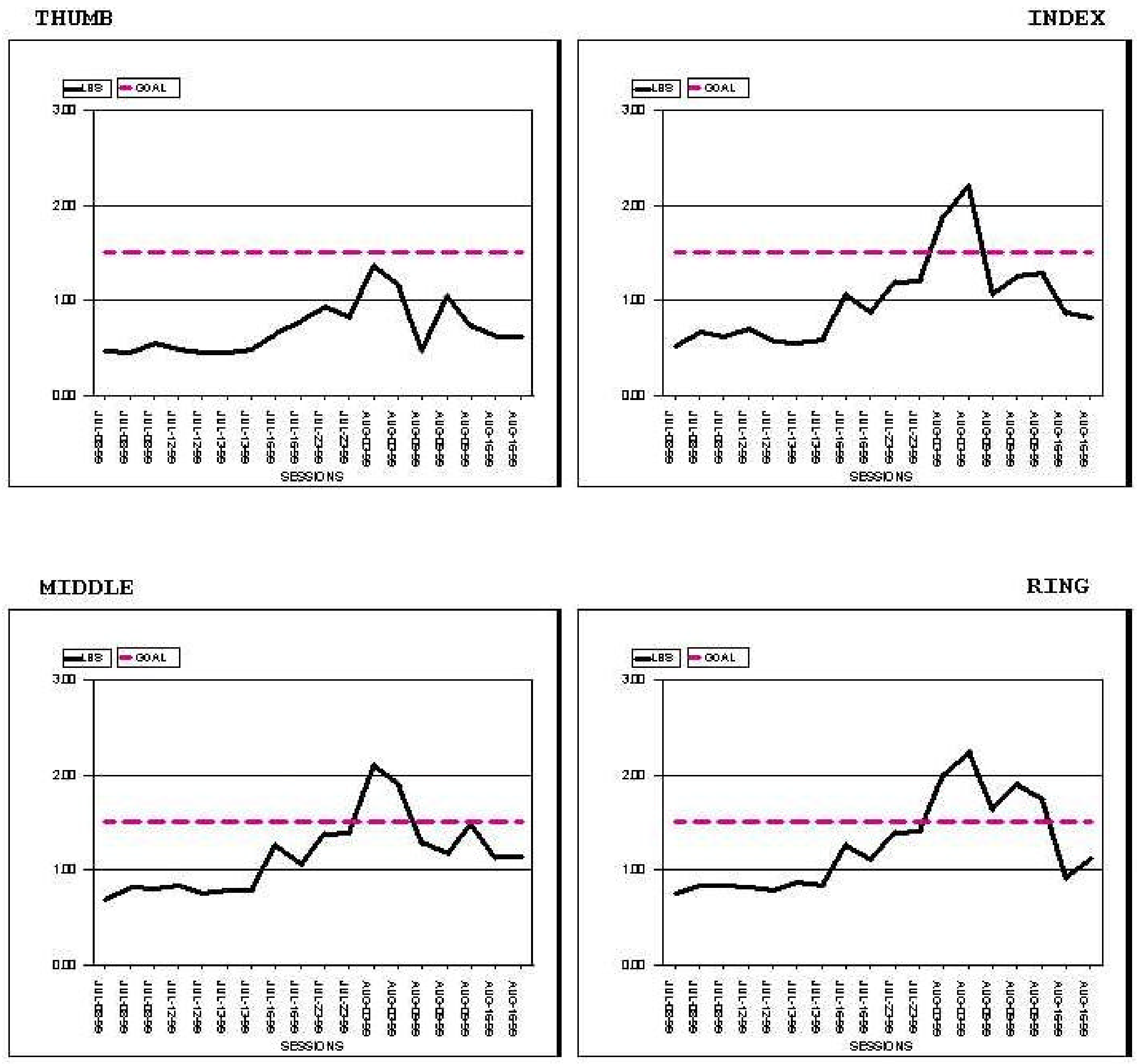


Figure 7.4: Clinical Database – Ball exercise: patient's progress history vs. the set target; tracked parameter: average force per finger (force in lbs. vs. sessions).

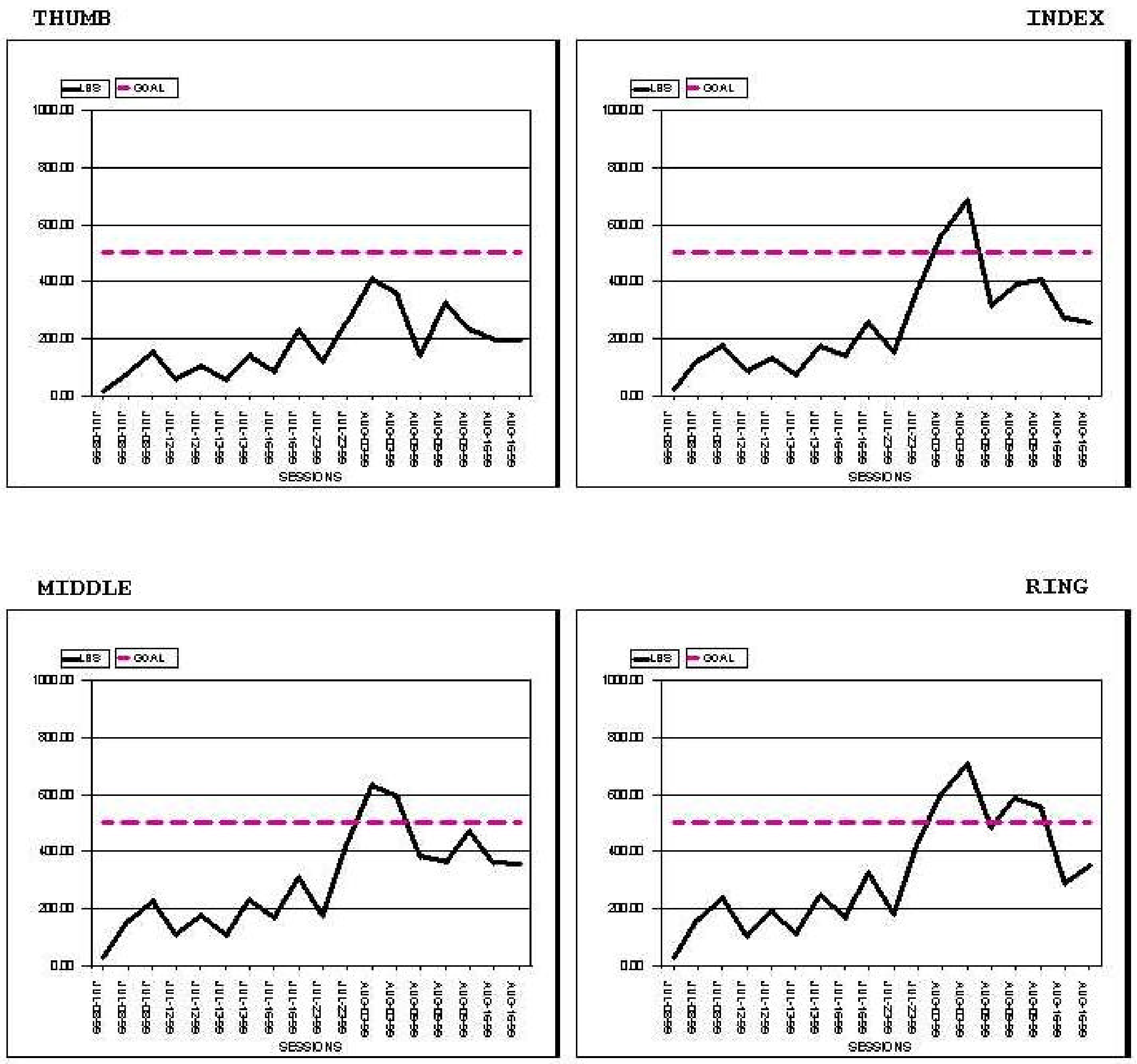


Figure 7.5: Clinical Database – Ball exercise: patient's progress history vs. the set target; tracked parameter: total effort per finger.

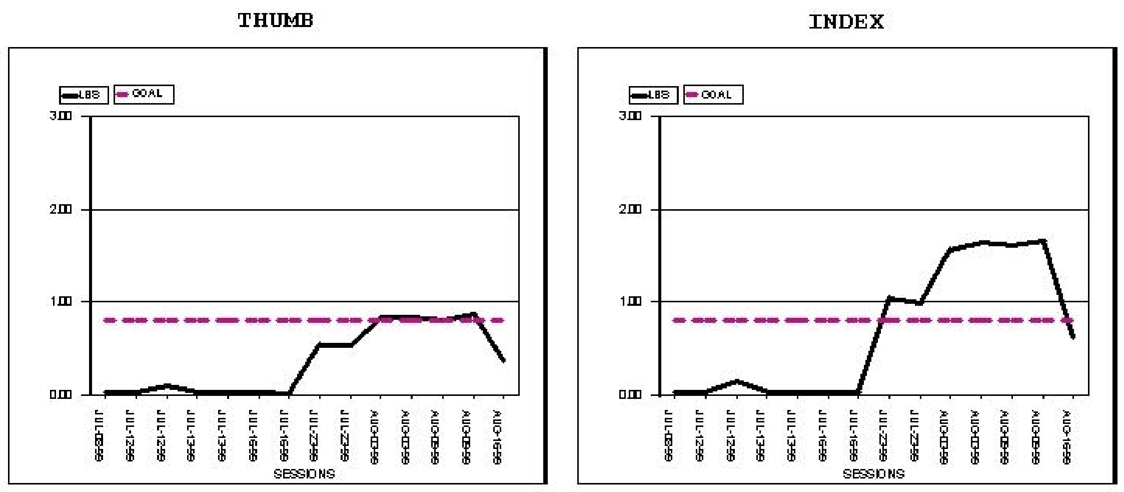


Figure 7.6 Clinical Database – Putty exercise: patient's progress history vs. the set target; tracked parameter: average force per finger (force in lbs. vs. sessions).

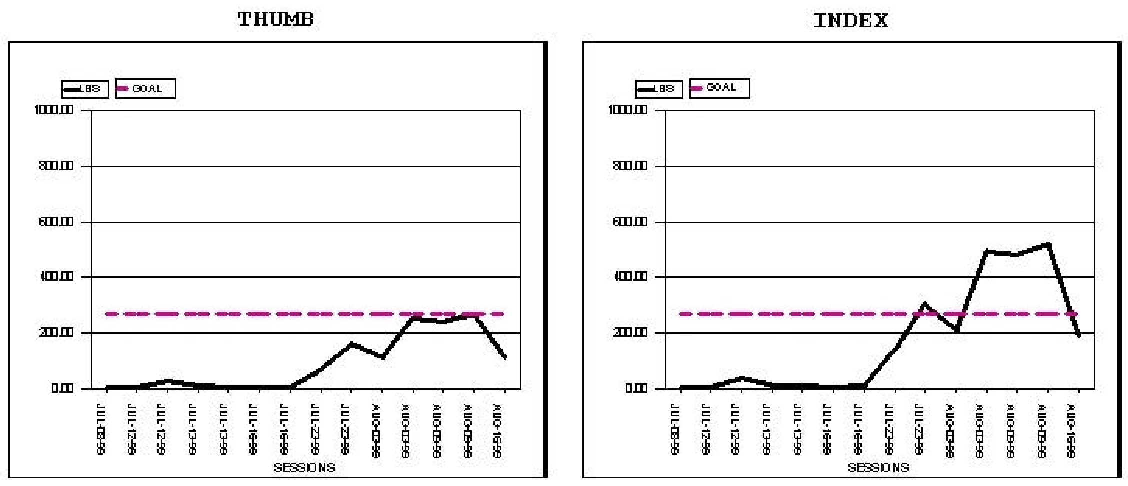


Figure 7.7 Clinical Database – Putty exercise: patient's progress history vs. the set target; tracked parameter: total effort per finger.

Hand-eye coordination function improved as well during the rehabilitation therapy. Figure 7.8 shows the progress of the patient during the Peg Board exercise. The patient started at *Novice* level and progressed to *Expert* level. The patient started with a large number of errors and reduced them as he improves his grasping skills. When changing exercise difficulty level, the number of errors goes up, but decreases after a couple of sessions. In conclusion, all exercise data showed that the patient was progressing during the VR-based rehabilitation protocol.

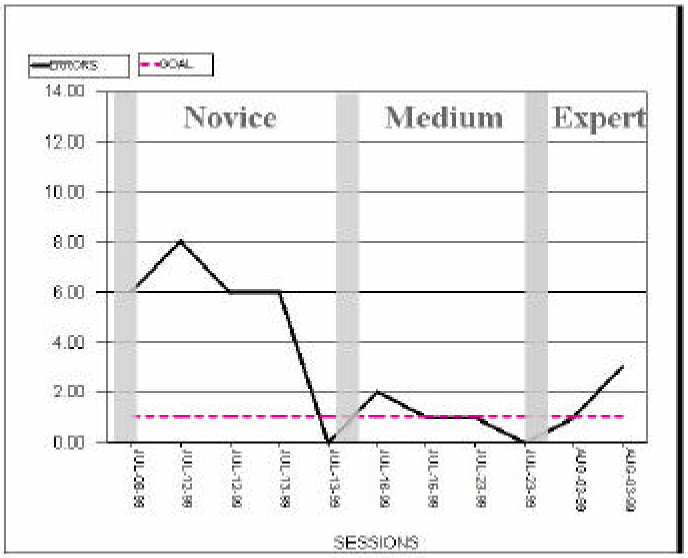


Figure 7.8: Clinical Database – Peg Board exercise: patient's progress history vs. the set target (errors vs. sessions).

Data collected from the other patients also showed progress during the rehabilitation process. As shown in Figure 7.9 and 7.10, the average force per finger and total effort went up during next patient trials. Glove mechanical problems affected patient performance during the first weeks, but the recovery started as soon as these were fixed. Data from the third patient showed a similar pattern. More patient trials are needed to gather sufficient data for understanding the VR-based rehabilitation process. Analysis of such clinical data will then be part of a follow up rigorous biomedical research study.

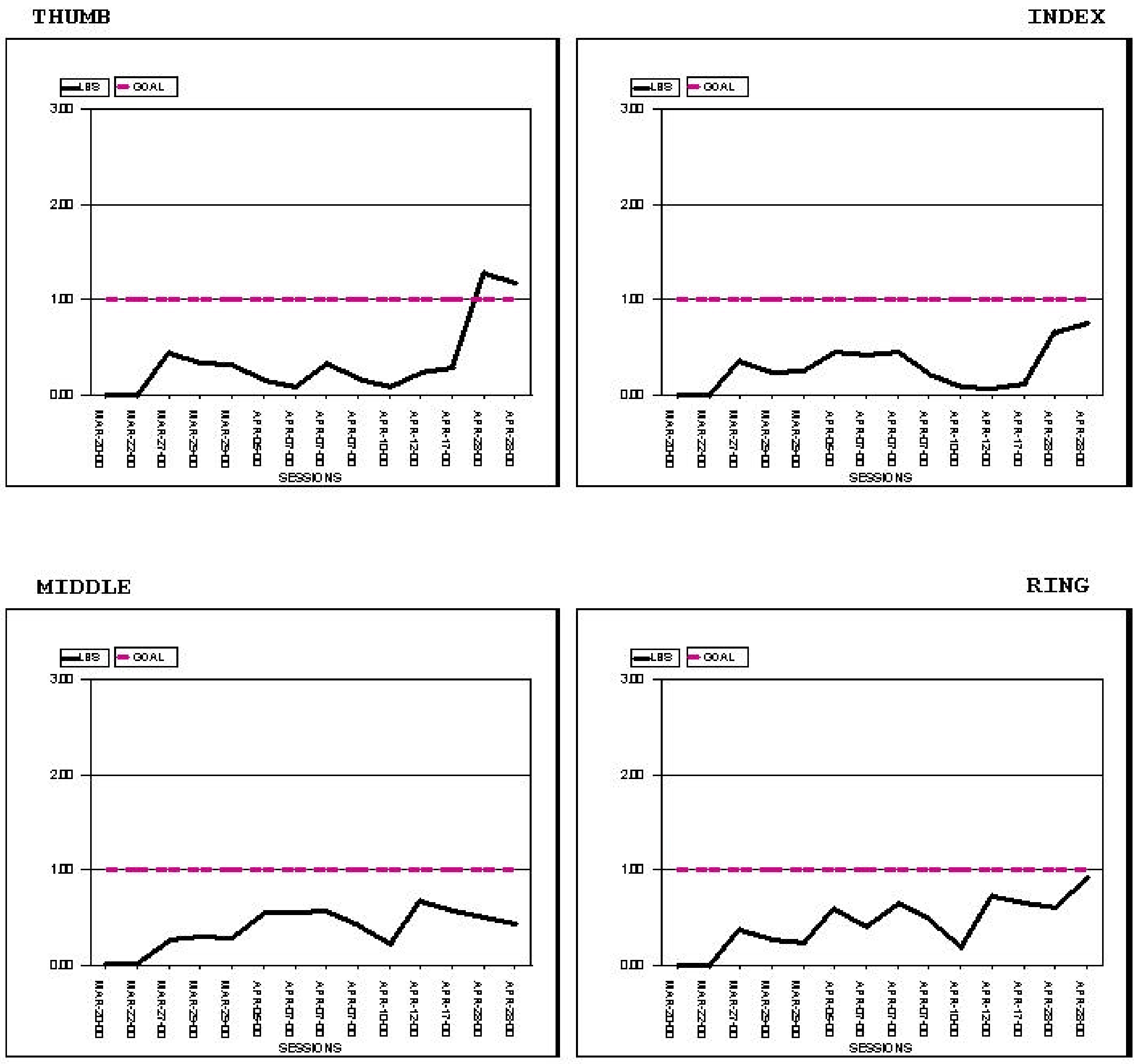


Figure 7.9: Clinical Database – DigiKey exercise: patient's progress history vs. the set target; tracked parameter: average force per finger (force in lbs. vs. sessions).

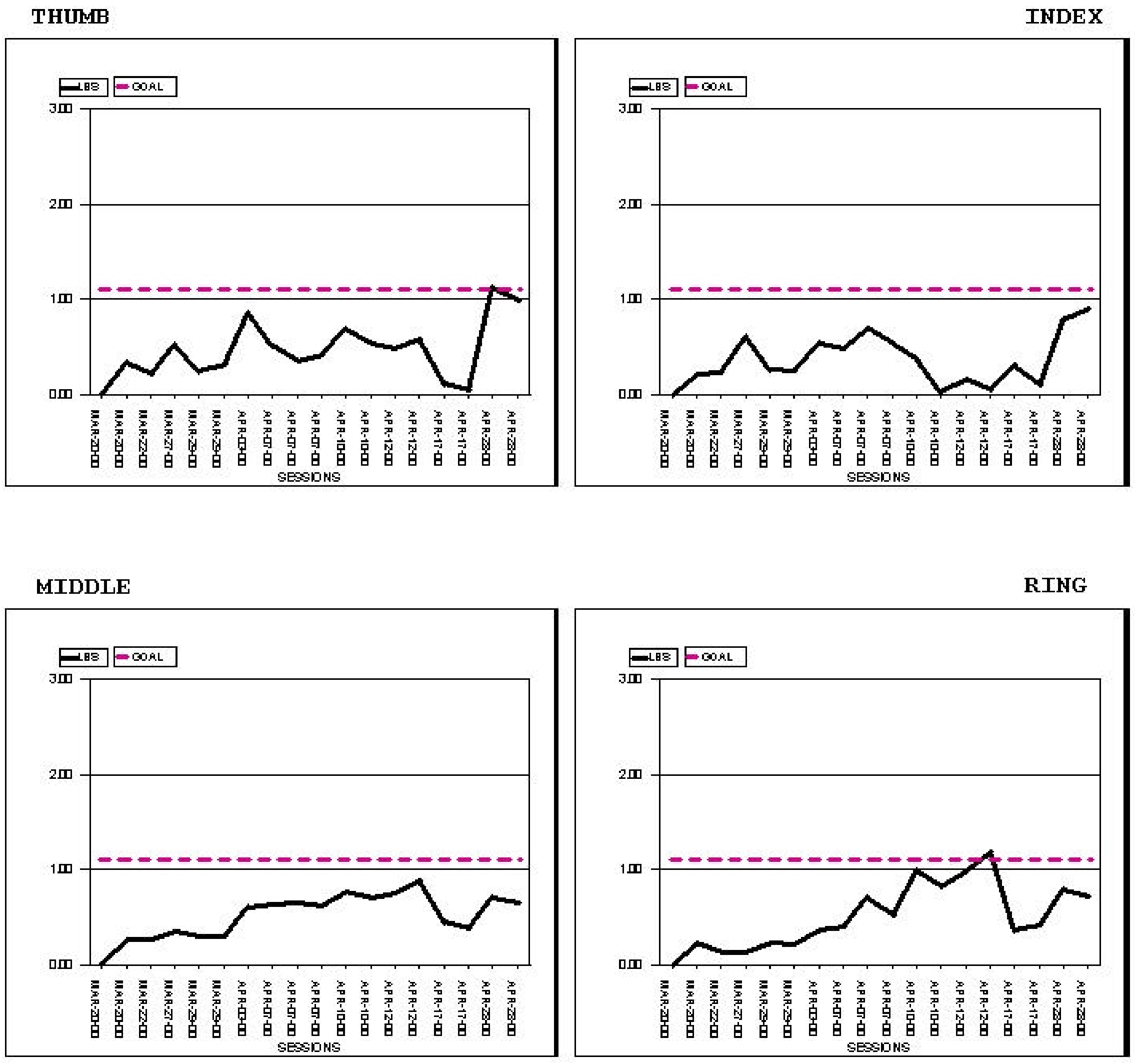


Figure 7.10: Clinical Database – Ball exercise: patient's progress history vs. the set target; tracked parameter: average force per finger (force in lbs. vs. sessions).

7.2 Ankle Experiment

7.2.1 Proof-of-Concept Trial

An orthopedic rehabilitation system using the ankle device was tested in the summer of 1999. The study used the rehabilitation system for patients with lower extremity pathology affecting ankle mobility and function. The proof-of-concept patient trial study was conducted at the University of Medicine and Dentistry of New Jersey (UMDNJ). The three females and one male participating in the study were heterogeneous in age (26-81 years), computer experience (0-13 years), and clinical presentation. Two patients (Patients 1 and 2) exhibited hypermobility secondary to chronic ankle instability and the other two presented with hypomobility as the sequelae of fractures.

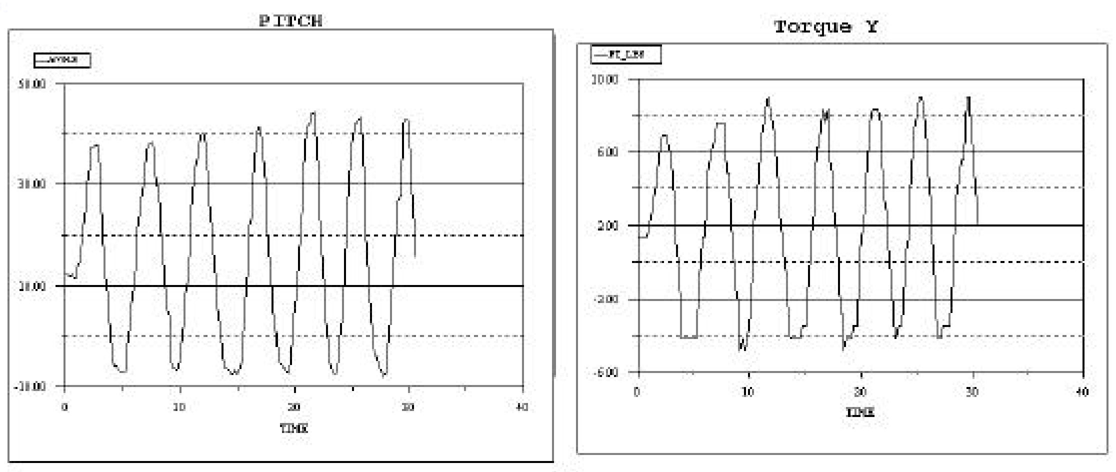
A physical therapist examined impairments of ROM, strength, and balance as well as sensation, pain, and skin condition. Patients presented varying degrees of ROM loss, weakness, standing balance deficits, as well as edema. Their functional status was determined by interview. Two of the patients were independent community ambulators, who could walk distances longer than 153 m and could negotiate curbs, ramps and stairs. These patients had limitations with recreational and sports activities. The other patients required assistive devices for gait and had limitations with distances. One was a household ambulator with mobility limited to level surfaces and a distance of less than 30 meters.

Patients signed an informed consent and were given an explanation of the study's purpose. They were assisted to the chair placed in front of the host PC, and their foot was positioned in the "Rutgers Ankle" foot restraint. Then they perform the ankle VR exercise described in the previous chapter. Patients had an opportunity to practice a few repetitions prior to data collection. Data were collected in 30-60 second intervals while subjects performed movements at either their self-selected or fastest speeds. Patients performed ten trials, each alternating between dorsiflexion/plantarflexion and inversion/eversion motions. There were one to two minutes of rest between each trial. Over the course of the ten trials, the device's opposing force and upward support were decreased incrementally. High opposing forces and upward support challenged patients' strength while low values targeted their endurance and coordination. At the end of each trial, subjects were asked to report any symptoms of pain or fatigue. Following the study, subjects were asked to complete a subjective evaluation questionnaire concerning their ease in learning to use the interface and its perceived utility.

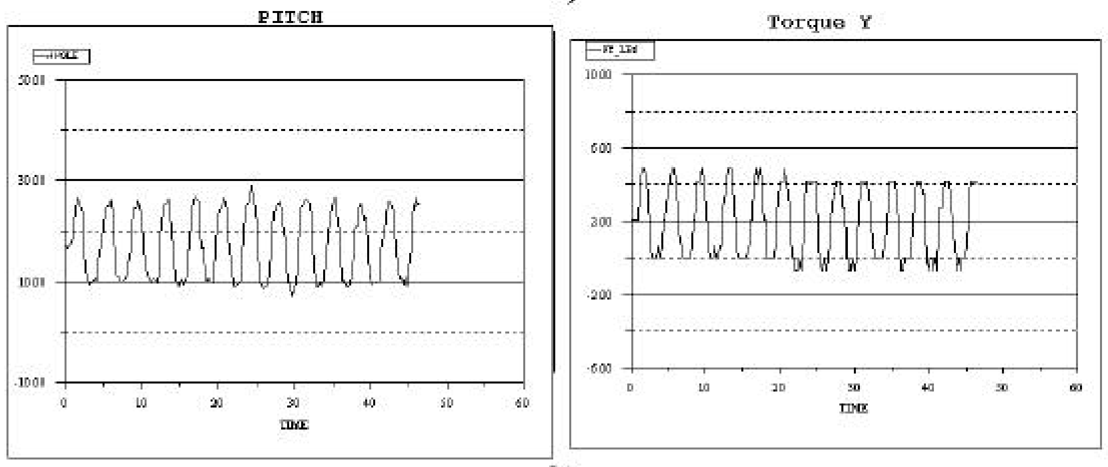
7.2.2 Results

All the patients learned to use the VR interface quickly. There were some reports of fatigue for those patients working at their fast velocity. There were no reports of pain. All subjects responded favorably to the use of the device and stated that they would enjoy having this device complement their current rehabilitation programs. The most notable limitation was the delay between ankle movement and the corresponding graphics feedback. This was particularly troublesome for the patients who were asked to exercise at their highest speeds. The lack of support for holding the limb in place while the interface was set at the lower levels of upward support was also reported as a concern.

Patients' displacement and torque data were collected and analyzed. This allowed the therapist to evaluate the patients' deficits. The performance of the involved and uninvolved ankle was compared. Figure 7.11 illustrates the larger displacement and torque generated by the uninvolved leg compared to that of the involved leg. The displacement of the uninvolved leg is comparable to normal ROM at the ankle with five degrees of dorsiflexion (the negative deflection on the plot) to 45 degrees of plantarflexion. The angle of the involved limb reflects a loss of ROM of –10 degrees of dorsiflexion and 28 degrees of plantarflexion. The maximum torque generated by the uninvolved limb is much larger (5.4 N•m for dorsiflexion and 10.8 N•m for plantarflexion) than that generated by the involved limb (0.67 N•m for dorsiflexion and 5.4 N•m for plantarflexion). The significant differences indicate that the ankle device can be successfully used as a medical diagnosis tool.



a)



b)

Figure 7.11. A patient's pitch (in degrees) and torque (in ft•lbs) vs. time (in seconds): a) for the healthy ankle and b) for the injured ankle (Girone et al., 2000).

7.3 Conclusions and Future work

Clinical trials of the telerehabilitation system presented in the previous chapters were conducted at the Stanford University Medical School, with remote monitoring from Rutgers University CAIP Center. A protocol for post carpal tunnel surgery was developed at Stanford. Several patients followed this protocol under therapist supervision. Clinical data collected so far indicates that patients level of effort and grasping strength increased after using our telerehabilitation system. A larger clinical study is needed to assess the efficacy of the VR-based hand rehabilitation vs. the clinical therapy.

Another set of clinical trials tested efficacy of ankle force feedback device for diagnostic purposes. Proof-of-concept patient trials conducted at UMDNJ concluded that “Rutgers Ankle” can be used for rehabilitation of patients with hyper and hypomobile ankles. Future studies will be required, however, to establish the systems’ measurements’ reliability, the predictive validity of training on the device, and its ability to improve patient’s function.

Chapter 8

Shared VEs for Telerehabilitation

A “store and forward” system as described in the previous chapter cannot implement medical services involving patient therapist direct interaction. Such services require real-time (at frame rate speed or better) communication between the sites of the medical simulation. The Shared Virtual Environment architecture provides a foundation for developing real-time telerehabilitation simulations. Shared (or networked) Virtual Environments are software systems in which multiple users located in different geographical locations interact with each other in real-time (Singhal and Zyda, 1999). Networked VEs allow multiple users to share information, manipulate objects in the Virtual Environment and even interact physically with each other in case the VR interface can feedback forces to the users. A two-user (therapist and patient) Shared Virtual Environment can be used in telerehabilitation applications. The following sections describe the implementation of such VE architecture.

8.1 The Shared VE Telerehabilitation System

The real-time telerehabilitation system we developed is a two-user shared Virtual Environment with networked force feedback. Each site has a telerehabilitation workstation as described in chapter three, with a videocamera and an RMII force feedback glove. Each user can control a virtual hand and interact hapticly with virtual objects. Simulated physical interactions between therapist and patient are implemented using hand force feedback. This is achieved by replicating to the remote user the forces felt by the local one. Force data read from a local haptic glove are sent over the network and displayed on the remote glove. Networking the force data require smaller time delays than the rest of the transmitted data. Network force feedback tasks can be classified as "collaborative" or "cooperative" (Burdea, 2000). In collaborative tasks the users take turns when a controlling virtual objects. In cooperative tasks the users interact with the same virtual object simultaneously. Cooperative force feedback is more challenging to implement, as it requires smaller time delays to avoid control instabilities. The force replication example described above is a cooperative network force feedback task.

The telerehabilitation SVE is a two-peer system with a replicated database of virtual objects (Figure 8.1). Additionally, patient motion and force data can be stored in a clinical database during the VR simulation. All static content (3D objects, textures, sounds, etc.) is stored at each site. Therefore only dynamic updates (3D positions and rotation angles) are transmitted during the Virtual Reality simulation. In addition to this the data transmitted between the two sites include audio, video, forces, images, graphs and control commands. Both sites include an application controller, which implements the logic of the simulation, and a network component. The network protocol thread is responsible for packing and unpacking the data and sending it to the remote site at specified update rates. The network thread is synchronized with the main VR simulation thread. The changes in the Virtual Environment (dynamic position updates, textures) are applied in the next simulation loop after being processed by the network thread. This automatically introduces a time delay of at most one graphic frame (~33ms). Similarly the forces displayed by the haptic thread are delayed with one haptic frame (~5ms) in addition to network delays. The telerehabilitation application built upon this SVE architecture is described in the next section.

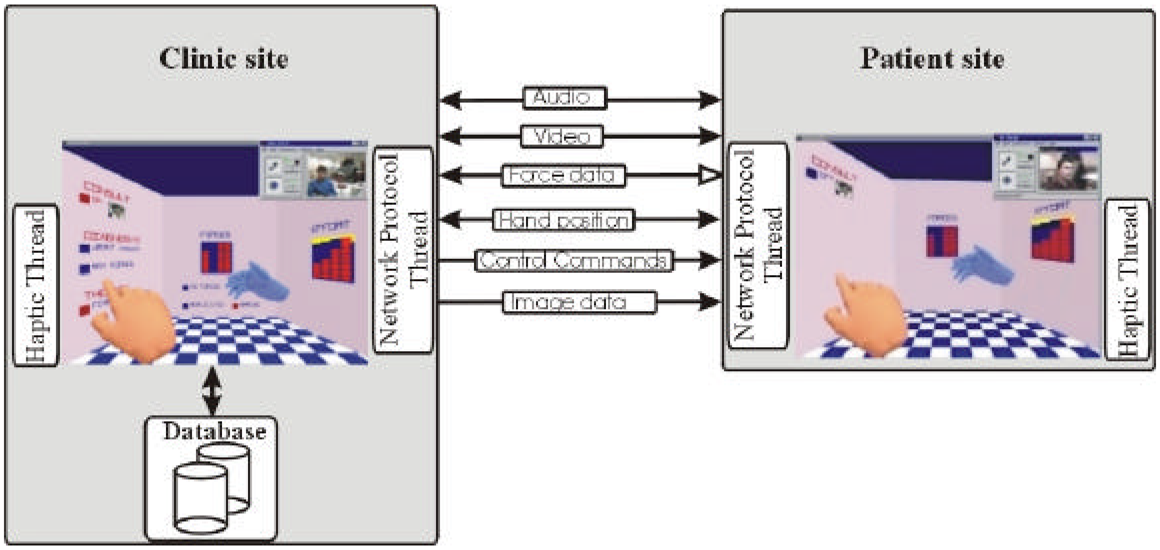


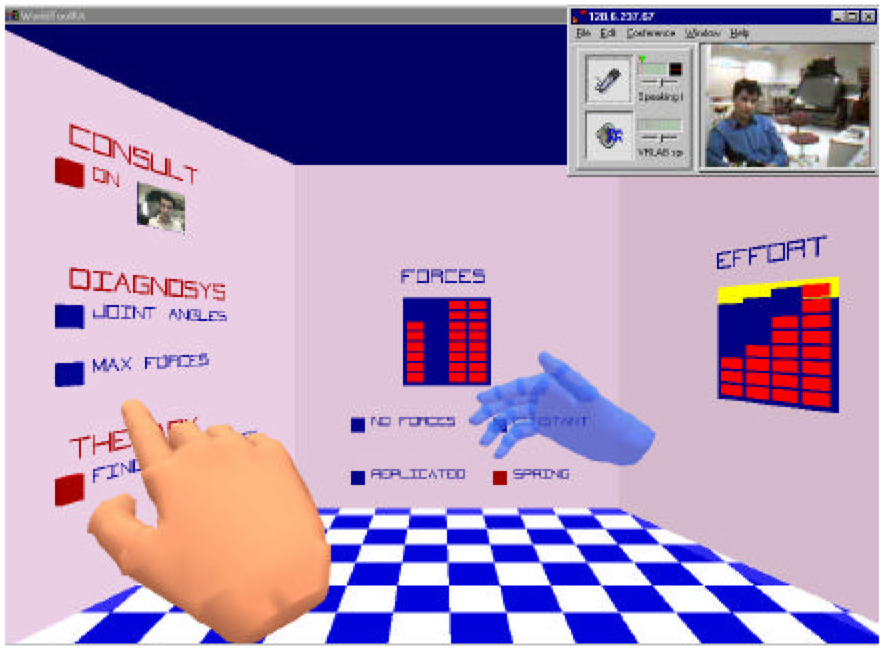
Figure 8.1: Shared Virtual Environment for Telerehabilitation

8.1.1 The Shared Virtual Rehabilitation Room (SVR^2)

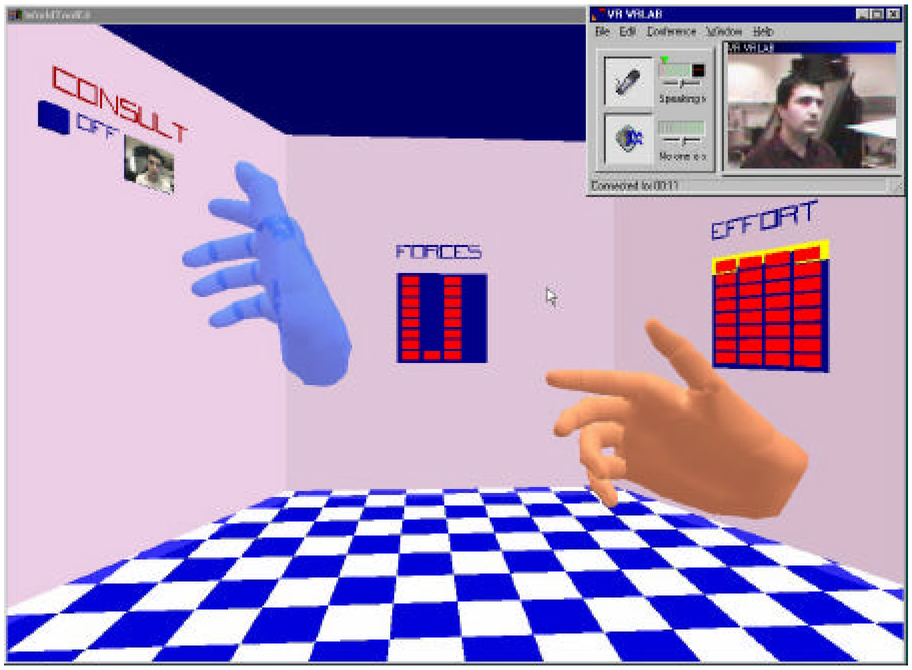
The SVR^2 telerehabilitation application is built upon a Virtual Rehabilitation Room concept described previously. The shared Virtual Environment is a *VR^2* configured as a master control interface at the therapist site and a slave system at the patient site (Figure 8.2-a, b). The environment contains virtual displays, user hands (local and remote) and control panels. At each site, the remote hand is displayed in transparent blue (50%

transparency), so that it doesn't obstruct the local user view (e.g. it doesn't obstruct the view of the 3D buttons). In order to navigate and control the virtual environment the user needs to see through objects controlled from the remote site. The transparency is also an intuitive way to identify remote objects due to their "ghostly" appearance.

The therapist interface (master system) contains several virtual panels, which allows him to control the rehabilitation process. He can start a videoconferencing session ("consult" panel), collect patient data ("diagnosis" panel) or apply therapy ("therapy" panel). The "consult" panel allows the therapist or the patient to start the videoconferencing window by pushing the on/off button on the wall. Videoconferencing uses the same CuSeeMe software described in chapter three. The "diagnosis" panel has two buttons: one for measuring finger joint angles and the other for measuring the maximum forces exerted by the patient's hand. The "therapy" panel has a single entry. When this is activated, the therapist can send a control command at the patient site to indicate the force level. Possible options are: "no forces"; "constant" - set constant level forces; "spring" - set forces proportional with finger displacement; "replicated" – send the target forces applied to therapist hand to be replicated at the patient site. In the case of replicated forces, the therapist feels proportional (or spring type, like in the virtual DigiKey) forces when he is closing the hand. Proportional force feedback provide a natural way to control the forces applied at the patient site. In addition to these panels, a "graphic board" switch can be used to display a virtual whiteboard (Figure 8.3). The whiteboard display a sequence of images, like a slide projector. The pre-loaded images represent X-ray, patient reports, graphs, drawings, etc. They are displayed as textures mapped on the whiteboard. Hand-based interaction modalities (e.g. drawing with a virtual pencil) will improve virtual whiteboard functionality in the future.



a)



b)

Figure 8.2: Shared VE for Telerehabilitation: a) Clinic site; b) Patient site.

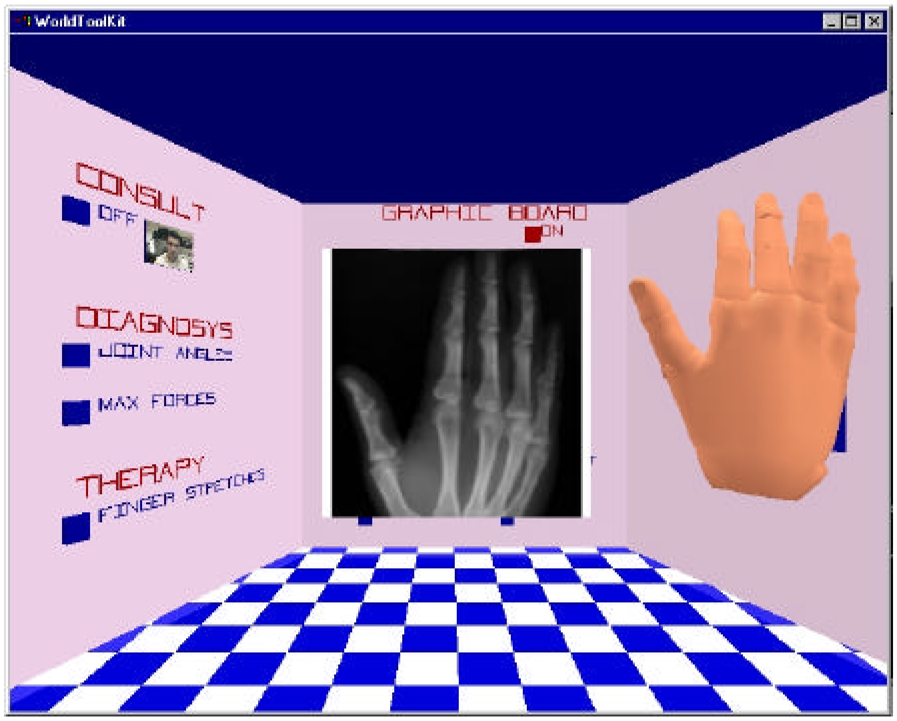


Figure 8.3: Shared VE for Telerehabilitation: Graphic Board.

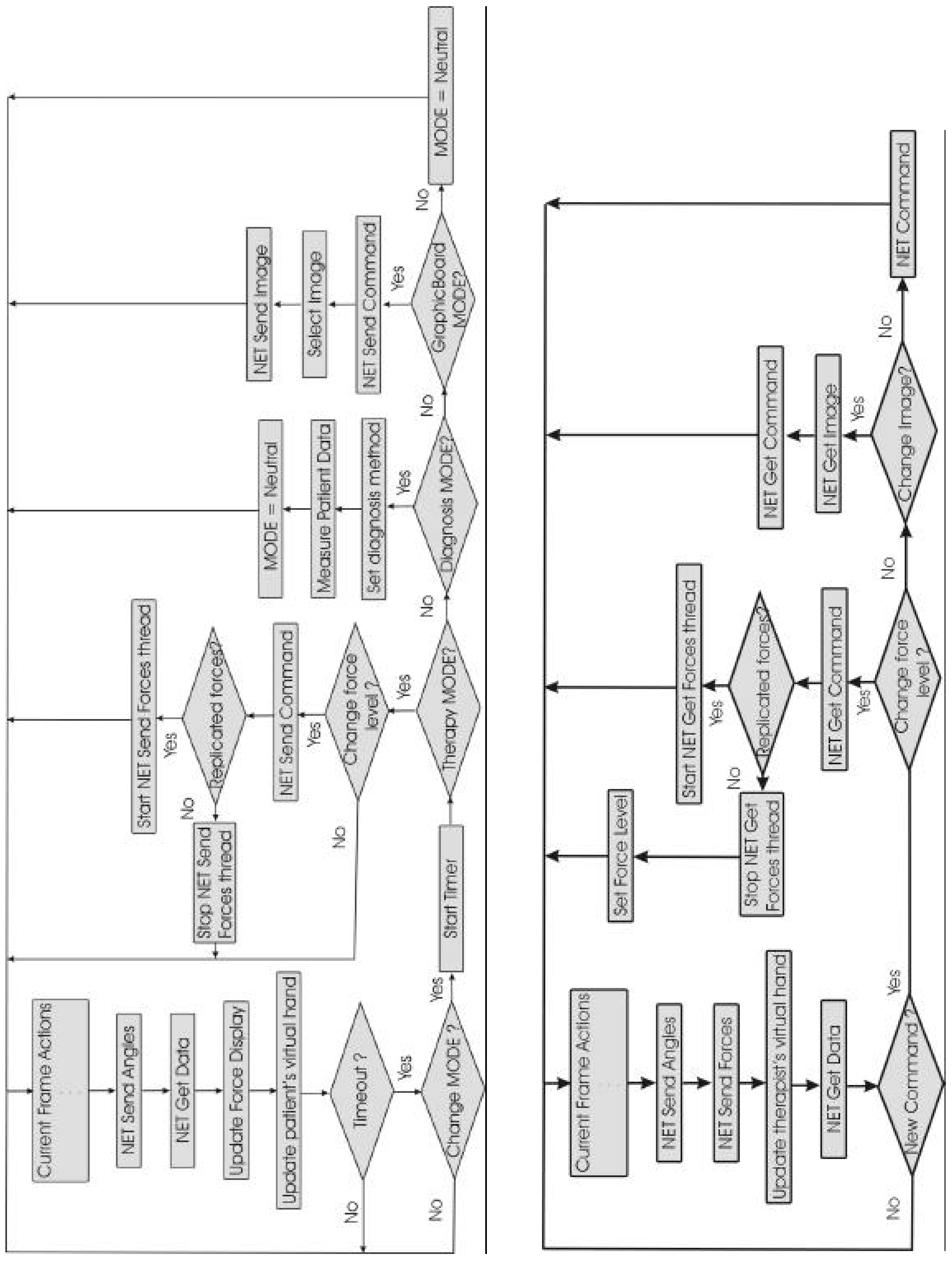
The slave system at the patient site has limited interaction modalities, its primary task being to feedback force and visual information. The forces rendered here are either the result of patient's hand interaction or commanded by the therapist from the remote site. The visual information (image textures) is also in part sent from the remote site. The only control items in the Virtual Environment are the "consult" and whiteboard switches. This allows the patient to start a videoconsultation session and to select the desired image to be displayed on the whiteboard.

In addition to control panels, both sites have two visual feedback displays to show force and effort levels The displays are used to visualize the force and effort exerted by patient's hand. For the therapist, this visual feedback closes the application control loop, while for the patient it has a motivational effect, pushing his performance to meet the rehabilitation session goals.

The application was developed using WorldToolKit graphics library. In addition to the graphics loop, separate threads run the force feedback loop and the network communication loop. The control diagram of the applications running at clinic and patient sites is shown in Figure 8.4-a, b. Synchronization of the two sites is based on a command protocol, which sets the application state in one of several modes: *diagnosis, therapy*, *graphic board* and *neutral*. Rapidly switching between application modes could destabilize the system by generating too much network traffic. In order to prevent this, a timer provides a minimum delay for switching between different control modes.

Hand positions and fingers joint angles are continuously sent over the network from each site, in order to animate the corresponding remote hand. Additionally, the forces displayed to the patient's hand are sent over the network to the clinic site. This information is used to display at both sites force and effort visual feedback. The effort level is calculated as an integral of either local forces (at patient site) or remote forces (at the clinic site). The position and patient force data (used in this context only for visual display) are sent at the graphics frame rate speed. Therefore patient force and motion data are continuously available at the therapist site and can be stored in the database when the *diagnosis* mode is enabled.

The *replicated force* therapy mode is similar to a robotics telemanipulation application. In this mode, the finger forces are sent from therapist site to the patient site in order to control the patient's force feedback. Since these forces are used as targets for force feedback control the transmission rate should match the force feedback loop update rate (about 200 updates per second for the RMII glove – see Chapter 3). Therefore when this mode is selected, the applications start an additional network thread, which runs at



a) b)

Figure 8.4: SVR^2 Session Management and Communication Diagram: a) Clinic site; b) Patient site.

the above-mentioned speed. Still, network delays could adversely affect stability of remote force control. To prevent this, the force targets are smoothed in the presence of large network delays.

In *graphic board* mode, images are sent remotely from clinic site to be displayed on the patient whiteboard. The size of this data is quite large (high quality medical images) and suffers important delays. Images are sent in TGA format (which is convenient for texture manipulation), but a compression module should be added in the future for compliance with the DICOM image transmission standard (DICOM, 2000).

8.1.2 The SVR2 Protocol

The two-peer SVE described above requires a simple communication protocol. There are only five message types: angles, positions, forces, images and commands. The packet format of the network protocol is: < *tag*, *data*, *length* >. The *tag* is used to identify the message type. Each type of message is parsed and processed differently by the network protocol thread (e.g. NET_ANGLES package contains 20 hand joint angles used to control the remote hand). Dynamic update messages (angles, position and forces) are sent via an UDP-based protocol. The command messages are used for session management (see the session management diagram in Figure 8.4). They synchronize the application modes between the two SVE sites (e.g. when the therapist site changes to *replicated force* mode, the patient site is switched to the same mode upon receiving the remote switch command). All packets have fixed size except for image messages. The image package is split at the transmitting end and reassembled at the receiver. The *length* field is therefore useful in this case, since the total length of the message varies. Table 8.1 shows the SVR^2 message types used by the protocol.

Tag	Message type	Bytes
NET_ANGLES	Hand Joint Angles	80
NET_POSITION	Hand Position	24
NET_FORCES	Finger Forces	16
NET_IMAGE	Image	Variable
NET_COMMAND	Command	1

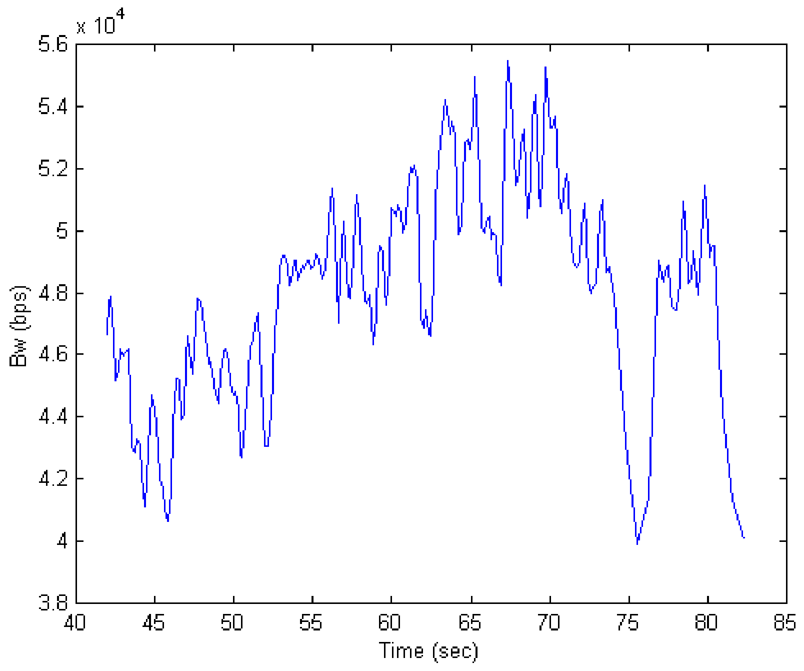
Table 8.1: SVR^2 packet types

Bandwidth requirements vary during SVE simulation. Some packets are sent continuously at graphics frame rate, others are sent at haptic frame rate only in some application modes while others are burst transmissions (image packets). Therefore the application produces a basic network load plus a variable one. The basic network load can be estimated as follows: for graphics running at 20 fps the estimated maximum bandwidth requirements are: 2*20*80*8 bps (angles) + 2*20*24*8 bps (positions) + 20*16*8 bps (patient forces for display) = 36 Kbps. When force targets are sent at 200 updates per second, this adds another 26 Kbps for a total of 62 Kbps. This is still much smaller than the hundreds of Kbps required by the videoconferencing application. The critical parameter for the force replicated case is the network round trip time (RTT). Haptic data require a maximum time delay of less than 100 ms. This requirement is easily met in a LAN setup (RTT in the order of couple of milliseconds), but can also be satisfied by Internet2 connections (as shown in chapter six, RTT for Rutgers-Stanford connection is about 80 ms).

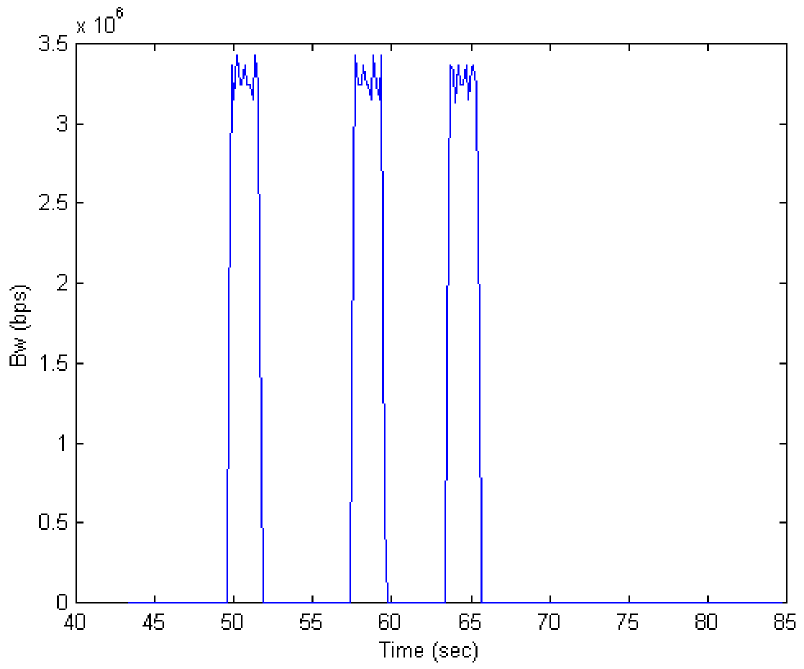
8.1.3 SVR^2 Experiments

The shared virtual environment with force feedback was designed for high-bandwidth/small-latency networks. The SVR^2 application was developed and tested on the CAIP LAN. Two PC systems with high performance graphics cards, videoconferencing capabilities and RMII haptic interfaces were used for testing the application. The two systems had different processing power, and therefore ran the graphic and network loop at different speeds. The minimum graphic refresh rate measured was 20 fps. Therefore the graphic and network update rate was limited to 20 fps at both sites. This was done to prevent the fast computer to overflow with unnecessary data the slower one. Patient force data were logged in the clinical database during the tests. Force data were sampled one time per graphic frame. Initially the therapist control system sends a *no force* command. Later spring forces are set at the patient site. Finally the therapist uses the *replicated force* mode to control forces at patient site.

To characterize SVR^2 bandwidth requirements, several network traffic sessions were recorded. As expected, the network traffic (in a LAN) produced during therapy and diagnosis modes is quite constant (Figure 8.5-a)). The application uses only 45 Kbps on average in these modes. Switching to whiteboard mode creates bursty traffic (Figure 8.5-b)). The bursts require an average of 3.2 Mbps. To this we should add the videoconferencing application traffic (on the order of hundreds of Kbps) which depends



a)



b)

Figure 8.5: SVR^2 network traffic: a) traffic produced during therapy and diagnosis modes; b) bursty traffic produced by the virtual whiteboard images.

on the quality (size, compression) of the transmitted images. A performance evaluation of the videoconferencing products can be found in (Tran et al., 1999).

8.2 SVR^2 Medical Applications

SVR^2 was designed to support real-time communication and remote physical interaction between patient and therapist. At the present time the system is the laboratory prototype stage and it was not used in pilot clinical studies. In the future the system should be used in the same pilot clinical trials involving the store-and-forward system described in chapter six. The scenario of potential telerehabilitation services using *SVR^2* is presented below:

Teletherapy: Teletherapy in *SVR^2* allows the therapist to control patient's hand movements. Two control modalities can be implemented: force control and position control. In the first case the therapist will control forces applied to the patient's hand. For the second case, the therapist controls the position of the patient fingers. The second case requires double-acting force feedback pistons at the patient site. Joint angles, which are sent from the clinic site, are used as target angles and forces are applied until the patient's hand reaches the requested configuration. The method can be used to implement a standard hand rehabilitation exercise: *finger stretching*. In this exercise the therapist is molding the patient's hand. While such detailed physical interaction cannot be implemented with the currently available haptic technology, force or position control methods described above provides a reasonable substitute.

Telediagnosis: Currently patient evaluation is based on the data analyzed remotely by the therapist. While this analysis might provide enough information about patient progress, it probably cannot completely substitute a traditional patient evaluation session. The *SVR*^{2} allows the therapist to collect data from the patient during a real-time session. The videoconference application allows the therapist to give instructions to the patient ("open hand", "close hand", "grasp", etc.) and ask the patient to execute standard tests for evaluation purposes. The therapist will measure joint angles, finger mobility, maximum force, hand range of motion, and other medical parameters.

Telemonitoring: The therapist can monitor patient hand movements, the forces and the effort applied by patient's hand. Since both virtual hands (local and remote) are displayed in the shared virtual environment, the therapist can observe unusual hand movements and incorrect routine execution. Force and effort information feedback is provided by the graphic displays mounted on the walls of SVR^2.

Teleconsultation: The therapist can discuss clinical data with the patient by requesting the whiteboard to be displayed in the *SVR*^{2}. The displayed data include X-ray of the hand, graphs with progress reports, graphical annotations and drawings of rehabilitation routines.

8.3 Conclusions

The shared Virtual Environment Architecture can be used to implement telemedicine services involving real-time patient-physician interactions. The shared sense of space and presence allow physical patient-physician interactions mediated by haptic devices. The SVR2 is a two-user SVE using hand force feedback for patient therapy. The system allows the therapist to apply remote physical therapy and collect patient data. Patient data are logged in a clinical database and can be analyzed using the same graphic tools developed for the store-and-forward system.

Chapter 9

Conclusions and Further Research

9.1 Conclusions

9.1.1 VR-based Telerehabilitation

VR-based telemedicine systems are currently expanding their use to home health care. The prototype PC-based telerehabilitation system developed as part of this thesis is an example of such telemedicine applications. The system uses Virtual Reality and force feedback interfaces for orthopedic rehabilitation. A library of Virtual Reality exercises was modeled after standard rehabilitation routines. This simulation library contains both physical therapy and functional rehabilitation routines. The RM-II and “Rutgers Ankle” force feedback devices were used to apply forces on the patient's body. Force data applied to patient’s fingers are also collected locally at the patient site. Data collected during the exercises are later forwarded and stored remotely at the server site (clinic). Here the therapist can analyze it, evaluate the patient's progress and modify VR exercise parameters or rehabilitation goals over the network. Remote consultation is supported using a videoconferencing system.

A second telerehabilitation prototype we developed uses a Shared Virtual Environment to enable real-time patient-therapist interaction. The physical interactions between the therapist and patient are implemented using hand force feedback. The therapist can control with the force feedback glove the forces applied to the patient's fingers. In addition patient's force and motion data can be recorded in real-time during the VR simulation. The communication protocol transmits audio, video, images, scene graph information, force, and control commands between the two simulation sites. The system can be used to implement telemonitoring, teletherapy, teleconsultation and telediagnosys applications.

The force feedback simulations represent a novel approach to rehabilitation. Patients interact with a force feedback glove, or a Stewart platform robot, exercising their muscles. The forces they feel are controlled by a host PC running the interactive VE simulation. Through the proof-of-concept patient trials we were able to receive feedback from patients and physical therapists. Their suggestions were taken into account for improving system design.

Clinical data collected from Stanford-Rutgers pilot study indicates that patients level of effort and grasping strength increased after using the telerehabilitation system. A follow-up clinical study is needed however in order to assess the efficacy of the VR-based hand rehabilitation vs. the clinical therapy. The study should compare rehabilitation performance of a group of patients using the VR-based system with the performance of a control group undergoing classical rehabilitation. An additional experiment should measure the retention factor of VR-based rehabilitation.

Currently the telerehabilitation system is being extended with new haptic devices.

Elbow and knee interfaces controlled by the same Multipurpose Haptic Control Interface hardware are currently being developed. Elbow and knee units are 1DOF systems using symmetrically mounted pneumatic actuators to oppose flexion-extension motion. VR hardware development was a major component of our telerehabilitation project. We complemented this with building haptic rendering tools, as described in the next section.

9.1.2 Haptic Rendering

Modeling deformations and haptic interactions of complex shaped objects needs more elaborate methods than previous point-based interaction models. These methods should take into account the shape of the virtual haptic tool. We developed such a rendering method and applied it for hand interactions modeling in Virtual Environments. The model uses a *haptic interaction mesh* to calculate haptic interactions and object deformations. This allows better quantification of local haptic interactions. Haptic interaction meshes were defined at each fingertip and used in the force calculation and virtual object deformation. The method is not dependent on the haptic interface hardware used. However, the mesh parameters need to be customized according to the geometry of the haptic device avatar in VE.

The virtual hand haptic rendering method was the foundation of a programming library containing the tools for developing haptic VE. The library was used to render the RM-II haptic interactions in our medical VR simulations. The haptic interaction modeling enhanced these real-time simulations that involve elastic and plastic deformations and physical modeling (dynamics, reflection law, and gravity). Currently the haptic library only supports the RM-II glove. In the future, the library will be extended for haptic modeling of new VR interfaces (elbow, knee, etc.).

9.1.3 Thesis Research Contributions

The main contributions are:

- contribution to control and communication of a hand haptic interface (RM-II);
- a haptic rendering method for virtual hand interactions;
- a haptic rendering programming interface for a virtual hand;
- haptic Virtual Environments for orthopedic rehabilitation;
- a prototype Client/Server architecture for telerehabilitation;
- a prototype shared haptic Virtual Environment for telerehabilitation.

9.2 Future Research Directions

9.2.1 Pieces in a Puzzle: Towards the Haptic Body Suit

The haptic interfaces presented in this thesis are scattered pieces of a big puzzle: the haptic suit for virtual reality simulations. A feasibility report of the haptic suit was presented in (Popescu & Burdea, 1998-a). The report concludes that a portable haptic suit generates at present insurmountable technical challenges. However, a research test-bed for a grounded full body haptic suit can easily be assembled using pneumatic (Figure 9.1) and electric actuators. Portable and non-portable force feedback devices (e.g. RMII, PHANToM, elbow, knee and ankle force feedback devices) can be integrated with commercially available kinestethic displays: gyroscope motion platform, motion bases and treadmill locomotion systems. This will allow complex simulations including varying terrain characteristics, gravity simulation, navigation through VE (flight), grasping, throwing, pushing, etc.

The haptic suit has immediate applicability to body rehabilitation. Extending the force feedback to the whole body will allow more complex orthopedic treatments. Some injuries are localized to a body part, while many others affect groups of muscles; therefore the need for an integrated force feedback solution. Functional rehabilitation will also benefit from using a haptic suit. With the haptic suit, the patient will develop strength using the proper synergies for everyday actions.

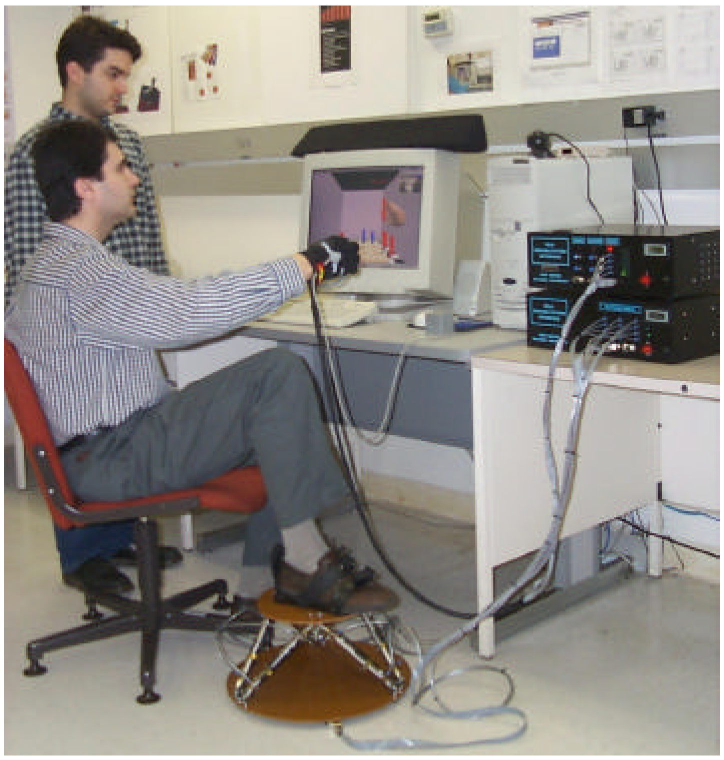


Figure 9.1: Rutgers Pneumatic-based Haptic Interfaces for Virtual Reality

9.2.2 Multiplexed Telerehabilitation

The store-and-forward system presented in chapter six can support multiple database clients, as shown in laboratory experiments. However, the Stanford-Rutgers project tested a system with only one server (Rutgers) and one client (Stanford). The implementation can be extended to allow therapists to simultaneously work with several patients using different rehabilitation devices (Figure 9.2). This was called Multiplexed Telerehabilitation (WebMT, 2000), since the medical services provided by a single therapist can be multiplexed between heterogeneous clients running different rehabilitation therapies and using different rehabilitation force feedback devices. In essence, the Multiplexed Telerehabilitation system is a Client/Server application, where patient data are shared among a pool of servers located in several clinics and the clients are located in the patient's house. Additional issues have to be addressed in order to implement this configuration: distributed database, remote consultation multiplexing (server site), patient identification (client site), and data security. Multiplexed Telerehabilitation should allow the testing of the full potential of telerehabilitation technology. Through multiplexing, a therapist can monitor the progress of several patients at a time, deciding for each of them whether to keep doing the current exercises or to step to another level. This represents a more efficient use of clinical resources while maintaining the required quality of telerehabilitation services. Additionally, since patient data are shared among several clinics, other telemedicine services can be developed (e.g. cross-evaluations of patient progress), increasing the quality of medical service received by the patient.

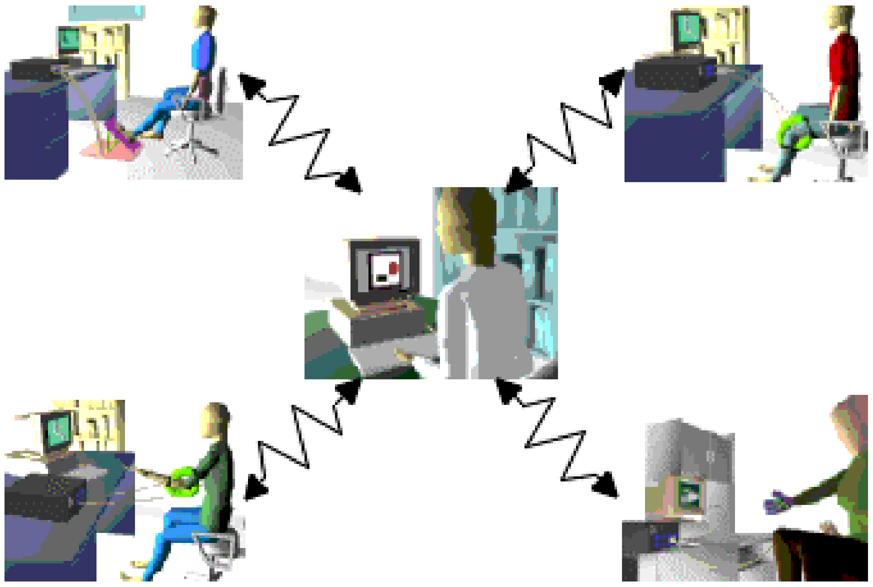


Figure 9.2: Multiplexed Telerehabilitation

The work on a Multiplexed Telerehabilitation project has began last year (WebMT, 2000). At the present stage the project focuses on developing a system with one server and multiple clients. The range of possible home rehabilitation applications using hand and ankle devices is also extending (Jack et al., 2000) (Deutsch et al., 2000). Additional rehabilitation devices (for knee and elbow) are currently under development, although not part of this thesis.

9.2.3 WEB-based Telerehabilitation Platform

The existing VR rehabilitation system consists of several software components glued together on a Windows-NT platform. The VR-based rehabilitation needs installation of additional commercial software such as a 3D graphics library and video conferencing, on the home system. Database client, forms and reports use Oracle development APIs in addition to the database server. The result of adding all these components together is a rigid, platform dependent system that is hard to manage and extend. In contrast, a Web-based platform would take advantage of available Internet technologies to create a distributed system (database, multimedia, VR exercises).

A possible implementation will use Java (Java, Java3D, JavaBeans) technology (Sun Microsystems, Inc., 2000). Java applets will implement rehabilitation exercise programs, help and training modules, will monitor (and log) progress and will provide feedback. Security, authentication, naming, dynamic discovery, database access (JDBC), and user-interface services will be provided by Java technology. This service integrated with intelligent agent support and conventional multimedia and video-conferencing will provide a true distributed virtual environment for effective telerehabilitation. The components of the Web-based system architecture are presented in the Figure 9.3.

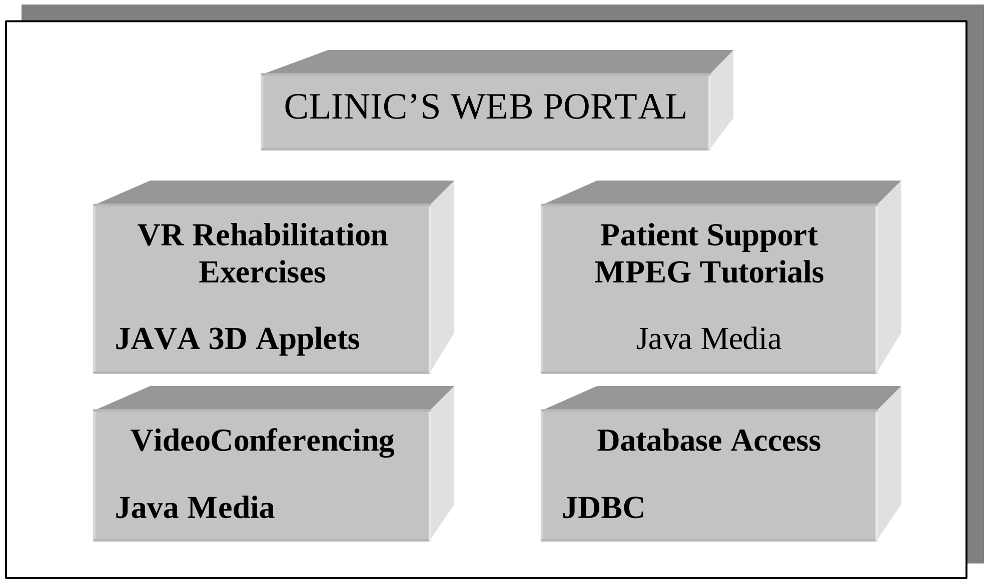


Figure 9.3: Web-based telerehabilitation platform

The patient will use a browser to access telerehabilitation services through clinic's Web portal. The portal will contain 3D rehabilitation exercises, database access and videoconferencing services. The rehabilitation applets will be downloaded and played using conventional web-browsers. The therapist will use the same portal to retrieve, interpret and evaluate patient data, and to customize patient therapy.

The intelligent agents in the proposed Web-based telerehabilitation architecture are primarily responsible for security of rehabilitation routines, patient monitoring and identification, and access to shared system resources. These agents will contain rules and heuristics and will be empowered with decision-making capabilities based on inputs from the user, camera, and rehabilitation applets. The agents will check user identity using video camera or biometrics devices, and will identify the user and his/her capabilities. This information will be used to guide and regulate choices, e.g. the type and level of exercises chosen or the information accessed. Additionally the agents will monitor and quantify levels of effort and discomfort while the user performs an exercise.

Developing Web-based, large-scale systems is an immediate goal for advancing telerehabilitation technology. In distant future, many other technical challenges await to be solved in order to demonstrate the effectiveness of VR-based telemedicine. We believe that advances in VR interface and networking technologies will eventually lead to their extensive use in home health care.

Appendix A

The Rutgers Haptic Library

The programming interface of the Rutgers Haptic Library contains the following functions:

Scene Management Functions	
rmHapticInit	initializes the haptic rendering for the RMII interface; the function should be called before entering the graphic simulation loop; before calling this function, the RMII driver need to be started and the hand object should be created.
rmHapticCollision	checks the collision between the hand node and a list of haptic nodes; collision is checked using the *rmFingerIntersect* function; in case of collision it calls *rmHapticRendering* for the colliding scene node.
rmHapticRendering	implements the haptic rendering algorithm for the RMII haptic system; it contains calls to the main haptic rendering routines: *rmHIPCollision*, *rmPolyCollision*, *rmSearchContactPoly*, *rmCalculateForce*, *rmHIMRendering*.

Rendering Functions	
rmCalculateForce	calculates a 3D force vector using the HIP method.
rmHIMRendering	implements Haptic Interface Mesh rendering method.
rmForceShading	calculates the “smoothed” normal of a polygon.
rmForceDisplay	maps the calculated vector forces to the RMII glove configuration; the forces are displayed on the RMII pistons.
Collision Functions	
rmHIPCollision	checks if the HIP is inside the colliding node.
rmPolyCollision	checks for collision between the HIP path and the current node; the function returns the contact polygon or NULL if there is no intersection.
rmSearchContactPoly	performs a search and returns the polygon closest to the current HIP point.
rmFingerIntersect	checks for the intersection between the hand fingertips and a virtual object passed as argument; each bounding box of the hand fingertips are checked against the bounding box of the virtual object.
rmHandIntersect	checks for the intersection between the hand and a virtual object passed as argument; each bounding box of the hand nodes (palm, finger segments) is checked against the bounding box of the virtual object.

Internal Functions	
rmGenerateHIP	generates an HIP point and attaches it to a fingertip.
rmGenerateHIM	generates all HIM points and attaches them to the hand fingertips.
rmGenerateHIPline	generates the HIP line for the corresponding HIP point.
rmUpdateHIPline	updates the HIP lines according to the new position of the corresponding HIP points; this function also calculates the speed of the corresponding fingertip, which is used internally for force display.
rmUpdateHIP	updates the position of a HIP according to the data received from the RMII driver.
rmUpdateHIM	updates the position of all HIM points according to the data received from the RMII driver.
rmSetHapticMode	sets the haptic rendering style; if the SHADING_ON is TRUE, the polygon normal is smoothed before force calculation, otherwise the defined polygon normal is used.
rmSetHapticProperties	sets the *spring* and *damping* parameters of the force model.
rmShowHIPTrace	turns on and off the *HIP line* visibility.
rmGetHIPPos	returns the position of the HIP point passed as argument.
rmShowHIP	turns on the visibility of HIP points; small red spheres centered at HIP point position are showed when this function is called; the spheres are attached to the hand fingertips.

rmShowHIM	turns on and off the visibility of all HIM points; the points are shown as small spheres (red for the center, blue for the other points in the mesh) centered at HIM point positions; the spheres are attached to the hand fingertips.

Table A.1: The Rutgers Haptic Library

Figure A.1 presents a basic example of using the Rutgers Haptic Library with WorldToolKit (Sense 8, 1997) graphics library. The example uses two additional classes: the *RMII* class (interface driver), and the *RMII_Hand* class (virtual hand). More details on using the Rutgers Haptic Library can be found in (Popescu, 1999).

```
/* Initialize the universe */
        set the lights, set the viewpoint
/* Setup sensors */
        Setup mouse sensor
        Setup insidetrak sensor
        RMII driver setup: new RMII class object
/* Create the universe scene */
        Create virtual objects
        Create the virtual hand: new RMII_Hand object
        Set hand constraints
        Start hand tracking
        Initialize haptic rendering: rmHapticInit(…)
```

```
/* Start the haptic rendering  thread */

        Create the haptic thread
                 ___________
                |           |
                |       Update finger positions
                |
                |       Check hand collisions: rmHapticCollision(…)
                |
                |       If (collision) execute haptic rendering: rmHapticRendering(…)
                |___________|


/* Start the graphic rendering loop */
 ___________
|           |
|       Handle keyboard input
|
|       Execute object behaviors
|___________|

/* Exit the simulation */

Close the haptic thread

Call RMII_Hand and RMII class destructors

Delete universe
```

Figure A.1: Using WorldToolKit with Rutgers Haptic Library

References

3D Café, 2000. [Online]. Available: http://www.3dcafe.org/ [2000, October 15].

ATSP. 1999. Report on U.S. Telemedicine Activity. 128 pp. [Online]. Available: http://www.atsp.org/ [2000, October 15].

Airpot Co. 2000. [Online]. Available: http://www.airpot.com/ [2000, October 15].

Autodesk Inc. 1994. AutoCAD User's Manual, Sausalito, CA.

Basdogan, C., Ho, C., Srinivasan M.A. 1997. " A Ray-Based Haptic Rendering Technique for Displaying Shape and Texture of 3D rigid objects in Virtual Environments," *Winter Annual Meeting of ASME'97*, Nov. 15-21, Dallas, TX, DSC-Vol. 61, pp. 77-84.

Bergamasco, M., 1993. "The GLAD-IN-ART Project", *Virtual Reality. Anwendungen und Trends*, in Forschung und Praxis, IPA/IAO Forum, Springer-Verlag, Berlin, pp. 251-258.

Bouzit M. 1996. "Design, Implementation and Testing of a Data Glove with Force Feedback for Virtual and Real Objects Telemanipulation," PhD. Thesis, Paris, France.

Bouzit M., G. Burdea, V. Popescu, R. Boian. 2000. "The Rutgers Master II-ND Force Feedback Glove", submitted to *IEEE Transactions on Mechatronics*, July.

Bowman, D. 1999. *Interaction Techniques for Common Tasks in Immersive Virtual Environments.* PhD Thesis, Georgia Tech University.

Brandt, E., & A. Pope (Eds). 1997. *Enabling America - Assessing the Role of Rehabilitation Science and Engineering* , National Academy Press, Washington D.C..

Bro-Nielsen, M. 1997. “Simulation Techniques for minimally invasive surgery.” Minimally Invasive Therapy & Allied Technology. Vol. 6, pp. 106-110.

Burdea, G., Langrana, N., Roskos, E., Silver D., & Zhuang, J. 1992. “A portable dextrous master with force feedback,” *Presence-Teleoperators and Virtual Environments*, MIT Press, 1(1), pp. 18-28.

Burdea G. 1996. *Force and Touch Feedback for Virtual Reality*, John Wiley & Sons, New York, NY.

Burdea, G., S. Deshpande, V. Popescu, N. Langrana, D. Gomez, D. DiPaolo & M. Kanter. 1997-a. “Computerized Hand Diagnostic/Rehabilitation System Using a Force Feedback Glove,” *Proceedings of Medicine Meets Virtual Reality 5 Conference*, pp. 141-150.

Burdea, G., Sonal Deshpande, Biao Liu, N. Langrana & Daniel Gomez. 1997-b. “A Virtual Reality-based System for Hand Diagnosis and Rehabilitation,” *Presence - Teleoperators and Virtual Environments*, 6 (2), pp. 229-240.

Burdea, G. 1998-a. “Medical Virtual Reality.” *VRAIS 98 Tutorial*. Atlanta, Georgia, March, pp. 1-44.

Burdea, G. 1998-b. *Multiplexed Orthopedic Telerehabilitation using Virtual Reality*, Proposal to NIDRR, CFDA 84.133G, September.

Burdea, G., Popescu, V., Hentz, V., Colbert, K. 2000. “Virtual Reality-Based Orthopedic Telerehabilitation”, *IEEE Transactions on Rehabilitation Engineering*, Vol. 8, No. 3, pp. 430-432.

Burdea, G. 2000. “Keynote Address: The Challenge of Large Volume Haptics.” *Proceedings of VRIC 2000*, pp. 101-111. May.

Cai, Z., Dill, J., Payandeh, S. 1999. “Haptic Rendering: Practical Modeling and Collision Detection,” *DSC – Vol. 67, Proceedings of the ASME*, pp. 81-88.

Chou, C.P., B. Hannaford. 1996. “Measurement and Modeling of McKibben Pneumatic Artificial Muscles,” *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 90-102, February.

Cok, K., Commike, A., Kuehne, B., True T. 1999. “Developing Efficient Graphics Software”, SIGGRAPH ‘99 Course Notes, pp. 1-145.

DataBeam Co. 1998. *A Primer on the H.323 Series Standard*, DataBeam Co. White Paper. pp. 1-16. [Online]. Available: http://www.databeam.com/ [2000, October 15].

Deutsch, J., Latonio, J., Burdea, G., Boian, R. 2000. “Post-Stroke Rehabilitation with the Rutgers Ankle System – A case study,” Submitted to *Presence* Journal, September. pp. 1-27.

DICOM. 2000. [Online]. Available: http://www.rsna.org/REG/practiceres/dicom.html [2000, October 15].

Foley, J., van Dam, A., Feiner, S., & Hughes, J. 1990. *Computer Graphics: Principles and Practice Second Edition.* Addison-Wesley Publishing Company, Inc.

Fox, S. 1991. “EVAL-Revolutionizing Hand Exams,” *ADVANCE for Occupational Therapists*, 7(3), pp. 1-7.

Girone, M. J., G. C. Burdea, M. Bouzit. 1999. “The Rutgers Ankle Orthopedic Rehabilitation Interface,” In *Proc. of the ASME, Dynamic Systems and Control Division Vol. 67*, International Mechanical Engineering Congress and Exposition.

Girone, M. J., G. C. Burdea, M. Bouzit, V. Popescu, and J. E. Deutsch. 2000. “Othopedic Rehabilitation Using the ‘Rutgers Ankle’ Interface,” In *The Proc. of Medicine Meets Virtual Reality 2000*.

Girone M., Burdea, G., M. Bouzit, V. Popescu, and J. Deutsch. 2001. “A Stewart Platform-based System for Ankle Telerehabilitation,” invited article, *Special Issue on Personal Robotics, Autonomous Robots*, Vol. 10, Kluwer, pp. 200-212.

Gomez D., G. Burdea and N. Langrana. 1995. “Integration of the Rutgers Master II in a Virtual Reality Simulation,” *Proceedings of IEEE Virtual Reality Annual International Symposium* (VRAIS), pp. 198-202.

Greenleaf Medical Systems. 1997. Business Overview, Palo Alto CA. [Online]. Available: http://www.greenleafmed.com/Products/pointofcare.html. [2000, October 15].

Grimes, G., H. Dubois, and S. Grimes, W. Greenleaf, D. Cunningham. 2000. “Telerehabilitation Services Using Web-based Telecommunication.” *Proceedings of MMVR* 2000 IOS Press, pp. 113-118.

HL7. 2000. [Online]. Available: http://www.hl7.org/ [2000, October 15].

Ho, C., Basdogan C., Srinivasan, M., 1997. “Haptic Rendering: Point- and Ray-Based Interactions,” *Proceedings of the Second PHANToM Users Group Workshop*, Dedham, MA, Oct. 20-21.

Internet2. 2000. [Online]. Available: http://www.internet2.edu. [2000, October 15].

ITU. 1998-a. *ITU-T Recommendation H.323: Packet-based multimedia communication systems*. pp. 112. [Online]. Available: http://www.itu.itn/ [2000, October 15].

ITU. 1998-b. *ITU-T Recommendation H.263: Video coding for low bit rate communication*, pp. 153. [Online]. Available: http://www.itu.itn/ [2000, October 15].

Jack, D., Rares Boian, A. Merians, M. Tremaine, G. Burdea, S. Adamovich, M. Recce, & H. Poizner. 2000. "Virtual Reality-Based Stroke Rehabilitation," Submitted to *IEEE Transaction in Rehabilitation.*

Kim, D., J. E. Cabral Jr., and Y. Kim. 1995. "Networking Requirements and the Role of Multimedia Systems in Telemedicine," *University of Washington Image Computing Systems Laboratory*; [Online]. Available: http://icsl.ee.washington.edu:80/projects/gsp9/spie1095/ [2000, October 15].

Krebs, H., N. Hogan, M. Aisen, & B. Volpe. 1996. "Application of Robotics and Automation Technology in Neuro-Rehabilitation," *Japan-USA Symposium on Flexible Automation*, ASME, New York, Vol. 1, pp. 269-275.

Lin, Q. Che, C.-W., Yuk, D.-S. & Flanagan, J. 1996. "Robust Distant Talking Speech Recognition," *Proceedings of ICASSP'96*, Atlanta, GA, pp. 21-24.

Luecke G., Y. Chai, J. Winkler, & J. Edwards. 1996. "An Exoskeleton Manipulator for Application of Electro-Magnetic Virtual Forces," *Proceedings of ASME WAM* , DSC-Vol 58, Atlanta, GA, pp. 489-494.

Massie T., Salisbury, J., 1994, "The PHANTOM haptic interface: a device for probing virtual objects," *Proceedings of the ASME Dynamic Systems and Control Division*, DSC-Vol. 55-1, Chicago, IL, pp. 295-301.

Marsh, A. 1999. *Virtual Medical Worlds*, Euromed Project Report, [Online]. Available: http://narcisus.esd.ece.ntua.gr/~www/final/report.html [1999, October 15].

Martinez, R., Chimiak, W., Kim, J., and Alsafadi, Y. 1995. "The Rural and Global Medical Informatics Consortium and Network for Radiology Services." Special Issue on Virtual Reality for Medicine, *Journal of Computers in Biology and Medicine*, Vol. 25, No. 2, pp. 85-106.

Medl, A., I. Marsic, V. Popescu, M. Andre, C. Kulikowski and J. Flanagan, 1998. "Multimodal Interface for Collaborative Mission Planning," *5th ISPE INTERNATIONAL CONFERENCE ON CONCURRENT ENGINEERING*, Tokyo Metropolitan Institute of Technology, Tokyo, Japan, July.

McNeely, W., K. Puterbaugh and J. Troy, 1999. "Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling," *Siggraph '99 Proceedings*, ACM, pp. 401-408.

Middleton, R. and F. van Millingen. 1999. “State of the Art Review of Presentation Technologies Report”. *JTAP Report*. [Online]. Available: http://www.jtap.ac.uk/reports/htm/jtap-056.html [2000, October 15].

Microsoft Co. 1998. *Whisper Speech Recognizer*. [Online]. Available: http://research.microsoft.com/msrinfo/demodwnf.htm [2000, October 15].

Microsoft Corp. 1999. *NetMeeting 3 Resource Kit*. [Online]. Available: http://www.microsoft.com/windows/NetMeeting/Corp/ResKit [2000, October 15].

National Center for Dissemination of Disability Research, 1998. “The National Institute on Disability and Rehabilitation Research (NIDRR)”. [Online]. Available: http://www.ncddr.org/ [2000, October 15].

National Center for Medical Rehabilitation Research. 1993. *Research Plan for the National Center for Medical rehabilitation Research,* NIH Publication No. 93-3509.

NIST 2000. [Online]. Available: http://www.nist.gov/itl/div894/ovrt/projects/vrml/h-anim/jointInfo.html [2000, October 15].

NGI 2000. [Online]. Available: http://www.ngi.gov. [2000, October 15].

NLANR. 2000. “Iperf version 1.1.1.” [Online]. Available: http://dast.nlanr.net/Projects/Iperf/ [2000, October 15].

North Coast Medical Inc. 1994. *Digi-Key*, San Jose, CA.

Oracle Co.1995. *Oracle User’s Manual*, Redwood City, CA.

North, M., North, S., Coble J. 2000. “Virtual Reality Therapy – An effective treatment for psychological disorders,” *Handbook of Virtual Environment Technology*, chapter 57, K. Stanney ed., Lawrence Erlbaum Associates, Inc., (to appear).

Patounakis, G., Bouzit, M. & Burdea, G. 1998. “Study of the Electromechanical Bandwidth of the Rutgers Master,” *Technical Report CAIP-TR-225*, Rutgers University, May 22.

Peifer, J., A. Hooper, & B. Sudduth. 1998. “ A Patient-Centric Approach to Telemedicine Database Development,” *Proceedings of Medicine Meets Virtual Reality*, IOS Press, pp. 67-73.

Polhemus. 1993. *Fastrak User's Manual* , Colchester, VT.

Popescu V. and G. Burdea. 1998-a. “Research on full body modeling and force feedback,” *CAIP-TR-223*, Rutgers University, April.

Popescu V. and G. Burdea. 1998-b. “Dextrous Haptic interface for Jack™,” *Proceedings of Seventh Annual Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, ASME WAM, Anaheim, CA., November.

Popescu, V., G. Burdea, Mourad Bouzit, M. Girone, V. Hentz MD. 1999-a. “PC-based Tele-Rehabilitation System with Force Feedback,” *Proceedings of MMVR7*, San Francisco, CA, January.

Popescu, V., G. Burdea, M. Bouzit. 1999-b. “Virtual Reality Modeling for a Haptic Glove,” *Proceedings of Computer Animation '99*, Geneva, May, pp. 195-200.

Popescu, V. G. 1999. “The Rutgers Master Haptic Library,” Human Machine Interface Laboratory Internal Report, Rutgers University, October, pp. 1-21.

Popescu, V., G. Burdea, and H. Trefftz. 2000-a. “Multimodal Interaction Modeling,” *Handbook of Virtual Environment Technology*, chapter 21, K. Stanney ed., Lawrence Erlbaum Associates, Inc., (to appear).

Popescu, V., G. Burdea, M. Bouzit and V. Hentz MD. 2000-b. “ A Virtual Reality-based Telerehabilitation system with Force Feedback,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 4, No. 1, pp. 45-51.

Popescu V. 2000. “The Rutgers Haptic Library,” invited presentation at the *IEEE VR 2000 Workshop: “Haptics in Virtual Environments”*.

ReachIn Technologies. 2000. *The Magma software development environment*. [Online]. Available: http://www.reachin.se/DevelopmentPlatform/Magma.htm. [2000, October 15]

Reinhardt, A., 1998, “ What Could Whip the World Wide Wait,” Business Week, February 16, pp. 83.

Rizzo, A., Buckwalter, G., van der Zaag, C. 2000. “Virtual Environment Applications in Clinical Neuropsychology,” *Handbook of Virtual Environment Technology*, chapter 58, K. Stanney ed., Lawrence Erlbaum Associates, Inc., (to appear).

Rosen, M., 1999. “Telerehabilitation,” *NeuroRehabilitation* 12, pp. 11-26.

Rosenberg, L. and D. Stredney, 1996. “ A Haptic interface for Virtual Simulation and Endoscopic Surgery,” *Health Care in Information Age*, Sieburg, Weghorst and Morgan (Eds.), IOS Press, pp. 371-387.

Rovetta, A., Lorini, F. & Canina, M. 1998. “ A New Project for Rehabilitation and Psychomotor Disease Analysis with Virtual Reality Support.” In *Proceedings of Medicine Meets Virtual Reality 6 Conference*, IOS Press, Amsterdam, pp. 180-185.

Singhal, S., M. Zyda. 1999. *Networked Virtual Environments*, New York, ACM Press, Addison-Wesley.

Satava, R., Jones, S. 2000. " Medical Applications of Virtual Reality," *Handbook of Virtual Environment Technology*, chapter 55, K. Stanney ed., Lawrence Erlbaum Associates, Inc., (to appear).

SensAble Technologies, 1998. *GHOST SDK Programmer's Guide*, Cambridge MA.

Sense8 Co. 1997. *WorldToolKit User's Manual, Release 7*, Sausalito, CA.

Srinivasan, M., Basdogan, C., 1997. "Haptics in Virtual Environments: Taxonomy, Research Status, and Challenges," *Computer & Graphics*, Vol. 21, No 4., pp 393-404.

Sun Microsystems, Inc. 2000. *The Source for Java Technology*. [Online]. Available: http://www.javasoft.com/. [2000, October 15].

Takeda, T., & Tsutsul, Y. 1993. "Development of a Virtual Training Environment," *Advances in Robotics, Mechatronics, and Haptic Interfaces*, ASME DSC-Vol 49. pp. 1-10.

Tran, B., D. Krainak, and J. Winters. 1999. "Performance evaluation of commercial POTS-based videoconferencing systems." *Technical Report HCTR11-v1.0. HomeCare & TeleRehab Technology Center. CUA.*

Trepagnier, C. 1999. "Virtual Environments for the Investigation and Rehabilitation of Cognitive and Perceptual Impairments." *NeuroRehabilitation* 12. pp. 63-72.

Trepagnier, C., M. Rosen, C. Lathan. 1999. "Telerehabilitation and Virtual Reality Technology for Rehabilitation: Preliminary Results." *CSUN's Conference Proceedings*. [Online]. Available: http://www.dinf.org/csun_99/session0241.html [2000, October 15].

Turner M., Gomez D., Tremblay M., Cutkovsky M., 1998, "Preliminary Tests of an Arm-Grounded Haptic Feedback Device in Telemanipulation," *Winter Annual Meeting of ASME'98*, Nov. 15-21, Dallas, TX, DSC-Vol. 64, pp. 145-149.

vBNS, 2000. [Online]. Available: http://www.vnbs.net [2000, October 15].

Viewpoint. 1999. *Catalog of 3D Models*, Orem, UT.

Ward, F. & Bullinger, M.. 1998. *Joint Monitor*, US Patent 5754121, May 19.

WebATA, 2000, American Telemedicine Association. [Online]. Available: http://www.atmeda.org/index.html [2000, October 15].

WebEHS, 2000. [Online]. Available: http://dm3host.com/websites2/ehs/charttrad.html [2000, October 15].

WebLafayette, 2000. [Online]. Available: http://www.licmef.com/assessme1.htm [2000, October 15].

WebMT, 2000. [Online]. Available: http://www.caip.rutgers.edu/vrlab/multi_telerehab_nojs.html [2000, October 15].

WebRERC, 2000. [Online]. Available: http://www.hctr.be.cua.edu/RERC/ [2000, October 15].

WebTelerehab, 2000. [Online]. Available: http://www.caip.rutgers.edu/vrlab/telerehab.html [2000, October 15].

WebVH, 2000. [Online]. Available: http://www.nlm.nih.gov/research/visible/visible_human.html [2000, October 15].

WebVRE, 2000. [Online]. Available: http://www.ece.arizona.edu/~cerl/VREProject.htm [2000, October 15].

Weghorst, S. & Furness, T., 1997. "Advanced Human Interfaces for Telemedicine," *HITL Technical Report R-97-33*. University of Washington, Human Interface Technology Laboratory. pp. 1-25. On-line: http://www.hitl.washington.edu/publications/r-97-33/ [2000, October 15].

WhitePine Software Inc. 1997 . *CuSeeMe User Guide*, Nashua, NH

Zilles C.,and Salisbury K., 1995, "A Constraint-based God-object Method For Haptic Display", *IEEE International Conference on Intelligent Robots and Systems '95*, Pittsburgh, PA, Vol. 3, pp. 146-151.

Vita

George V. Popescu

1988-1993	B.S./M.S. in Electrical Engineering, University "Politehnica" Bucharest.
1993	Research intern at LTIRF Grenoble, France.
1993-1996	Teaching assistant at the Applied Electronics Department, University "Politehnica" Bucharest, Romania.
1996	Research assistant at University of Colorado at Colorado Springs.
1996-2000	Research assistant at CAIP Center, Rutgers University .
1997	"Computerized Hand Diagnostic/Rehabilitation System Using a Force Feedback Glove," *Proceedings of Medicine Meets Virtual Reality 5*.
1998	"Dextrous Haptic interface for Jack(TM)," *Proceedings of Seventh Annual Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*.
1998	"Virtual Reality Training for the Diagnosis of Prostate Cancer," *Proceedings of 1998 IEEE International Symposium on Virtual Reality and Applications*.
1998	"Virtual Reality-based Training for the Diagnosis of Prostate Cancer", *IEEE Transactions on Biomedical Engineering*.
1998	"Research on full body modeling and force feedback," CAIP-TR-223, Rutgers University.
1998	"Multimodal Interface for Collaborative Mission Planning," *5th ISPE INTERNATIONAL CONFERENCE ON CONCURRENT ENGINEERING*.
1998	"Integration of Speech and Gesture for Multimodal Human-Computer Interaction," *Second International Conference on COOPERATIVE MULTIMODAL COMMUNICATION*.

1999	"Virtual Reality Modeling for a Haptic Glove," *Proceedings of Computer Animation '99*.
1999	"PC-based Tele-Rehabilitation System with Force Feedback," *Proceedings of Medicine Meets Virtual Reality 7*.
2000	"Orthopedic Rehabilitation using the 'Rutgers Ankle' Interface," *Proceedings of MMVR 2000.*
2000	Video Chair at IEEE VR 2000.
2000	"The Rutgers Haptic Library," invited presentation at the *IEEE VR 2000 Workshop: "Haptics in Virtual Environments"*.
2000	" A Virtual Reality-based Telerehabilitation system with Force Feedback", *IEEE Transactions on Information Technology in Biomedicine*.
2000	"Virtual Reality-based Orthopedic Telerehabilitation," *IEEE Transactions on Rehabilitation Engineering*.
2000	Summer Coop at IBM TJ Watson Research Center.