INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

A Virtual Environment for Training Space Station Teleoperators

Pierre Allard

B. Sci., (Collège militaire royal de Saint-Jean), 1995

Department of Electrical Engineering
McGill University
Montréal
August, 1997

A thesis submitted to the Faculty of Graduates Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

0-612-37255-3

Canada

Abstract

This thesis presents a virtual reality system developed to support the training of astronauts as operators of the Mobile Servicing System (MSS) at the Canadian Space Agency. The objective of this system called Virtual Operations and Training Environment (VOTE), is to provide 3D visualization and simulation tools used by instructors to help astronauts understand MSS remote manipulator operations conducted in the context of the International Space Station assembly. The thesis first presents both telerobotics and virtual reality fields with accompanying literature survey followed by the description of the MSS hardware, operation and ground based simulator. A description of the design and details of the current implementation is followed by an evaluation of performance results and anticipated future work.

Sommaire

Cette thèse présente un environnement de realité virtuelle développé en support à l'entrainement d'astronautes comme opérateurs du Système d'Entretien Mobile (SEM) à l'agence spatiale canadienne. Ce système, appelé *Virtual Operations and Training Environment (VOTE)* , a pour but de fournir des outils de visualisation et de simulation 3D. Ces outils seront utilisés par les instructeurs afin d'aider les astronautes à mieux comprendre les opérations du télémanipulateur du SEM dans le cadre de l'environnement de la station spatiale internationale. Cette thèse introduit premièrement les domaines de la télérobotique et de la réalité virtuelle, accompagnés d'une revue littéraire. Une description des éléments, des opérations et du simulateur du SEM est ensuite présentée. La conception et les détails de l'implémentation de VOTE sont ensuite discutés, et sont suivit d'une évaluation des performances du système ainsi que de suggestions concernant de futurs travaux.

Acknowledgments

First and foremost, I wish to thank my family. They have taught me the importance of hard work, shown confidence in my ability, always encouraged me to seek new experiences and knowledge and to pursue my dreams. Their support and understanding have been greatly felt and much appreciated throughout my graduate studies.

My sincere thanks must go to my supervisor, professor Alfred Malowany, for guiding me into the exciting exploration of the world of Virtual Reality technology, and for providing constant feedback, constructive ideas and fruitful discussions during my graduate studies.

I would also like to express my gratitude to Michel Vachon, technical manager of the Training Department at the Canadian Space Agency, for providing me with the opportunity to participate in the MSS astronaut training program with the design and implementation of VOTE, and use the department's top of the line facilities. His feedback and ideas on the VR Tools were much appreciated.

Special thanks go to Zaven Joukakelian, contractor for the Canadian Space Agency, who was a patient and optimistic partner throughout the implementation of VOTE. Sincere thanks go to Elaine Greenberg and Michelle Hoffman for having reviewed parts of this thesis.

Working with the MSS Training Team was a great experience, and I wish to thank each of its members for the support, discussion and feedback they have provided me during my stay at the Canadian Space Agency.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

Since the mid-1980's, Space Shuttle missions have relied heavily on robotic operations for a variety of tasks, including satellite deployment and retrieval, docking with the MIR space station and servicing the Hubble Space Telescope in orbit. The nature of these robotic tasks has grown in complexity and has become of prime importance to mission success. With the in-orbit assembly of the International Space Station (ISS), telerobotic operations will be required on a unprecedented scale.

As part of the international team participating in this important project, Canada is contributing a key component to the ISS: the Mobile Servicing System (MSS). This robotic system will be the workhorse of the ISS assembly phase, and consist of a 7 degree of freedom (DOF) arm called the Space Station Remote Manipulator System (SSRMS), which sits on a mobile base. Whereas Space Shuttle astronauts have been able to operate a robotic manipulator while looking out the window, Space Station astronauts will have to carryout teleoperations without direct line of sight. Navigating and operating the SSRMS from the inside the ISS U.S. Lab Module using only camera views will require efficient training of the astronauts. As a result, the MSS Training group at the Canadian Space Agency (CSA) has chosen to use Virtual Reality (VR) technology to help astronauts develop 3D mental models of the Space Station, and augment the dynamic simulation conducted on the MSS Operations and Training Simulator (MOTS). This will enable the trainee to fill in the cognitive gaps resulting from the limited information conveyed by 2D imagery on the display screens.

This thesis presents the design and implementation of VR technology in the Virtual Operations and Training Environment (VOTE), which will be used in the training of astronauts in preparation for ISS telerobotic operations. In the current chapter, an overview of both telerobotics and virtual reality technology is first given, together with a discussion of previous work and examples of applications. The the benefits of using virtual reality in the training of teleoperators are presented, followed by previous research

and applications of virtual reality applied to training. Some important human factors involved in the use of virtual reality are discussed, followed by an outline of the remainder of the thesis.

1.1 Telerobotics Technology

This section presents an overview of the field of telerobotics together with the challenges encountered by teleoperators and how current research is addressing these challenges.

1.1.1 Telerobotic System Overview

Telerobotics is the science studying the remote operation of robotic systems. It encompasses the study of robotics, control, data communications, and the human-machine interface.

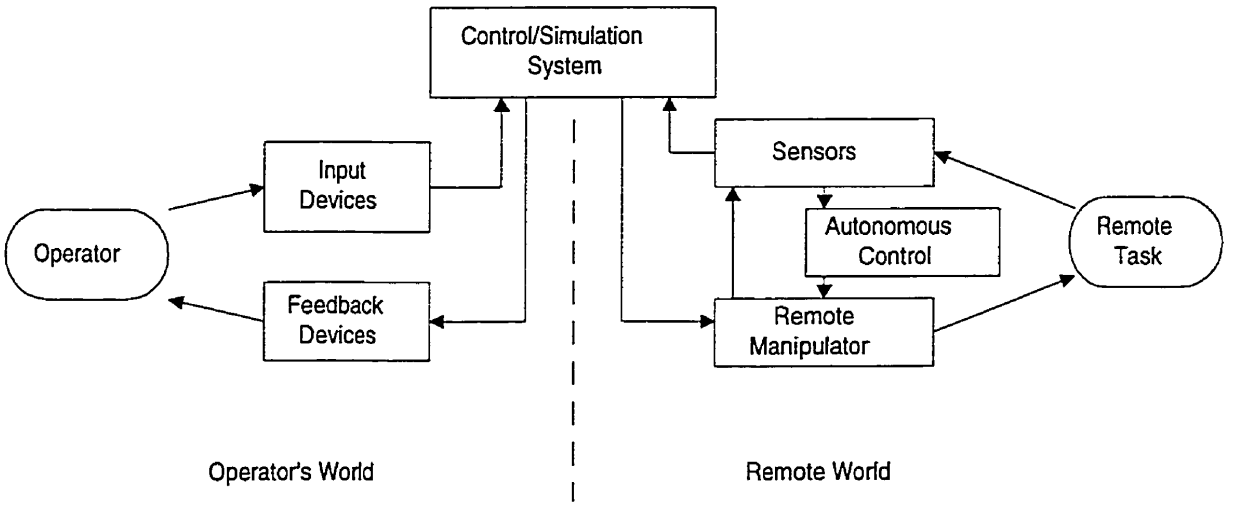


Figure 1.1 : The components of a telerobotics system.

A typical telerobotics system can be decomposed into two separate worlds, the operator's world and the remote world, which are linked by a control/simulation system.

The operator's world contains, apart from the human operator, all components needed to provide control inputs and feedback to the operator. The control inputs can be fed in using a master robotic arm [Vertut, 1986], joysticks , voice recognition [Miner, 1995] or data gloves [Cannon, 1994]. CRT screens, video overlays and force feedback on controls are examples of components used to provide the operator with information on the manipulator and environment responses to the control inputs.

The commands are fed to the control/simulation system, which interprets the controls inputs and sends the appropriate control commands to the manipulator. The control/simulation system also provides the operator with feedback on the manipulator status and environmental data (e.g. cameras views) collected on site by the manipulator sensors.

In the remote world, we find the manipulator itself, sensors and, sometimes, some level of autonomous control. The sensors are used to provide the operator and control/simulation system with feedback on the environment and the manipulator using camera, position sensors, etc. Some level of autonomous control is provided by many manipulators, such as joint limits soft stop [Spar, 1995], collision avoidance and system environmental control.

According to Milgram et al. [1995], three main factors act on telerobotics systems: robot autonomy, level of structure of the remote world, and the "modellability" [Milgram, 1995] of the remote world. The interaction of these factors can be represented as an operational space, as shown in figure 1.2.

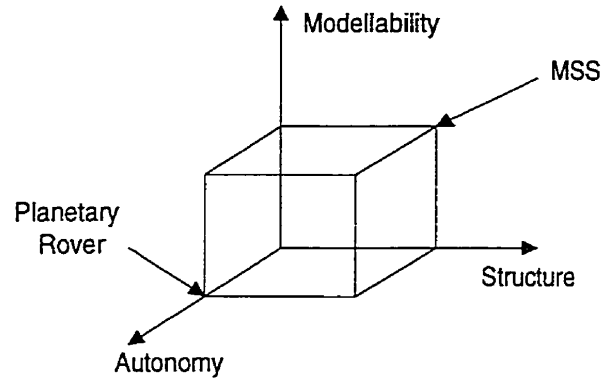


Figure 1.2: Operational space representation of telerobotics systems.

The autonomy of a telerobotic system represents the amount of decision making and control that is given to the system. As a manipulator becomes more autonomous, the role of the operator becomes one of supervisor, as control moves from manual teleoperation to high level monitoring.

The level of structure of a remote environment represents the amount of a priori knowledge of the environment's components, locations, orientation and size. As the level of structure of an environment increases, it becomes easier for the operator to interpret feedback and usually allows a higher degree of autonomy to be implemented in the telerobotic system.

Finally, the level of "modellability" of the remote world represents the extent to which the manipulator-world interaction can be predicted. When the behavior of the environment can be predicted, accurate teleoperations simulations can be performed.

As an example, one can compare the autonomous planetary rover operating in an unknown and difficult to predict world (i.e. a planet surface) to the MSS, which has low autonomy in the highly structured and well modeled ISS environment. The effects of the MSS operational space location on teleoperator tasks will be discussed in chapter two.

1.1.2 Operation challenges encountered in telerobotics.

Providing an intuitive interface for controlling a multiple degree of freedom (DOF) manipulator is one of the goals of telerobotics. Some systems use a master-slave arm configuration [Vertut, 1986] in which a replica of the remote manipulator (the master) is manipulated by the operator, while the slave arm in the remote environment follows. However, this system tends to become tiring to use for the operator for long periods of time and difficult to use in reduced space [Vertut, 1986]. Many telerobotic systems, such as the MSS, use handcontrollers to feed operator control inputs to the control/simulation system. In order to provide control flexibility, many modes of operation are used to change the relationship between the handcontroller inputs and the manipulator response. This latter mode dependence requires the operator to adapt his/her hand-eye mapping model at each change of mode [Held, 1991].

In order to properly use any telerobotic system, an operator needs feedback on the manipulator status and the remote environment configuration. As manipulator operations become more complex, the number of feedback parameters increases [Kim, 1993]. To facilitate feedback interpretation for the operator, a few solutions have been proposed such as Graphical User Interface (GUI) [Kim, 1993], robotics vision aids [Brooks, 1992], stereoscopic camera viewing [Drascic, 1991] and audio and tactile feedback [Caldwell, 1994]. Even when using such feedback tools, an operator needs to monitor many parameters at the same time and maintain a mental representation of the current situation; a phenomenon described in the literature as "situation awareness" [Stytz, 1993].

As a result of time delays in the communication links between the control/simulation or slow manipulator response to inputs due to inertia constraints, there is often a delay between operator inputs and the display of the manipulator and environment response to the given manoeuver. This time delay greatly slows down the execution of a task [Pook, 1995]. In order to alleviate this problem, predictive displays [Pook, 1995; Kim, 1993; Brooks, 1992] and operation preview [Kim, 1993] have been proposed. Predictive display shows the predicted position of a manipulator in the near future (i.e. usually a few

seconds or minutes) superimposed on a live image of the real manipulator, as the system is operated. Operation preview is the next step in simulation, whereby a complete representation of the manipulator and the remote environment is built and can be controlled with no time delay.

Intuitive interfaces, feedback mechanisms, and manipulator response times are some of the general difficulties encountered in teleoperations. System limitations and environment specifics can further aggravate the presented problems. Such factors will be presented in chapter two, when the MSS is discussed.

1.1.3 Applications.

Telerobotics is used in many areas where the remote environment is too hazardous for human operators. Telerobotic applications are found in dangerous working environments, such as nuclear waste disposals [Immega, 1995] , bomb removals [Drascic, 1993], and space structure assemblies [Backes, 1993].

Telerobotics is also used in applications where sending a human operator is costly or impossible. Such applications are found in sub-sea exploration [Sayer, 1992] and lunar exploration [Hart, 1987] with the Russian Lunokod .

In all of these applications, the use of telerobotics allows the operator to perform a task from a safe location, while the manipulator works in some distant or harsh environment.

1.2 Virtual Reality Technology

Due to the fact that training in the 0g environment of space on board the International Space Station, which is yet to be built, is not possible, new ways of conducting training had to be found [Logan, 1995]. As a result, the Canadian Space Agency decided to explore the use of VR technology as a tool for teleoperator training. This led to the design

and implementation of VOTE, which is presented in this thesis. The following section introduces the components of VR systems, VR applications, and current research in training and other fields, together with some important human factors involved in VR system design.

1.2.1 Virtual Reality Systems Overview

Virtual reality (VR) can be defined as "a high-end user interface that involves real-time simulation and interaction through multiple sensorial channels" [Burdea, 1993], where these sensorial channels usually consist of visual, audio and tactile senses.

Burdea describes three features that should be part of any virtual reality system:

1. Immersion, or the feeling of "being there", which can be achieved, in part, by coupling head position and orientation with visual and audio feedback generated by a computer system [Kalawsky, 1993]. As the user moves inside the simulated world, the feedback information is modified accordingly in real-time.

2. Interaction, which is the process by which the user can modify, by his/her actions, the virtual environment in which he or she is immersed. This interaction can be supported by joystick, instrumented gloves, voice commands, etc.

3. Imagination, which takes advantage of the fact that the virtual environment can include objects, tools and behaviors that have no counterpart in the real world [Krueger, 1991].

In order to provide the aforementioned features, different technologies must be brought together to form a virtual reality system. These are input devices, computer simulation, and feedback devices.

1.2.1.1 Input Devices

Data gloves are the best-known input devices for VR applications. These allow hand and finger movements to be measured and sent to the simulation, which in turn creates and animates a virtual hand which allows object manipulation in the virtual world [Slavkoff, 1997]. Gesture recognition of the hand and fingers has also been used to provide command inputs to a robot [Cannon, 1997].

Mouse [Coryphaeus, 1996] and handcontrollers (or joysticks) can also be used for navigation. Handcontrollers are often used to provide inputs to either drive a vehicle, fly a plane, or control a manipulator in the VE.

As previously mentioned, most VR systems achieve immersion by tracking head motion and updating the visual channel. To achieve proper graphics-head position mapping, a 6 DOF sensor is attached to the head of the user. In applications using a Head Mounted Display (HMD) the sensor is attached to the HMD structure. These sensors provide position (i.e. x, y, and z coordinates) as well as orientation data (i.e. yaw, pitch and roll angles).

Tracking systems use a mechanical boom [McDowall, 1990], acoustic ranging [Applewhite, 1991], optical tracking [Wang, 1990], magnetic fields [Ascension, 1996] or extraction of position from camera views [Zeevie, 1990] to locate a sensor. Acoustic and magnetic tracking systems are the ones most frequently used, since they do not require special room set-up (as optical tracking does), and do not require extensive processing by the host computer (as is the case with extraction from camera views). A complete survey of tracking performance (e.g. noise level, accuracy, range) is found in Meyer [1992]. Some tracking performance requirements for VR systems will be discussed in section 1.3.2.

Whereas data gloves and handcontrollers are popular input devices, voice recognition systems have been proposed to provide users with additional or alternative control inputs [Miner, 1995], when their hands are not free. Voice recognition usually requires dedicated hardware and software, and is classified as either user dependent or user independent. User dependent voice recognition requires training with each individual user in order to operate properly, whereas user independent systems can recognize voice commands from most users without previous training. The Verbex 7000 System is a good example of a speaker-dependent dedicated speech recognition peripheral [Verbex, 1990].

1.2.1.2 Computer Simulation

The computer simulation is the heart of the VR system. One of the main tasks of the simulation computer is to render the graphical representation of the objects composing the virtual environment. This task is accomplished using computer graphics techniques [Foley, 1990]. Each object present in the VE has an associated geometry depicted as a polygon mesh, a position and orientation, as well as some behavior. In order to produce visual feedback adapted to the user's inputs, the rendering software follows a cycle whereby object characteristics and user position are updated, and sensorial feedback channels are computed and transmitted to the user.

Improving the realism of a VE usually involves a larger number of objects of increased geometric complexity, as well as higher screen definition [Fuchs, 1989]. This, in turn, means a larger number of polygons to process, and an increased number of pixels to process. In order to prevent a large lag time and low update rate, the simulation software / computer hardware must perform a large number of computations per second. Graphical rendering then becomes the bottleneck of a VR system [Burdea, 1993], leading to extended research in real-time graphics.

Dedicated hardware, such as the Reality Engine series from SGI, is now available which allows typical graphical operations, such as Z-buffering, texture mapping, anti-aliasing and Gouraud shading, to be off-loaded from the general purpose CPU. Such hardware element chains dedicated to graphics form a "graphic pipe" [Silicon Graphics, 1994b]. Parallel computer architecture [Fuchs, 1989] has been used, resulting in high frame rate, but at a very high cost.

Rendering speed can also be increased using certain techniques at the virtual environment design stage, such as cell segmentation [Pimentel, 1993] and multiple levels of details [Latham, 1993]. Cell segmentation is a process by which the virtual world is divided into smaller "universes" [Burdea, 1993]. This process allows only the objects within visible cells to be processed, thus leading to a lower average polygon count. This technique is often used for virtual environments representing large buildings [Funkhouser, 1992], where each cell represents a single room. Multiple levels of detail, on the other hand, use different geometrical representations of objects in relation to the distance between the viewer and the object. As the user moves away from an object in the VE, fewer details are visible, therefore less complex geometry is used to represent the object. As in the case of cell segmentation, this technique reduces the number of polygons to be processed, thereby increasing the achievable frame rate.

1.2.1.3 Feedback Devices

Feedback devices include visual, audio and tactile output devices. Since human vision is the most powerful sensorial channel [Burdea, 1993], visual feedback is the most important feedback mechanism used in VR technology. Most VR systems use stereoscopic viewing devices that provide each eye with an image generated for that particular viewpoint position in the VE. Providing a 3D view of a VE can be achieved using a single monitor with shutter glasses synchronized with a high refresh rate CRT screen [StereoGraphics, 1993]. By alternatively showing left and right eye views on the entire screen and shutting the appropriate lens of the glasses, stereo viewing of the scene

is achieved. However, the viewing volume is constrained to the one of the monitor, and large movements of the head make the monitor to be seen as moving outside of the field of view (FOV) of the user.

Typically, HMDs are used for highly immersive VR systems. HMDs are equipped with separate screens closely located to the user eyes. Through the use of special optics, screen images are magnified to fill the maximum FOV for each eye [Burdea, 1993]. Screens used to produce images usually include CRT or LCD technology. Apart from being more expensive and heavier, CRT-based HMD provide higher image resolution, better contrast and stronger brightness than a LCD display [Holloway, 1993]. In order to alleviate the weight problem associated with high resolution CRT screens, while keeping high resolution, fiber-coupled head-mounted displays have been successfully tested [CAE, 1986]. The very high cost of such systems currently prevents their widespread use [Holloway, 1993]. Other types of 3D displays have been tested [McKeena, 1992], but are not widely used.

Binocular FOV, image resolution, overlap percentage, and weight are the main characteristics used to define HMD performance. The relationship between these parameters and the user will be discussed in section 1.3.2.

Just as visual feedback is processed for each eye, sound needs to be processed for each ear, in order to convey the position and distance information of the sound source. By modeling sound amplitude change as a function of distance as well as interaural time differences and head shadowing effects (known as Head-Related Transfer Function or HRTF), it is possible to produce stereophonic audio signals that convey the necessary sound distance and orientation cues, referred as *sound spatialization* [Wenzel, 1992]. Accurate modeling of sound propagation to the human ear includes the computation of HRTF, which requires frequencies domain calculations (i.e. Fourrier's Transforms) which can slow down the VR application. As with tracking, sound spatialization is usually implemented using a peripheral hardware, to which the VR application provides user,

sound, and source parameters. The Convolvotron and the more recent Acoustetron from Crystal River Engineering [Foster, 1992] are examples of such sound spatialization peripherals.

Audio feedback has also been used to provide system status or confirmed voice activated commands [Miner, 1995] in robot control applications.

Tactile feedback includes force, pressure, vibrations and temperature, and is usually applied either directly to the user's hands via an instrumented glove [Caldwell, 1994], or as feedback on handcontrollers [Silicon Graphics Inc, 1996] for telerobotics applications. Piezo-electric transducers have been used, with some success, to simulate texture sensation and pressure feedback using variable frequency and amplitude vibrations [Caldwell, 1994]. Thermal feedback has also been tested by the same team.

Providing immersive tactile feedback is a difficult task, since tactile sensory organs are not localized [Holloway, 1993], and properly "fooling" haptic senses cannot be achieved though the use of one or two displays, as it is the case for visual feedback. Limitation and complexity of the hardware needed for complete hand or body tactile immersion have prevented the widespread use of this technology. However, mechanical feedback via handcontrollers or other hand-held objects (such as a box on which force feedback is applied to simulate the inertia of a Hubble Space Telescope unit) is now being used in telerobotics and astronaut training [Cater, 1995].

1.2.2 Virtual Reality Applications

Today's applications of Virtual Reality technology can be grouped under three different categories:

1- **Visualization and design**

2- **Telerobotics**

3- **Training**

The fields of Science and Engineering have long seen the potential of VR technology for the visualization of complex 3D systems and processes. VR applications are found in engineering for airflow visualization in wind tunnels [Robertson, 1991], for planetary surface viewing [McGreevy, 1993], for assessing parts accessibility in aircraft design [Adam, 1993], in architecture design [Stredney, 1995], CAD interfacing [Cooke, 1993], and in medicine for endoscopic imagery viewing [Satawa, 1997], just to name a few.

Telerobotics, through telepresence, uses the immersion achieved by VR technology in order to provide intuitive control inputs and feedback to teleoperators. VR tools have been used to issue high level commands to robots [Cannon, 1997; Miner, 1995], and as operator aids in the form of predictive displays [Kim, 1993]. Human-to-robot skill transfer, by means of operator demonstrations, is another field of robotics in which VR has been successfully applied [Takahashi, 1992].

1.3 Using Virtual Reality for Training.

Training applications of virtual reality are countless. The U.S military has been one of the driving forces behind the development of VR. It has applied VR technology to pilot training, battleship bridge simulations [WTH, 1997], large scale land forces operations in a networked, multi-user VE called SIMNET [STRICOM, 1997] and missile launch trainer [Division, 1993]. VR is also being used in the field of emergency medicine for the training of surgeons [Delp, 1997], and for the training welders in the power industry [Tam, 1996] .

While VR is widely used as a means of controlling telerobotics systems, its application to the training of teleoperators is less common. One of the best examples of VR used for teleoperator training is the simulation conducted by NASA for Hubble Space Telescope Repair Mission [Loftin, 1994]. In this particular application, one astronaut operated the Shuttle Remote Manipulator while another astronaut, immersed in a VE , was "sitting" on the manipulator's end and manually handling the payloads. This training VE was aimed at

developing the crew situational awareness and practicing EVA astronaut-operator voice communication protocols. The report showed a reduction in training time and greater training transfer (i.e. from the simulation to job task) as compared to traditional training methods involving real hardware.

While VR systems dedicated to training are not common, the tools developed as operator aids in teleoperations have been used for training purposes. Operation preview and playbacks provided by VR simulation are reported in Miner [1995] as being used as training aids.

1.3.1 Benefits of using Virtual Reality Technology

The immersion achieved using VR technology tools is useful for developing mental models of complex 3D environments. Stereoscopic viewing together with the tracking of head movement provide most of the visual cues that humans use in their everyday life.

One of the features of VR is that, as an interface, it preserves the visual-spatial characteristics of the simulated world [Regian, 1993]. As an individual navigates and views objects inside the VE, the relative position and size of objects are maintained. It is assumed that the visual-spatial and motor-response fidelity found in VE will enhance both performance in the VE and transfer of skill to the operational environment [Kreuger, 1991]. These assumptions have been generally supported by experimental research [Regian, 1993; Bliss, 1997; Bailey, 1994].

Another advantage of VE is that it offers the possibility of presenting both small-scale and large scale spatial information in a 3D format that eliminates the need to translate multiple 2D views to 3D [Regian, 1993]. It has been shown that individuals can construct a mental representation of 3D environments using sequential, isolated views of small-scale space presented in 2D [Hochberg, 1986; Regian, 1993]. With its user-controlled

navigation capability and 3D representation of the environment, VR can readily provide the user with the needed spatial knowledge in a 3D format [Regian, 1993].

The use of stereoscopic viewing provides the human visual system with some important cues about depth and relative size of objects. Retinal image disparity is one binocular depth cue that can be simulated in a HMD. By providing each eye with a different image taken from slightly different locations (eye locations on the head), HMD allows the human brain to extract depth information. Inter-pupillary distance (IPD) varies from one person to the other [Burdea, 1993]. In order to ensure that images provided by the display are correctly generated, both eye viewpoint position in the VE must be set accordingly and the convergence angle adapted to the HMD used. The use of head motion tracking allows the user to take advantage of a monocular depth cue called motion parallax [Teittinen, 1996].

Just as the human brain can process two different images perceived simultaneously though stereoscopic viewing, it can also process monocular images presented sequentially to extract depth information. As a user moves from one location to another, object projections on the retina move in relation to one another by varying amounts, in accordance with their related relative positions [Levine, 1985]. This is a phenomenon called *monocular movement parallax* [Teittinen, 1996].

Imagination is an important feature of VR technology, as previously discussed in section 1.2.1. The application presented in this thesis has relied heavily on imagination through the creation of "virtual tools" that provide functionality and representation objects for which there is not always physical counterparts in the real world. These will be discussed in detail in chapter three.

1.3.2 Important Human Factors in the Design of Virtual Reality Systems.

As for any GUI design, human factors are of prime importance in VR systems. While some faulty design features can lead to frustration or difficulty in using most GUI applications, insufficient performance of VR systems can lead to more severe user discomfort, such as motion sickness.

Motion sickness can arise in VR systems when discrepancies develop between visual feedback and the user's position information provided by his/hers vestibular system [Regan, 1993]. Such effects appear when there are low update rates and large lag times. Update rate refers to the frequency at which the computer modifies or updates the displayed image. When update rates are lower than 15 Hz, motion appears to be discontinuous [McKenna, 1992], which becomes distracting, and nausea can then be induced [Logan, 1995]. Lag time represents the elapsed time between a change of viewpoint position (i.e. user head movement) and the update of the visual imagery. Large lag times can arise from lag in tracking, whereby making the system unable to follow sensor movements as speed increases (called *phase lag*) [Kalawsky, 1993] or as a result of the simulation computer being unable to quickly update the graphics. Lag times smaller than 100 milliseconds are considered to be acceptable [Chung, 1989].

The resolution of the generated image and field of view (FOV) of the display contribute to the level of immersion achieved in VR, which increases with the display FOV [Hendrix, 1996]. The human binocular FOV is 120^0 vertical by 180^0 horizontal with 120^0 overlap [McKenna, 1992], which is much wider than what a typical HMD can provide [Holloway, 1993]. Resolution of the foveal region of the human eye is considered to be around 30 arc-sec, which would require a 4800 by 3840 pixel display screen viewed at 46 centimeters for equivalent resolution ! The resolution specifications of the human visual system can not be matched with currently available technology, which leads to the use of lower resolution, typically 640 x 480 pixels. Achieving photorealism in modelling, is estimated to require the display of 40 millions triangles/sec/eye [Fuchs, 1989] which is beyond today's computing capability [Burdea, 1993].

The abovementioned human visual system parameters cannot be fully met with currently available hardware. This leads to tradeoffs aimed at achieving acceptable performance [Logan, 1995]. Since low update rate and large lag time can lead to motion sickness, tradeoffs are usually made to maintain a minimum acceptable update rate by reducing image complexity (i.e. by reducing the number of polygons and using texture mapping), for example. Limiting image resolution produces degraded images with extended FOV. While high resolution with narrow a field of view might be needed for surgery training, a wider FOV with less resolution is an acceptable tradeoff for astronaut training for EVA [Logan, 1995].

1.4 Thesis Overview

In this thesis, the design and implementation of the Virtual Operations and Training Environment (VOTE) will be presented. In order to provide a context, Chapter 2 describes the functionality and limitations of the Mobile Servicing System within the International Space Station environment and their impact on teleoperations. A Virtual Operations and Training Environment system overview is presented in chapter three, along with a description of each module and its intended functionality. Chapter four presents the VOTE implementation. Using the same sequence as in chapter three, the implementation strategies for the overall system and a description of each module are presented. This is followed by a presentation and analysis of test results. The last section of the chapter outlines anticipated future work and improvements to be made to VOTE. The conclusion is presented in chapter five.

Chapter 2 The Mobile Servicing System

This chapter presents an overview of the International Space Station environment and a description of the Mobile Servicing System. The Mobile Servicing System functionality and the challenges its utilization provides to the operator are also discussed.

2.1 The International Space Station

2.1.1 General Description

The International Space Station (ISS) will provide access to a space based laboratory to the international scientific community. Apart from the use of microgravity in crystal growth, drug purification and metallurgy to name a few, other experiments will use this observatory above the atmosphere for astronomy study and Earth observation. The study of the long term effects of microgravity on the human body will also take a large part of the astronaut's time on board the space station.

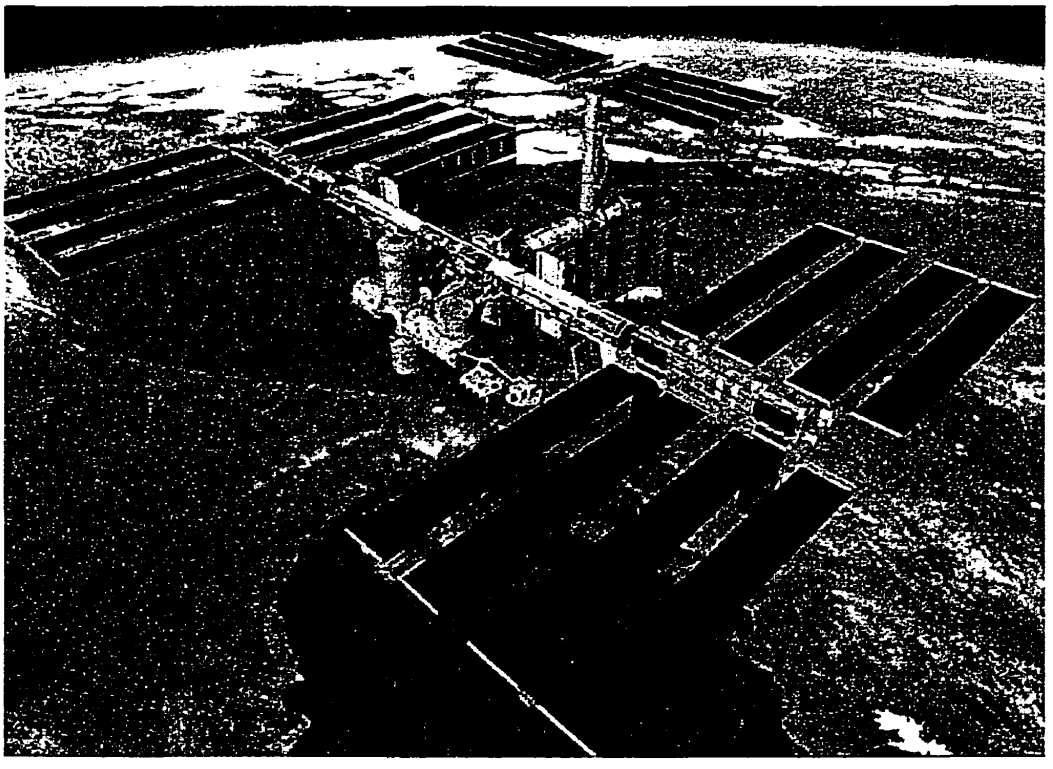


Figure 2.1 : The completed International Space Station as it will appear in 2003.

Figure 2.1 shows the ISS upon completion in 2003. The ISS is built around a central truss which spans over 100 meters, and which supports the large solar panels needed to provide electrical power and the radiators used to cool the ISS equipment. Centrally located are the pressurized modules in which the astronauts live and work. The modules are connected together using docking ports located at the modules' ends, or located on the multiple docking nodes. The ISS is accessed by the Space Shuttle or Russian Soyouz capsule using docking ports located at the ends of the US LAB module and the Russian Universal Docking Module.

As an international project, the ISS includes modules from United States, Russia, Japan, Europe and the Canadian contribution, the Mobile Servicing System (MSS). These modules will provide the living quarters, working space, and the needed life support systems to allow a crew of up to 7 to conduct experimental and observation work. Using the Space Shuttle, the ISS crew will be rotated, fresh supplies brought on board, and experiments brought back to Earth every three months. The ISS crew members will be selected from the participating country astronaut programs and will be trained together prior to their stay on the ISS.

2.1.2 Assembly Sequence

As it is currently impossible to lift the complete space station using a single launcher, placing the station on orbit will be achieved by launching each of the elements separately and assembling them together in orbit. It is planned that 31 Space Shuttle launches and 12 Russian rockets launches will be needed to bring the necessary components together with fuel and food supplies. While the first few modules will be either docked automatically or assembled using the Shuttle Remote Manipulator System (Canadarm), the later phases of the assembly sequence will be conducted using the Space Station Remote Manipulator System (SSRMS) provided by Canada. The SSRMS will be deployed on the ISS by astronaut Chris Hadfield in January 1999, and the ISS should be completed by 2003.

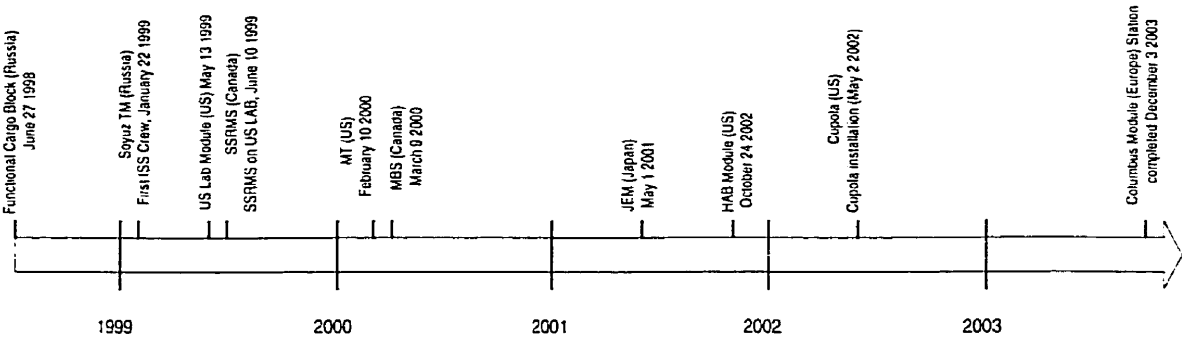


Figure 2.2 : The ISS assembly sequence time line.

Figure 2.2 shows the currently planned assembly sequence including the flights on which important ISS elements will be brought into orbit for assembly. The ISS will stay unoccupied from the first element launch in June 1998 until the assembly and power up of the necessary life supports systems in January 1999. During this period, the Space Shuttle manipulator will be used for the assembly, and the crew will live on-board the Shuttle and return to Earth after the modules brought up are assembled. From January 1999 , astronauts will be able to live on board the ISS, and later assembly tasks will be performed by both Space Shuttle and ISS crew.

On flight 6A in June 1999, the SSRMS will be brought-up and attached to the US LAB. From then on, the SSRMS will be able to conduct assembly, being controlled from the inside of the US LAB Module. On flight UF-2, the Mobile Base System (MBS) will be carried into orbit and placed on the Mobile Transporter (MT). The MBS will then become the storage attachment point for the SSRMS, and will allow the manipulator to move along the station's main truss.

2.1.3 The Space Environment

The ISS will be orbiting the Earth at an average altitude of 220 miles (350 km) at 51.6 degrees of inclination and will have a period of about 90 minutes. The ISS will thus experience an average of 16 sunrises and sunsets each day. Lighting conditions will thus change rapidly as the ISS leaves the day side of the Earth to enter our planet shadow. As currently experienced with the Space Shuttle missions, near Earth orbits require the operator to deal with many cycles of day and night work conditions within a single robotic manoeuver. The absence of air greatly reduces the amount of ambient light present in space, which makes shadows sharply defined and objects not directly exposed to sunlight extremely dark.

As space is free of gases, movements are not damped by air resistance. Motion induced oscillations, such as when the Space Shuttle docks with the ISS, can persist for long periods, and can damage long and light structures of the ISS, such as the solar panels. Robotic operations thus need to have to use a "gentle touch" to prevent shocks that could induce resonance within the ISS structure.

While the concept of weight disappears in space, the law of inertia is still at play. If microgravity allows the SSRMS to move the Space Shuttle around for docking, inertia prevents using high speed motion to do so. Trying to stop a module of a few tons by locking the manipulator joints at a speed of a few kilometers an hour would tear the SSRMS apart. Low speeds thus have to be used in space robotic operations in order to keep momentum at a manageable level.

2.2 The Mobile Servicing System

2.2.1 Physical description

The MSS consists of three main components: the Mobile Base System (MBS) which is the main attachment point for the Space Station Remote Manipulator System (SSRMS) arm. The Special Purpose Dextrous Manipulator (SPDM) will later be added to provide

the SSRMS with a "hand" that will allow the manipulation of small objects. Figure 2.3 shows the different components of the MSS sitting on the Mobile Transporter (MT) which allow the MBS to travel along the main truss of the ISS.

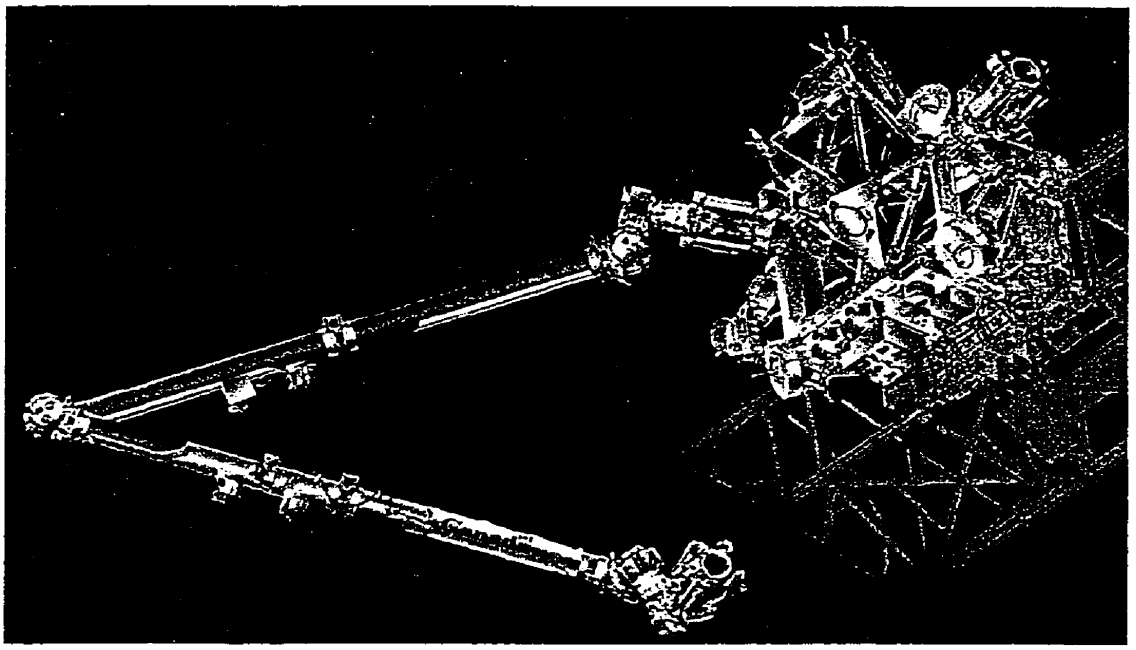


Figure 2.3 : The MSS components mounted on the Mobile Transporter (MT).

The SSRMS is the main component of the MSS, and will be the first one to be sent into orbit. The SSRMS is a 7 degree of freedom (DOF) manipulator symmetric with respect to its elbow, and equipped on both ends with a Latching End Effector (LEE).

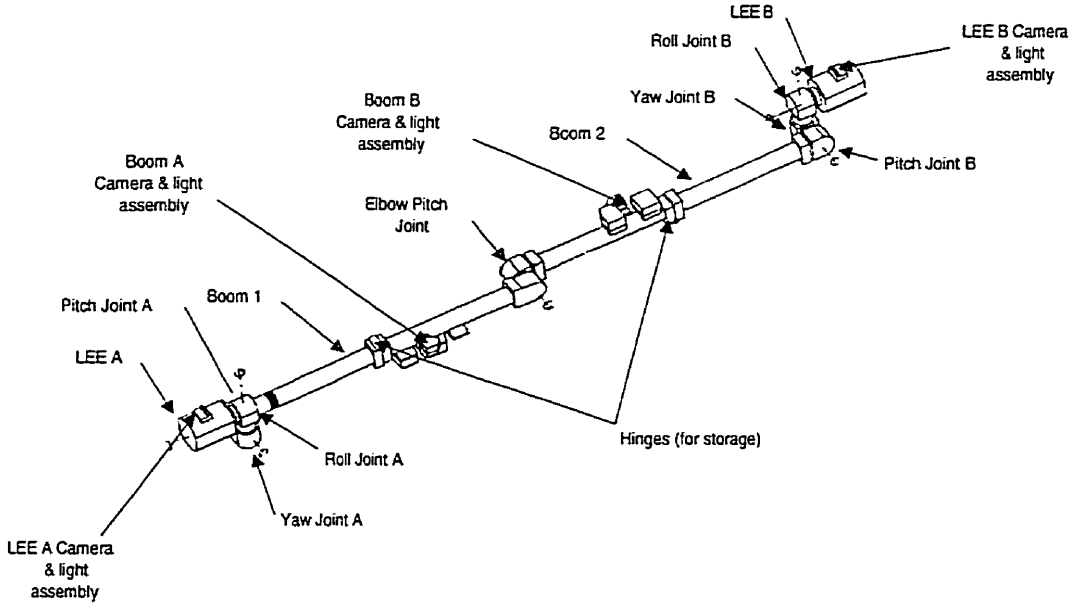


Figure 2.4 : The Space Station Remote Manipulator System (SSRMS).

The arm is 17 meters long, and has a mass of over 1500 kilograms. All of the manipulator's components are assembled using removable fasteners and electrical connectors which allow all of the units to be replaced in-orbit. Figure 2.4 shows the main components of the SSRMS.

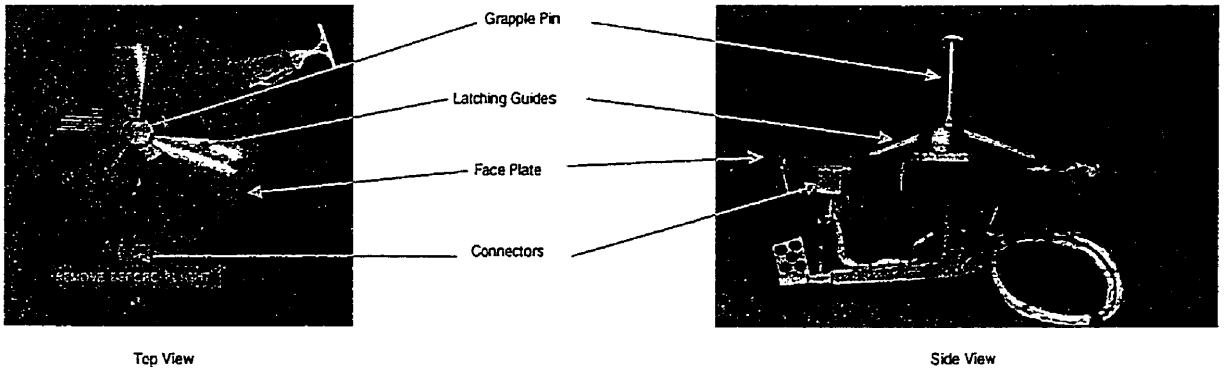


Figure 2.5 : A Power and Data Grapple Fixture (PDGF).

The LEEs use snare wires and a retractable wire carrier to mechanically secure the arm to special anchor points called *grapple fixtures* that are found on all the payloads that need to be handled by the SSRMS or the Shuttle manipulator. Some of these grapple fixtures

are also equipped with power and data connectors that allow the SSRMS electrical and computer systems to be connected to the ISS power and data bus for control. Such fixtures (see figure 2.5) are called Power and Data Grapple Fixtures (PDGF), and are found on multiple locations on the ISS.

The availability of electrical power and control data at PDGF, together with the SSRMS symmetry, allow the manipulator to connect one of its LEE on any of the PDGFs and use it as its base while using the other LEE for payload manipulation.

2.2.2 The Robotics Work Station

The Robotics Work Station (RWS) is the control console of the MSS. The RWS, pictured in figure 2.6, is composed of three screens on which exterior camera views are displayed to the operator, a Portable Computer System (PCS) used to configure the SSRMS control modes, two rate input handcontrollers (i.e. the Translational HandController (THC) and the Rotational HandController (RHC)), and finally, the Display and Control (D&C) panel on which camera controls, video routing, SSRMS lights and emergency stop switches are located.

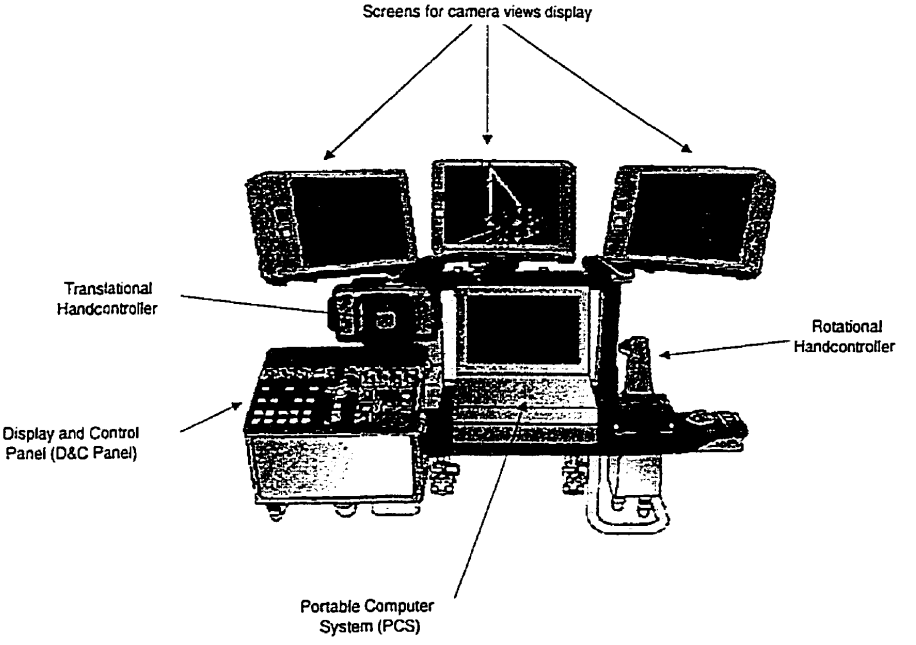


Figure 2.6 : The Robotics Work Station (RWS).

Each of the handcontrollers have both coarse and vernier control modes, the former being used for long displacements at higher speed and the latter used for shorter, low speed control for precise alignment tasks such as PDGF grappling.

2.2.3 Control modes

The control inputs given to the handcontrollers by the operator can be mapped to SSRMS movements using different *control modes*. Here are a few of the control modes that can be selected by the operator using the PCS and the D&C panel:

1- *Manual Augmented Mode*: In this mode, the inputs to the handcontrollers are interpreted as translation speed for the THC, and rotation speed for the RHC. It is the mode normally used.
2- *Single Joint Mode*: This mode allows the control of one joint of the SSRMS at the time. In this mode, a selection switch on the D&C panel is used to select the joint, and the push-pull movement of the THC is used to control the joint angular speed.
3- *Pitch Plane Mode*: This mode allows moving the SSRMS without changing the orientation and position of either LEEs using the RHC roll input. It is used to move the SSRMS elbow out of the way in certain manoeuvers.

For all these modes, the coarse and vernier speeds are available using the speed selection switch on the RHC.

2.2.4 Video Feedback and Overlays

Any of the ISS and MSS camera NTSC signals can be routed to any of the 3 display screens on the RWS. Most cameras on the ISS can be controlled in pan, tilt, zoom and aperture, while the cameras on both LEEs of the SSRMS can be controlled only in zoom and aperture. Camera control and video routing is done using the D&C panel switches.

Overlays providing numerical displays of the position and orientation of the LEE are available on the center screen, as well as graphical representations of force and moment sensed at the LEE. On each screen, the camera identification number can also be provided as a numerical display.

2.3 Operating the Mobile Servicing System

2.3.1 Manipulator Base Location

As explained in section 2.2.1, the SSRMS can operate from multiple locations on the ISS using PDGFs. This places the operator in a situation where the remote environment in which the manipulator operates changes from one manoeuver to the other. For each manipulator base location, the operator has to adapt his/her large and small scale mental model of the ISS and needs to be able to visualize the SSRMS movements in the nearby environment.

This change is even more dramatic when operating the SSRMS from the MBS after being carried from one end of the ISS to the other using the MT.

2.3.2 Payload Handling

The assembly of the ISS will requires the handling of payloads of different size and mass. The size of the payload changes the volume needed to carry out any manoeuver, and calls for the operator to adapt in order to keep safe clearance distances. The mass of the payload also affects the maximum speed allowable and the minimum distance required to immobilize the payload at any time, called *stopping distance*. Payload handling brings another difficulty: the LEE camera FOV is usually blocked by the payload.

While most payload grappling will be done using grapple fixtures, some tasks will require an astronaut to be standing at the tip of the arm using foot restraints. Such Extra

Vehicular Activity (EVA) will require the use of voice communication protocols between the EVA astronaut and the SSRMS operator inside the US LAB Module.

2.3.3 Pedipulation

Pedipulation is the manoeuver by which the free LEE attaches to a PDGF to become the new SSRMS base. This manoeuver first involves grappling and mechanically securing a fixture with the free LEE, and establishing power and data connections. The arm kinematics are then inverted, which make the newly grappled fixture the new SSRMS base. The connectors on the LEE initially used as the base are then retracted, and the PDGF unberted.

Performing pedipulation manoeuvers many times allows the SSRMS to "walk" on the ISS from PDGF to PDGF.

2.3.4 Frames of Reference

Using the SSRMS to assemble modules with precision requires the use of frames of reference with respect to which positions and orientation can be expressed. Many of the frames of reference defined on the ISS are used for robotic operations and allow manipulator trajectory points to be defined within teleorobotic procedures [Ferrara, 1996].

The Point Of Reference or POR is defined as being the tip of the free LEE. The POR position and orientation can be expressed with respect to any frame of reference and displayed to the operator on the RWS. The frame of reference used to express the POR coordinates is called the *display frame* .

Another reference frame is used to map the POR movement to the handcontroller inputs. This frame thus defines in which direction, for example, the POR moves when a +X input is feed in to the THC. A reference frame used for such mapping is called the *command frame.*

In a typical grappling operation, the display frame used is the one located at the PDGF origin, and the command frame is the one located at the POR itself. Using such a selection, the operator manoeuvers the SSRMS in order to bring all of the displayed x,y,z coordinates to zero (i.e. the POR is then on the PDGF origin). Having selected the command frame at the POR makes the interpretation of the LEE camera view easy since the inputs on the THC and RHC directly control the LEE view: up on the THC move the view up, positive pitch make the LEE to look up, etc. Figure 2.7 shows the interactions between control, camera view and displayed values in the grappling manoeuver.

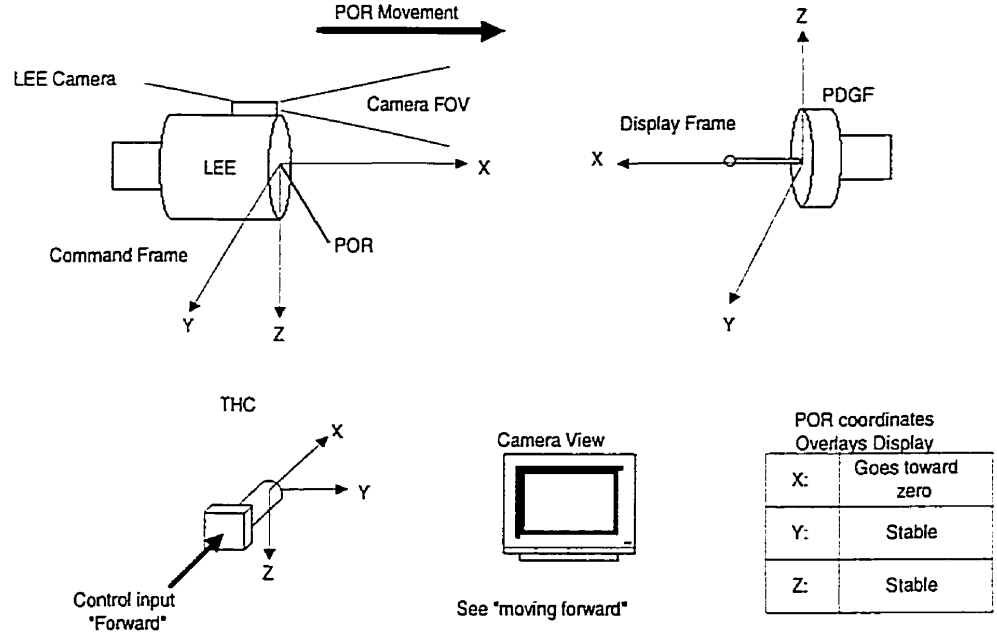


Figure 2.7 : Interactions between control, camera view and displayed values in a grappling manoeuver.

Properly interpreting camera views, the POR coordinates displayed and the effect of inputs on both handcontrollers on the POR movement given a display and command frame is not an easy task. When the LEE camera is not available (i.e. when a payload is attached to the LEE) and the command and display frames are not aligned, interpreting other camera views, displayed POR coordinates and control input mappings is a

challenging task. Figure 2.8 shows an example of the interaction between the handcontrollers inputs, LEE camera view, POR display and command frames in such a situation.

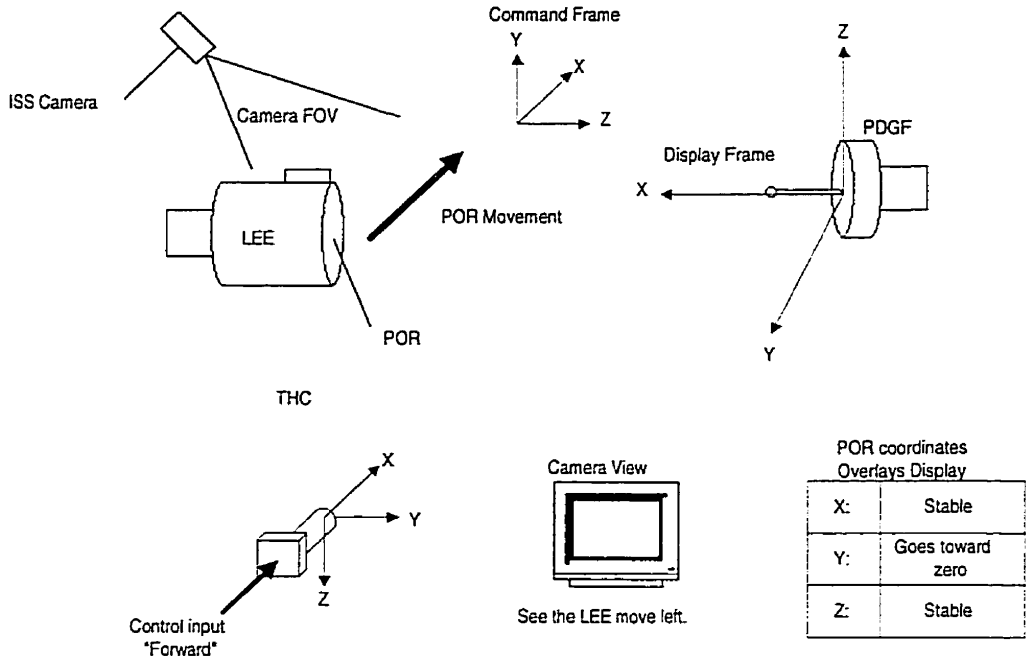


Figure 2.8 : Relationship between handcontroller inputs, LEE camera view, POR display and command frames.

Proper selection of the display and command frames by the operator and constant awareness of the relationship between both frames is critical for safe and quick operations.

2.3.5 Camera selection

Before the installation of the cupola, operators will have to rely only on cameras when operating the MSS. The selection of the camera views to be displayed on the three RWS display screens can be a difficult task to perform. Knowing each ISS camera accessible area for viewing and object shadows positions will be needed from the operator. While a single set of cameras could be used for a manoeuver taking place in a localized area,

switching between many cameras might be needed for a longer manipulator path. Some manipulator configurations will render the SSRMS boom cameras unusable, or will block the view of an ISS camera.

Apart from choosing which camera to use, the pan and tilt angles must be adjusted, together with the zoom and iris aperture. Maintaining the work area within the depth of field of the camera, avoiding pointing a camera toward the sun, and adjusting the iris and SSRMS spotlight when entering the Earth shadow or when moving into a module's shadow are examples of a few difficulties that the SSRMS operator will be facing on board the ISS.

2.3.6 SSRMS Autonomy, Mechanical Limitations and Safety Mechanisms.

The SSRMS autonomy is limited to performing automatic point to point linear trajectory and prerecorded manoeuvers. Some level of autonomy will be provided for docking manoeuvers by the Space Vision System (SVS) which uses optical tracking.

Each joint on the SSRMS can rotate from -270^0 to $+270^0$. This, together with the configuration of the joint clusters at each end of the manipulator, make possible self-collision between the LEE and the pitch joint. Some kinematics constraints can make the manipulator unable to respond to translation or rotation command when in manual augmented mode in certain configurations. These particular configurations are called singularities, and should be avoided by the operator.

Joints angles are monitored and joint motors are automatically stopped if a limit is reached. However, no safety mechanism can prevent manipulator self-collision or collision with the ISS components.

2.3.7 SSRMS Telerobotic Operational Space

As the "remote world" and "manipulator autonomy" have now been presented in section 1.1.1 , the MSS location in the operational space cube previously presented in figure 1.2 can now be better situated. The ISS represents a highly structured environment since all of its components are clearly identified and their locations precisely known. The behavior of the ISS and payloads to the SSRMS actions is predictable, leading to a high degree of "modellability". Finally, the SSRMS has very limited autonomy, which leads to high operator involvements.

2.4 The MSS Operations and Training Simulator

2.4.1 MOTS Functional Description

The MSS Operations and Training Simulator (MOTS) provides a real-time environment coupled with monitoring and control functions to allow MSS operations planning and analysis together with effective ISS crew and ground controller training [CAE, 1996].

The MOTS simulator consists of a RWS, two instructor control and monitoring consoles, and a supercomputer on which the simulation and graphic rendering engine run.

The replica of the RWS provides the operator with both handcontrollers, PCS interface for SSRMS configuration, D&C panel for camera and video routing, and 3 display screens on which artificial camera views are displayed.

The instructor stations are both equipped with a set of hancontrollers that can be used to override the RWS's ones. These consoles run an interface that is used to load a simulation and allow video routing for the instructor, overriding of the D&C panel, triggering of MSS malfunctions and simulation monitoring. This allows an instructor or operation personnel sitting at any console to have full control over the simulated SSRSM.

Within MOTS, graphical representations of the MSS hardware and the ISS environment are used to allow generation of the artificial camera views used by the operator. A kinematic and dynamic mathematical simulation of the manipulator allows all the control modes of the SSRMS to be used by the operator, as well as mimicking the arm dynamic response to those inputs. Video and simulation recording allows the camera views or a full simulation to be replayed to conduct trainee debriefing or perform procedures development and validation.

2.4.2 MOTS Implementation Overview.

The MOTS system architecture is built around a Common DataBase (CDB) in which the simulation parameters, graphical models, control status and commands are stored, accessed and updated by the functional modules (see figure 2.9).

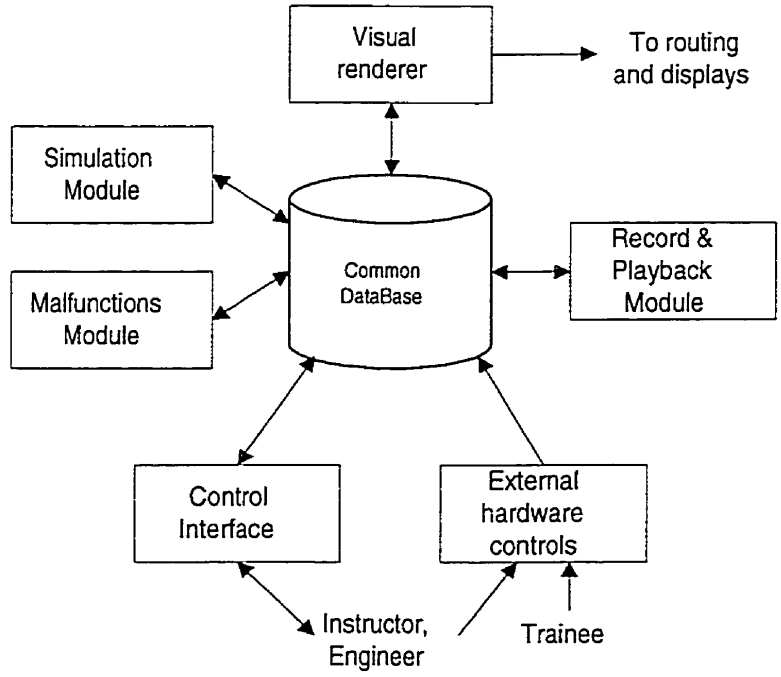


Figure 2.9 : MOTS Functional Modules.

The behavior of the SSRMS (i.e. kinematics and dynamics) is modeled by the simulation module. This module uses control inputs and current manipulator mode of operation to

periodically update the values of joints angles, POR positions, velocities and other physical parameters. These parameters are used by the visual renderer to update the graphical model and generate the appropriate camera views for display and routing. Record and playback capabilities are implemented by the Record & Playback module which allows CDB values to be saved, and displayed camera views to be recorded on video tapes.

The malfunction module modifies CDB variables to provide simulation of SSRMS malfunctions on request by an instructor. The access to MOTS functionality is provided by a control interface that sends requests to the functional modules via the CDB, while the external hardware, such as the handcontrollers and D&C panel switches, directly changes CDB variables used by the simulation module. The latter allows the mathematical model of the manipulator to be controlled.

2.4.3 Training with the MSS Operations and Training Simulator.

The astronauts selected as operators for the MSS will use Computer Based Training (CBT) in order to get the basic robotic concepts and knowledge of the MSS systems in preparation to coming to Canada for training. Once training starts at CSA facilities, they will attend classroom lessons covering the MSS systems and operation theory in depth, with manoeuver demonstrations on MOTS. As the course becomes more and more focused on the operation of the manipulator, trainees will practice chosen manoeuvers aimed at developing their operator skills and understanding of operations difficulties. These manoeuvers will be scenario based using realistic operation procedures.

The instructor will monitor the trainee's actions, and sometimes intervene during a manoeuver using his/her override controls. Instructors will also, at a later stage in the training process, trigger SSRMS malfunctions to make the trainee practice emergency procedures, malfunction diagnostics, and trouble shooting. Playbacks of a trainee's manoeuver will allow the instructor to conduct a debriefing using the simulator.

Chapter 3 The Virtual Operation Training Environment

In this chapter, the Virtual Operation and Training Environment (VOTE) is presented. The first section of the chapter describes the purpose and intended functionality of the system, while later sections present each of the functional VOTE modules used to achieve this training environment.

3.1 The Virtual Operation Training Environment Concepts

3.1.1 VOTE System Purpose and Functionality

The primary purpose of the Virtual Operations and Training Environment is to provide VR technology tools for the training of MSS teleoperators in the ISS context. As a secondary purpose, support to operation development and validation is considered. In order to provide the needed support for training and operation, the VOTE functionality has first to be defined.

As previously described in chapter one, the immersion achieved by virtual reality technology can help trainees develop functional 3D models of an environment. Thus, VOTE needs to provide trainee immersion in the ISS environment This is achieved in VOTE by providing the following :

1. **A 3D representation of the International Space Station in a VE.**
2. **Stereoscopic display of the VE to the trainee.**
3. **Trainee navigation in the VE, including head movement tracking.**

Chapter two outlined the complexity of the SSRMS operations and trajectory control. In order to allow the trainee to visualize such manipulator movements in the context of the ISS, the following functionalities are also provided by VOTE:

4. **An animated 3D representation of the MSS in the VE.**
5. **Generation of MSS robotic animation using real operations data.**

As explained in Chapter one, today's VR technology cannot provide optimal human vision cues with the current display technology, which makes it difficult to perform some distance judgment tasks in VOTE. The latter problem, together with the fact that VR provides the possibility to create graphical representations of abstract concepts (such as frame of reference), leads to the development of the following :

6. **Visualization tools used as training aids within the VE.**

These visualization tools, called Virtual Tools in VOTE, are discussed in section 3.1.2.

As a training tool, and later as a visualization tool used for robotic operations development and validation, VOTE functionality had to be readily accessible and tool parameters easily changed to fulfill instructors', trainees' and operations personnel needs. This lead to the need for :

7. **A Graphical User Interface providing visualization tools, robotic animation, trainee navigation monitoring, and control capability to the instructor or operation personnel.**

In order to provide the aforementioned functionality, the Virtual Operations and Training Environment system is divided into four functional modules:

1. The *instructor graphical user interface module* providing access to VOTE simulation and Virtual Tools.
2. The *rendering engine module* generating the required 3D graphics, such as the ISS, the SSRMS and the Virtual Tools.

3. The *peripherals module* supporting trainee inputs to VOTE, such as head tracking.
4. The robotics *simulation module* used to provide realistic MSS operation data to VOTE.

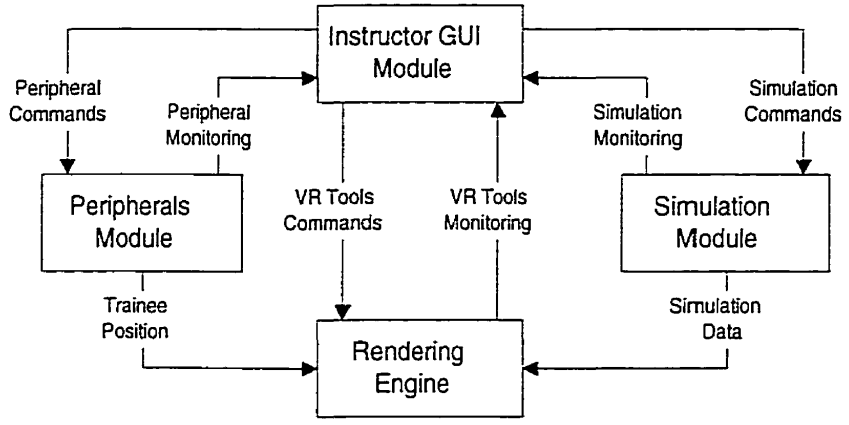


Figure 3.1: Components of the Virtual Operation Training Environment.

Figure 3.1 show the different modules composing the Virtual Operation Training Environment and the interconnections between these modules.

A detailed description of each of the VOTE modules' design and functionality will be presented in the following sections, after introducing the Virtual Tools concept.

3.1.2 The Virtual Tools

As mentioned in the previous section, the idea of using graphical visual aids for distance judgment tasks arose from the limitation of the visual display, together with the complexity of the MSS operations. Virtual Tools thus use the "3rd I" feature of VR: Imagination [Burdea, 1993] to convey in a more intuitive way physical concepts which are otherwise difficult to visualize.

Some of the functionalities provided by the Virtual Tools currently implemented within VOTE draw their inspiration from graphical tool ideas proposed by Thurston Brooks and Ilhan Ince [Brooks, 1992] as operator aids for telerobotic operations. The ARGOS Toolkit [Milgram, 1995] which is used to provide what is called *enhanced reality* to teleoperators also provided ideas that led, for example, to the development of the *virtual ruler*, discussed later.

The currently implemented Virtual Tools uses graphics to display physical parameters within the VE, such as display target range, manipulator speed and trajectory path. The Virtual Tools currently implemented include: the *Virtual Ruler*, the *Line Tool*, the *Axis Tool*, the *CamView Tool*, the *Handcontrollers Tool*, the *Stopping Distance Tool*, the *Robotic Tool*, and the *Teleport Tool*. Each tool is built as a graphical object whose shape, size, color, position and orientation are dynamically modified in real time to provided the intended behavior.

The construction of a virtual tool requires a 3D model of the tool to be built using a 3D graphical editor, and the model parameters to be controlled (such as scaling along the z axis of the tool for example) to be defined. To achieve the desired tool behavior, a procedure needs to be constructed. Such procedures use both command inputs and data from the "remote world" modeled in the VE to modify the tool. The modification of the tool's model is done by changing the previously defined graphical model parameters to achieve the desired graphical behavior. The integration of the tool model in the VE and the real-time execution of the associated behavior procedure in the simulation complete the process needed to have a virtual tool usable in the VE.

Following are examples of Virtual Tools currently implemented in VOTE. They are pictured in the same grappling manoeuver to clearly show how each tool presents different aspects of the same manoeuver.

a) The Virtual Ruler.

The Ruler tool mimics a tape used to measure distance between two points. A 3D representation of a measuring tape is created in the V.E. between the two selected points. This 3D representation consists of the tape itself, represented as a cylinder, and tick marks equally spaced at 0.5 meters intervals along the tape. The length of the ruler is also displayed numerically as a head-up display in the upper left corner of the left eye image.

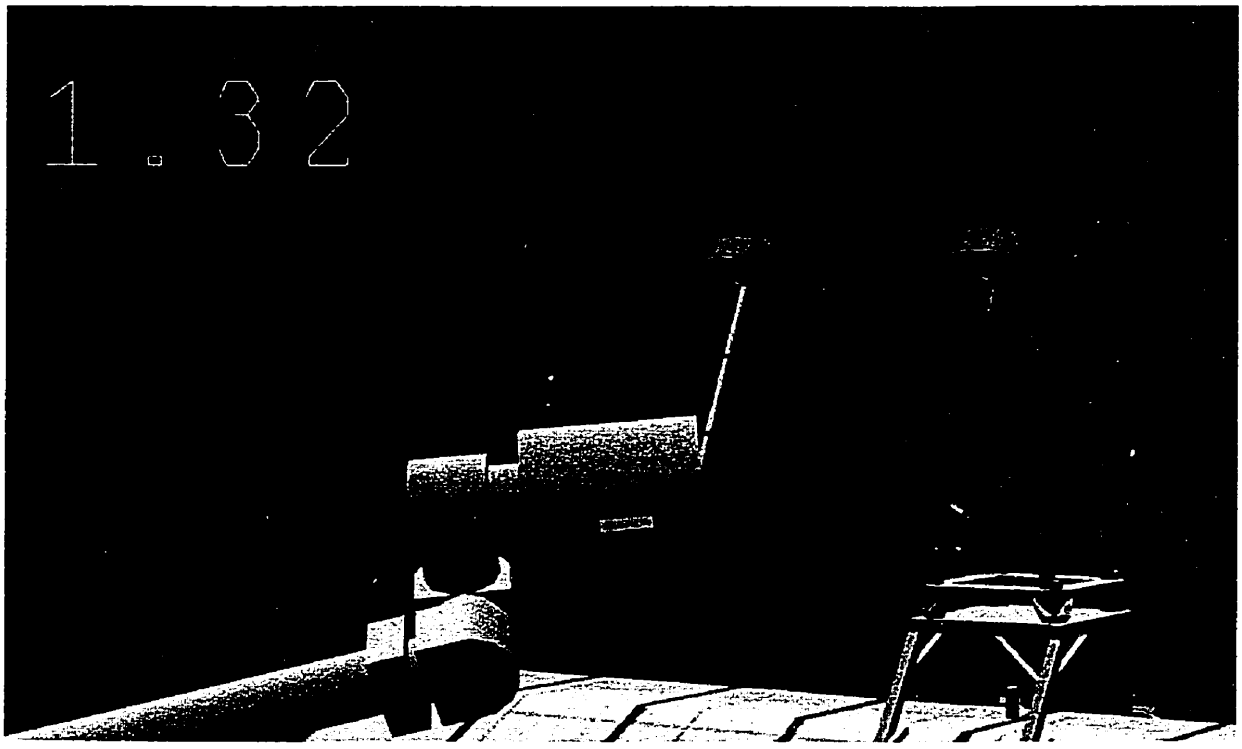


Figure 3.2 : The Virtual Ruler being used in a grappling manoeuver.

Figure 3.2 shows the Virtual Ruler being used to display the distance remaining between the SSRMS LEE and a PDGF on the Logistic Deployment Assembly (LDA) in the grappling manoeuver.

The Ruler can be switched on or off using the instructor GUI, and both of its ends can be attached to either preset fixed or selected mobile positions, or to sensors attached to the trainee's hands.

b) The Line Tool.

The Line Tool provides the capability to draw tridimensional lines in the V.E.. These lines are represented as tiny cylinders. A line is drawn by a mobile point, such a the SSRMS LEE tip or a tracking device, to which the Line Tool is attached. This tool thus allow the SSRMS trajectory followed by the trainee to be displayed in 3D in the V.E.. Such a trajectory is shown in figure 3.3. The Line Tool can also be used by the instructor to draw in the V.E., much like a chalk on a blackboard, when explaining clearance or other trajectory difficulties. This is achieved by attaching the Line Tool to a sensor manipulated by the instructor.

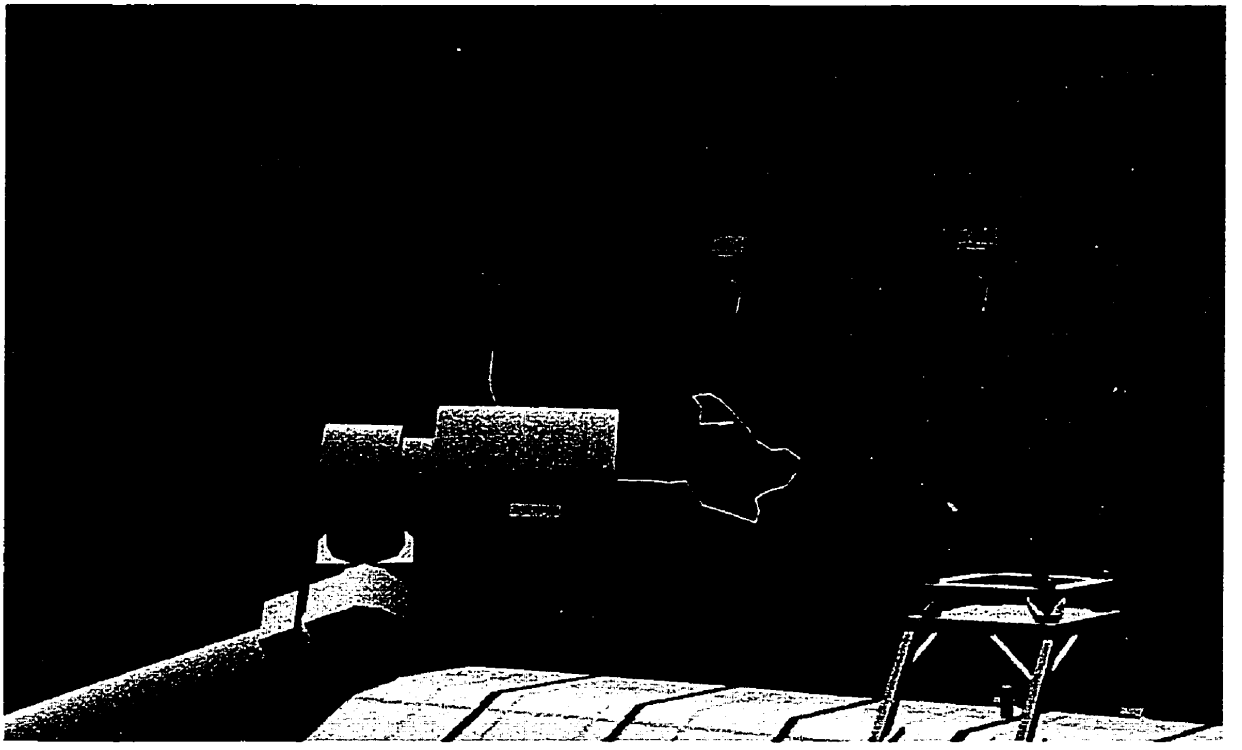


Figure 3.3: A SSRMS trajectory followed by a trainee during the approach to the LDA PDGF displayed using the Line Tool.

c) The Axis Tool.

The Axis Tool provides a graphical representation of the different frames of reference used for ISS operations. The Instructor can place coordinate axes at different positions and orientations in the V.E. to help the trainee to visualize the orientation of such frames of reference (for example, when used as CDF) . Figure 3.4 shows the Axis Tool being used to visualized the display frame (here attached to the LDA PDGF) used for the PDGF grappling.

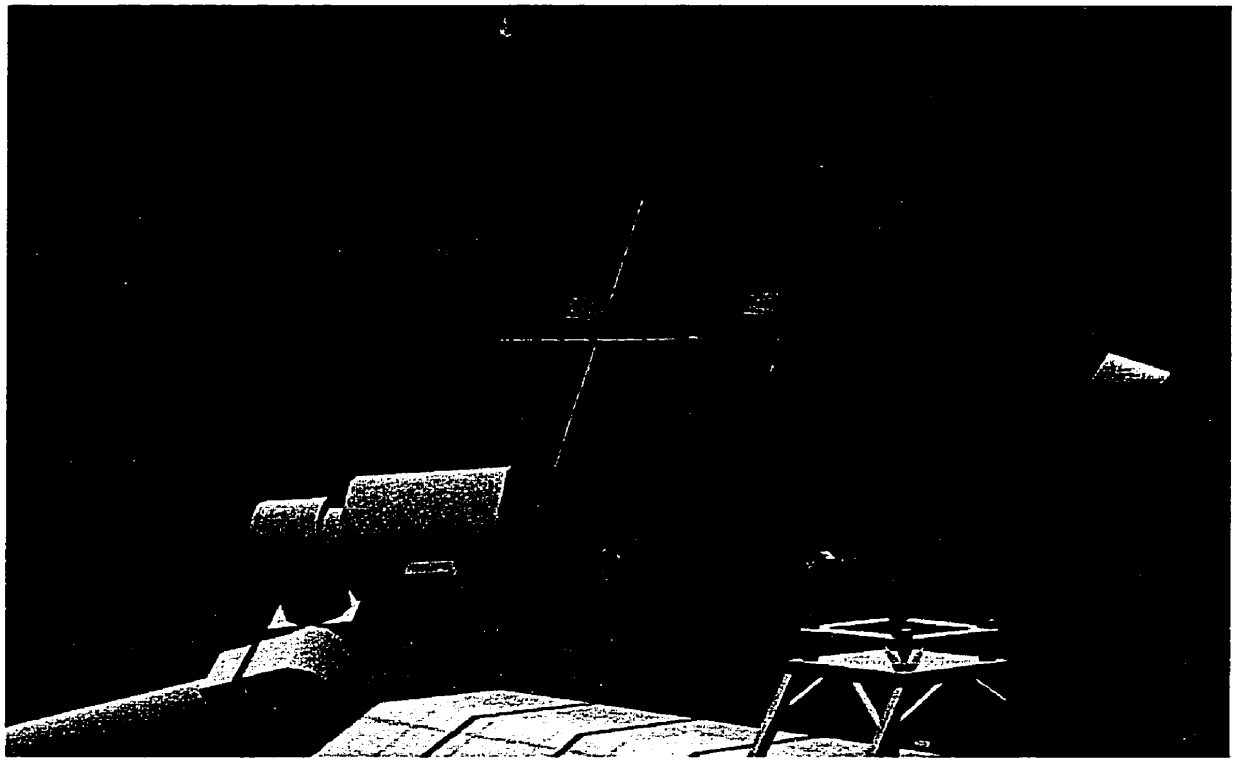


Figure 3. 4 : The Axis Tool being used to show the display frame used for a grappling manoeuver.

A coordinate frame can be attached to the SSRSM LEE tip to display the POR. This coordinate frame also includes a mobile vector that displays both the orientation (by its orientation) and the magnitude (by its length) of the POR speed. The XYZ axes are all labeled and also color coded (X = Red, Y = Green, Z = Blue) for easy identification.

d) The CamView Tool.

The CamView Tool provides multiple camera views to the trainee which are viewed as windows displayed in the left eye image of the HMD. Up to 3 windows can be displayed at the same time, and each window can be assigned a different camera. The background of all the windows is of a different color than the V.E. background in order to be easily seen.

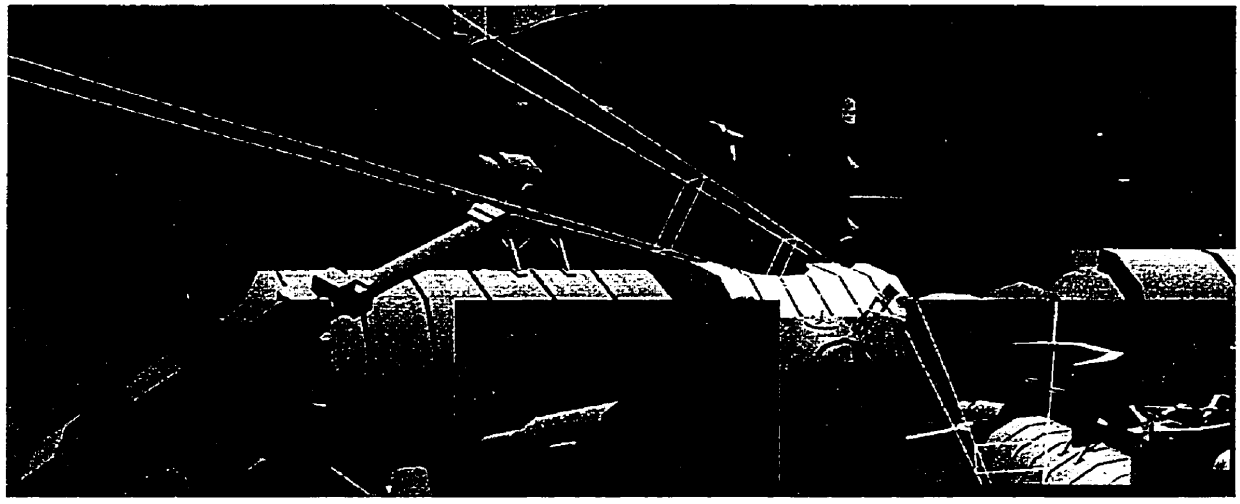


Figure 3. 5 : CamView Tool used to display camera views and FOV volume.

The instructor can turn any window on or off, and can assign a different camera to each of them. The camera selected can be controlled in pan, tilt, zoom and aperture. New cameras can be created at any location and orientation in the V.E. by the instructor as required. For each camera, a 3D representation of the FOV can be displayed in the V.E.. This FOV is displayed as a four sided pyramid for which the apex angle changes according to the zoom factor. The focal plane, the far and near limits of the depth of field are also represented as planes cutting the pyramid volume. The positions of these planes are computed using the zoom factor and iris aperture in a simple mathematical model of the camera optics and sensors [Serway, 1989]. Figure 3.5 show the CamView Tool being used to display three camera views and one FOV volume. The FOV visible here represent the volume viewed by camera associated with the center window.

e) The Handcontrollers Tool.

In the current implementation of VOTE, only off-line replay of operations performed on MOTS are possible. Relating the SSRMS movements to handcontrollers inputs then becomes difficult for the trainee since he/she would have to remember what controls inputs he or she provided at each moment to produce the movement shown on the replay. This led to the implementation of the Handcontrollers Tool.

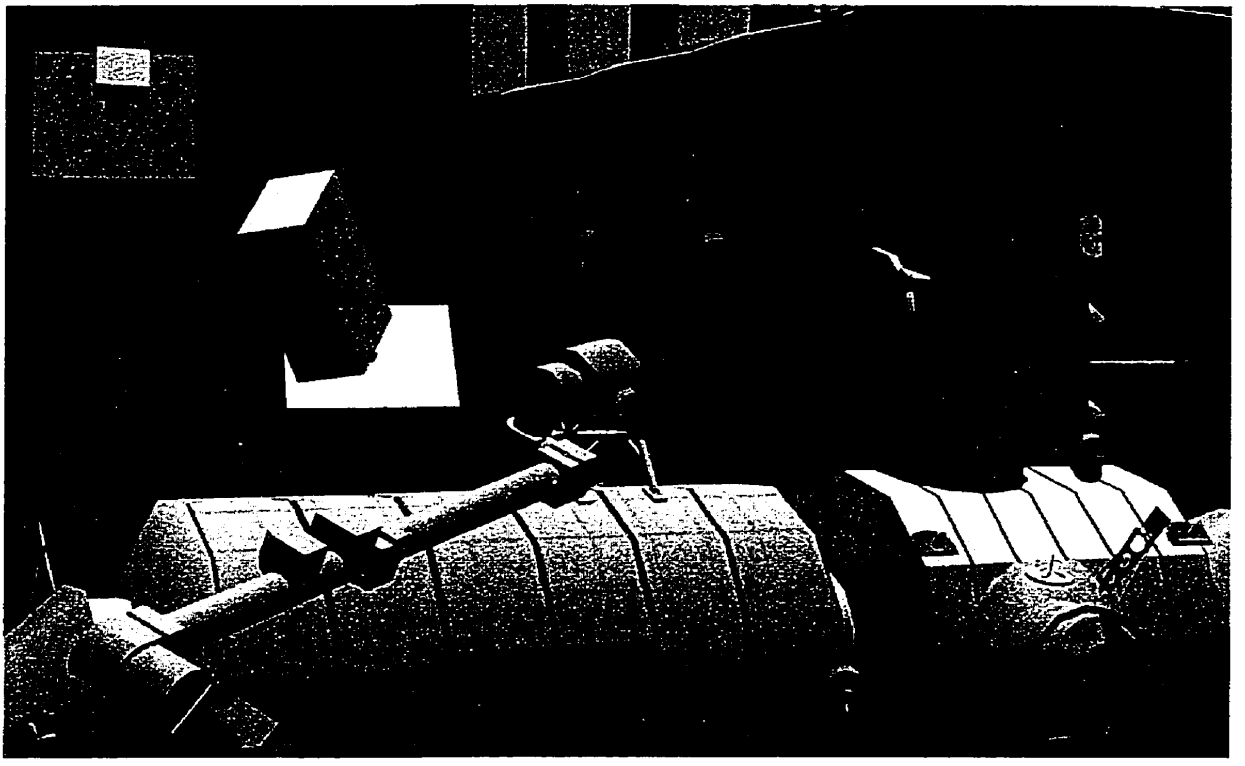


Figure 3. 6 : The Hancontrollers Tool being used in the grappling manoeuver replay.

The HandControllers Tool provides the capabilities to show the trainee his/her inputs on both the RHC and THC while viewing the movements of the SSRMS in the V.E.. Tridimensionnal representations of the RHC and THC are shown on both eyes in the HMD and stay fixed with respect to the trainee FOV, making them behave like a Head Up Display. The position and orientation of both controllers are proportional representations of the trainee inputs to the handcontrolers. These can be switched on or off by the instructor. Figure 3.6 shows the Handcontrollers Tool being used in the grappling manoeuver replay.

3.2 The Instructor Graphical User Interface

The instructor graphical user interface module is used by the instructor to configure the graphical simulation as well as for controlling and monitoring the robotic simulation, the training aids, and the peripherals. The interface main menu bar contains the following item groups:

1- Session
2- Virtual Tools
3- Monitoring
4- Simulation

The *session* group contains the options used to select the virtual environment initial parameters such as the virtual environment to be loaded, the data file used for simulation, or a VOTE initialization file. It also allows to exit the application and save the current VOTE session setups (all tools, simulation, and environment values) in an initialization file for later use. Once the VOTE initial setups have been selected, a VOTE session can be launched. On a session launch, the GUI launches the rendering engine with the selected initialization file, executes the robotic animator with the proper simulation file, and starts the needed peripheral servers.

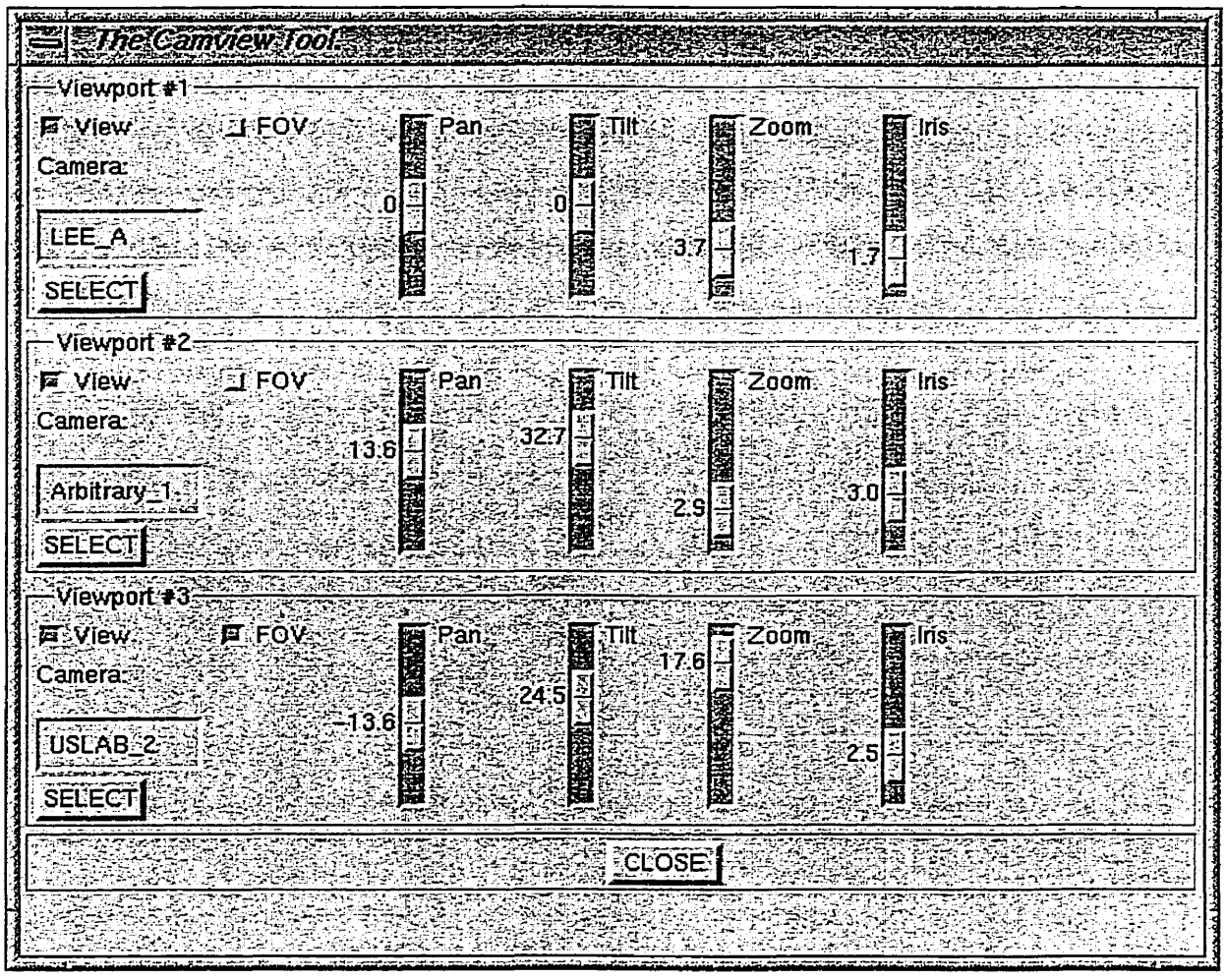


Figure 3.7 : The CamView Tool control window.

The *Virtual Tools* group allows the instructor to control each of the available training aids. On selection of a Virtual Tool, a window appears which allows the tool to be turned on or off, and its parameters to be set using text entries, scrollable lists, sliders or push buttons. Figure 3.7 shows the window used to control the CamView Tool, previously described.

The *monitoring* group allows the user to display some Virtual Tool parameters which change dynamically, such as the ruler length, as they are being used. As for the VR Tools group, a window appears on selection of a tool to monitored. Monitoring and control are displayed separately in order to require smaller windows and allow many of them to be seen simultaneously on the screen.

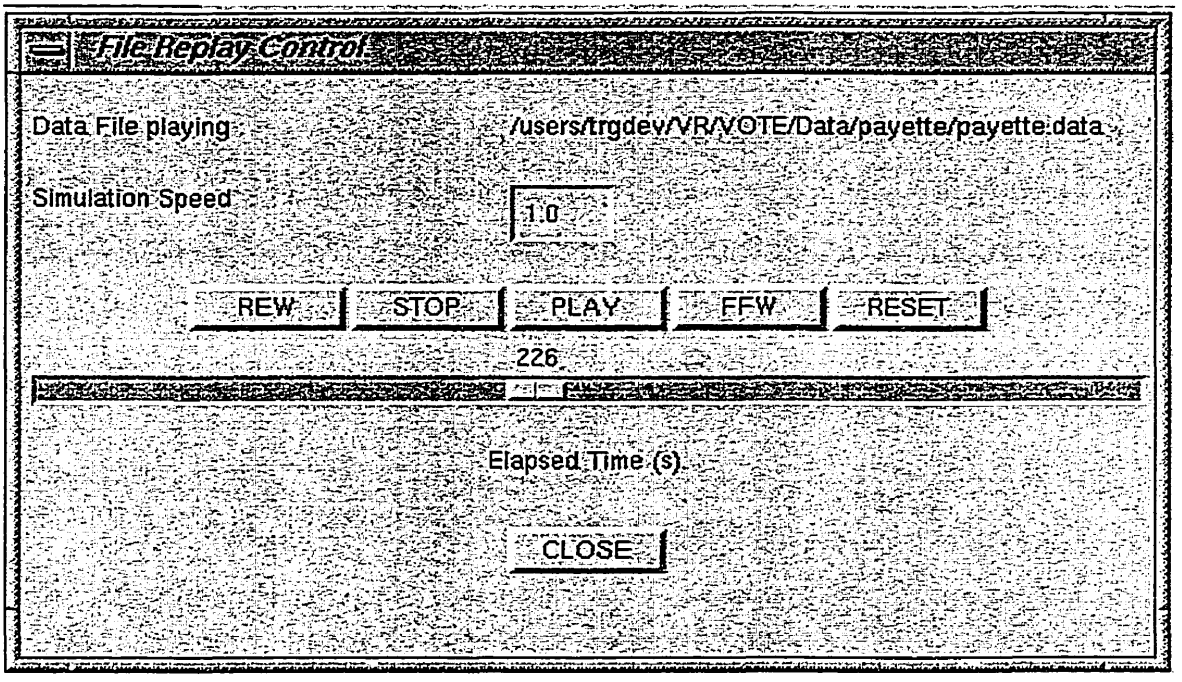


Figure 3.8 : Simulation control window.

The *simulation* group provides the instructor with control and monitoring over the simulation. Selecting the data file to be used for robotic operation replay can be done using a file menu while running a VOTE session. In the replay control window, shown in figure 3.8, the name of the simulation file being used is displayed, and the elapsed time of the simulation is shown as a moving cursor. The simulation replay can be controlled using standard playback functions such as rewind, stop, play and fast forward. The simulation can be replayed at a specified speed set by the instructor. It is possible to jump to a specific time in the simulation by moving the cursor to the needed elapsed time, which allows faster debriefing of a long robot manoeuver. The instructor can also select a different simulation file by changing the data file name and pressing the RESET button while an animation is running. This allows fast switching between simulation record files.

3.3 The Rendering Engine Module

The rendering engine module provides the graphical simulation in which the trainee is immersed. VOTE uses Easyscene from Coryphaeus Software as rendering engine which in turn uses Performer functions to manipulate graphical objects.

The EasyScene rendering engine uses a three phase rendering cycle that follows the application setup [Coryphaeus, 1996], as shown in figure 3.9. At each cycle the display is updated.

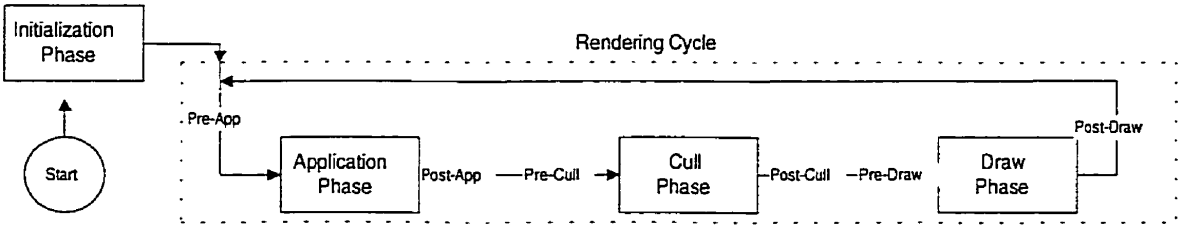


Figure 3.9 : EasyScene rendering engine cycle.

Upon EasyScene startup, an initialization file is read and used to setup the display, insert procedures from user defined modules to be used in the rendering cycle, and load the 3D models geometry in its "world database". These models are created using a 3D editor software called Designer's Workbench, which allows graphical models to be built, edited, and saved as .dwb format geometry files. Designer's Workbench allows hierarchy and dynamic link behaviors of objects to be defined within a model, which is needed when building articulated objects such as the SSRMS or the Virtual Tools. Such models contain geometry data and dynamic link behavior information, and are saved as .drt files. Both formats of models can then be loaded into EasyScene and displayed. For VOTE, a graphical model of the proper ISS configuration is loaded together with the articulated model of the SSRMS. Figure 3.10 shows the ISS together with the SSRMS as modeled in VOTE.

Figure 3.10 : The ISS and SSRMS modeled in VOTE.

Viewing the VE is done through the creation of "channels". Channels are associated to a window on the screen where the visual output is to be sent, and contains viewing frustrum parameters such as near and far clipping planes, vertical and horizontal FOV, viewpoint position and orientation. Trainee head-visual coupling is achieved by modifying two separate channels positions and orientations to match those of the trainee's eyes once each rendering cycle. Extra camera views are handled in the same manner.

The API provides the rendering engine with the basic functions used to manipulate graphical models within a 3D graphical environment, and view this environment in real time. These functions are used within user defined C language librairies, called "user modules" that are compiled together with the EasyScene main program and called within the rendering cycle. Procedures inserted at the initialization phase are executed once upon EasyScene startup, while procedures inserted in pre/post application, pre/post cull and pre/post draw are called at each cycle.

While some operations are done once in the initialization phase, such as the viewport and background settings, all the dynamic model behaviors, such as the positioning of the user

viewpoint, the robot joint angles or the virtual ruler length, have to be handled at each frame. These are handled through the use of the previously mentioned user-defined module's procedure, called once each frame to modify the VE database. As an example, the SSRMS joint values are update by a user-defined procedure at each frame, leading to a smooth animation.

As part of VOTE, the rendering engine uses the initialization file provided by the GUI upon startup. At runtime, the user-defined function within the rendering cycle uses commands issued by the GUI to make the Virtual Tools behave accordingly, and provide in return the tool status parameters for display. Data provided by the simulation module is used to animate the SSRMS, while data from the tracking peripheral couples the trainee head movement with the displayed graphics.

3.4 The Peripherals Module

The peripheral module provides VOTE with access to external peripheral hardware through the use of C programs. Such hardware could be tracking systems, handcontrollers, or a sound spatialization unit. In the present configuration, only the tracking system is available.

In order to use the tracking system, the peripheral module first configures the hardware by sending configuration commands to the system via a serial port. Once the tracking system is properly configured and running, the peripheral module reads the data tracking data sent by the hardware unit over the serial port, applies some low level formatting operations such as byte merging, and then provides the position and orientation data to both the instructor GUI for monitoring and to the rendering engine for VE updating (such as the trainee viewing position). This process is repeated in a continuous cycle, thus providing sensor position and orientation data to VOTE in real-time. Note that the peripheral module is executed upon launching a VOTE session in the Instructor GUI and is terminated upon exiting from VOTE.

Future peripheral units, such as the sound spatialization unit, could be handled in the same way. Despite the fact that the peripheral module is presented as a single entity, it can contain many independent programs associated with as many hardware units. Each hardware could thus be provided with its own driver program.

3.5 The simulation module

The simulation module provides VOTE with the capability to replay recorded operations from MOTS. By design, it is not intended to provide inverse nor forward kinematic calculations, since those functionalities are available through MOTS. As a result, VOTE cannot provide real time operator control of the SSRMS as of now. However, it is planned to later connect the simulation module to MOTS CDB in order to get the needed parameters in real time, allowing both SSRMS operator control and visualization in VOTE simultaneously.

```
TITLE                RESULT1C.VIS
TYPE                 C
XLABEL               CTSTSTTM
XNAME                TEST ELAPSED TIME
XUNIT
XFORMAT              F
YLABEL               MEASURED_JOINT_POSITION_SR
YNAME                Measured joint position (rad)
YUNIT
YFORMAT              F
NUMPTS               500
SMPLRT               1.000000
DATE                 Thu Apr 17 13:13:15 1997
      0.0000000000000000e+00       2.5245103836059570e+00
      1.0000000000000000e+00       2.5245273113250732e+00
      2.0000000000000000e+00       2.5245454311370850e+00
      3.0000000000000000e+00       2.5245633125305176e+00
      4.0000000000000000e+00       2.5245761871337891e+00
      5.0000000000000000e+00       2.5245802402496338e+00
      6.0000000000000000e+00       2.5245783329010010e+00
      7.0000000000000000e+00       2.5245840549468994e+00
      8.0000000000000000e+00       2.5245933532714844e+00
      9.0000000000000000e+00       2.5245997905731201e+00
```

Figure 3.11 : Example of CTS output file.

Operation recordings are made using a CAELIB utility called CTS [CAE, 1993] that allows parameters of the simulation to be read and saved in a log file along with a time stamp, at regular time intervals while a manoeuver is performed on MOTS. Figure 3.11 shows an example of file produced by CTS where the leftmost column contains the time stamps representing the simulation first 9 seconds and the rightmost column contains the shoulder roll SSRMS joint angle values in radians at each time interval. For VOTE, parameters such as the SSRMS joints angles and POR position and orientation are recorded at one second intervals. Each parameter versus time data are saved in separate files, which are then merged together to produce a single data file containing all the parameters, grouped by time stamp.

Upon launching of a VOTE session, the simulation module loads the data file selected by the instructor and awaits further commands. When simulation commands such as PLAY, FFW or REW are received, the simulation module enters a loop in which simulation data is read from the data file, interpolated, and provided to both the instructor GUI and the rendering engine. Data has to be interpolated by the simulation since the sample rate used in building the file is lower than the anticipated rate at which the rendering engine will update the SSRMS position. Upon the receipt of a JUMP command and jump time, the simulation module finds the data group having the proper time stamp and computes interpolated data using those values.

As previously mentioned in chapter two, the SSRMS can operate from different base locations on the ISS or even travel along the ISS thrust using the MT. In order to properly locate the arm base in time, the base location is also included within the simulation data. The effect of the change of base location, as well as pedipulation on the computation of the POR in the VE "world coordinates" will be addressed in chapter four.

Since the data provided by the simulation module is used to move the SSRMS joints in the VE, the control of the graphical simulation is thus achieved using the simulation module commands.

3.6 Training with VOTE

The first step in using VOTE to conduct training will be the development of a detailed manoeuver sequence and associated checklist to be used as a training scenario. This sequence will then be performed by an instructor or an experienced astronaut on MOTS and the needed parameters recorded using CTS.

VOTE will then be used to provide the trainee with an immersive overview of the task. The trainee will be wearing the HMD while the instructor will be sitting in front of the console on which the VOTE Instructor GUI will be displayed and accessed. Using the replay functions together with the appropriate Virtual Tools, the instructor will guide the trainee through the operation and point out the difficulties and critical manoeuvers to the trainee. The trainee will then be asked to perform the manoeuver on MOTS. Again, the needed parameters will be recorded using CTS.

Following the completion of the scenario on MOTS, the instructor will be able to conduct a debriefing session with the trainee, now using his/her own manoeuver data for driving the MSS animation. The trainee's manoeuver can then be replayed, seen by the trainee from any location to clearly visualize arm clearance, and other information that was missing from the camera view provided on MOTS. The virtual tools will again be available and could be used to further explain the manoeuver difficulty and trainee errors if the need arises.

3.7 Using VOTE for Operation Procedures Development and Validation.

As a visualization tool, VOTE will be used to help operations people in the development and validation of robotic procedures to be used for the ISS assembly and maintenance. A typical robotic operation conducted using the MSS will be the retrieval and docking to the ISS of a module taken into orbit by the Space Shuttle. A first draft of the procedure will be tried on MOTS and recorded, and then viewed in VOTE, allowing clearance between

the module and the ISS to be assessed by an engineer navigating around the ISS in the VE. As the procedure is reviewed and modified, the availability of camera views for the different steps of the manoeuver will be studied using the camera view and FOV volume representation. This will allow engineers to choose the camera and camera parameters (pan and tilt angles, zoom and iris factors) which provide the best viewing angle and the required depth of field. This information will then be included into the written procedure that is to be followed in orbit by the MSS operator.

Chapter 4 Implementation, Results and Future Work

This chapter describes the implementation and integration of the modules comprising VOTE. This is followed by a presentation of performance results and anticipated future work.

4.1 VOTE System Implementation

4.1.1 VOTE System design overview

As previously discussed in Chapter 3, VOTE is divided into four functional modules which exchange data and receive commands. Since VOTE is being used within a UNIX environment, implementing the functional VOTE modules as independent processes running concurrently was the next logical step.

Thus VOTE was designed as a multi-process application using inter-module data flow. This in turn called for a way to provide inter-process communications. Shared memory proved to be a simple and efficient way to provide such communications.

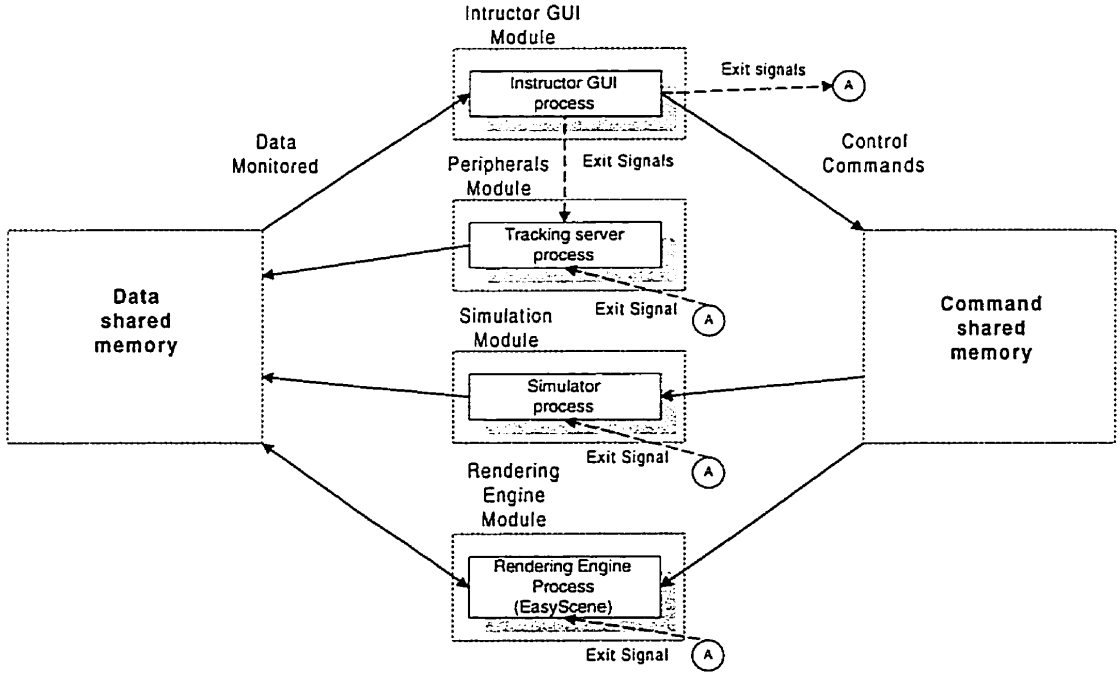


Figure 4.1 : VOTE architecture uses multi-processes and shared memory.

Figure 4.1 shows how the VOTE modules are implemented as independent processes and how inter-module communication is handled using shared memory. Commands and data are stored into two separate shared memory blocks, which allow read/write access of each process to be restricted, thus preventing accidental command or data overwriting by the processes.

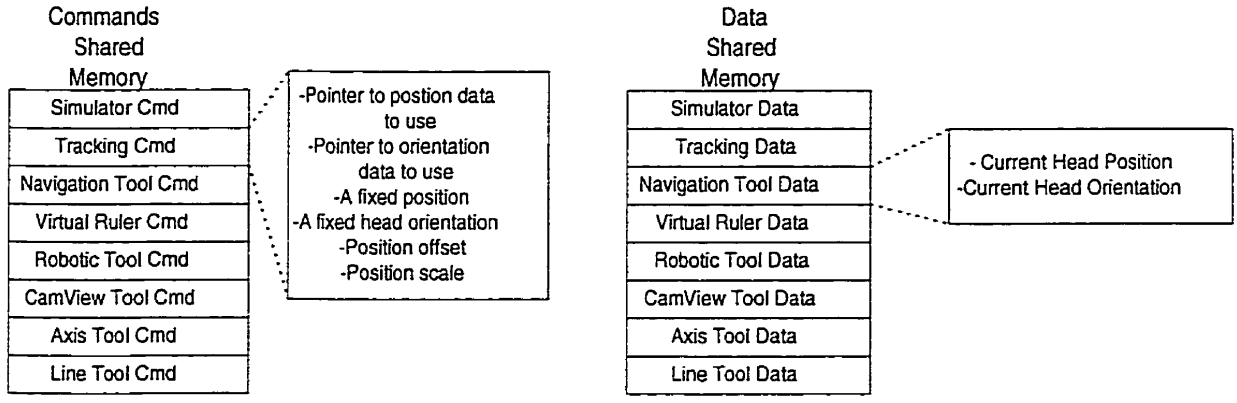


Figure 4.2 : Command and data shared memory structures.

Both shared memories (command and data) are defined as data structures. The command shared memory structure is divided into sub-structures, each storing command parameters used by a virtual tool, a peripheral, or the simulation, as shown in figure 4.2. The data shared memory is divided using the same sub-structures tree, but data instead of commands are stored. Both command and data shared memories are created by the Instructor GUI upon launching of a VOTE session, as will be discussed later.

Upon startup, each process needs to attach itself to both command and data shared memory in order to receive commands, read other process outputs and provide its own data to the other processes. Each process gets attached to the shared memory segments using the *shared memory id* of both command and data memory, which are stored in a file upon shared memory creation by the Instructor GUI. Shared memory read and write synchronization is achieved using record locking [Stevens, 1993].

Upon startup, each process (except the Instructor GUI) writes the processes id number it was given by the operating system into the data shared memory. This allows the Instructor GUI program to send *signals* to each process of VOTE when necessary to exit. Within the UNIX environment, signals are a technique used to notify a process that some condition has occurred (here for example the user has asked for VOTE termination) [Stevens, 1993]. A procedure to be called upon the receipt of a given signal is defined in each of the VOTE process, and the signal-procedure association is done once within each process at initialization. Upon receipt of an exit signal, each process goes through a terminating sequence in which it first detaches from shared memory and then terminates execution. The Instructor GUI program then frees the shared memory segments and exits.

The Instructor GUI, tracking server, and simulation server processes are allocated among the machine CPUs by the UNIX operating system. The rendering engine module (EasyScene) application, culling and drawing processes are allocated to specific a CPU by using a set-up command in an initialization file. In the current implementation which runs on two CPUs, the application and culling processes are assigned to one CPU, while the drawing process runs on the other one. The EasyScene processes are locked into memory automatically, thus preventing them from being swapped to disk, an operation that would slow down the execution. The other VOTE 's processes are also locked into memory using the *plock* function, which is called by each process at the beginning of execution.

4.1.2 Software Engineering Issues

Implementing VOTE as a multi-process application has the following advantages:

1- Permit each module to be implemented and debugged independently.
2- Permit off-loading time consuming computations needed to provide the simulation data.
3- Allows easier future connection to the MOTS simulation database.
4- Provides continuous monitoring of the peripherals.
5- Makes the Instructor GUI easy to integrate in VOTE.

Each module being an independently running program, the testing and debugging was easier to perform. Each module was tested by hardwiring the needed command and data in both command and data shared memories upon startup, and printing on screen the numerical outputs, or displaying the graphical results. Mutiprocessing also maintains modularity at the application level.

Implementing the simulation module as a process separate from the rendering engine instead of inserting it as a procedure in the rendering cycle allows the *application* phase of the cycle to be kept simple, and the update rate to be kept higher. It also provides an easily modifyable entry point for MOTS simulation data for future on-line operation visualization using VOTE.

Having an independent process running as server for the tracking peripheral make possible easier monitoring of the communication port. Low level data formatting can also be implemented at the server level, such as coordinate transformations, thus providing ready to use values in the data shared memory.

The Instructor GUI is an event driven program. Implementing it as a separate process which writes commands into shared memory makes it easy to integrate the GUI with the rest of VOTE.

The use of shared memory proves to be simple, and is the fastest way to communicate between processes running on the same machine [Silicon Graphics Inc., 1994a].

In order to support future improvement and maintainability, a coding standard based on ANSI C standard was established [Allard, 1997a] at the beginning of the project and followed throughout the implementation. The coding standard addresses issues such as variable naming convention, function headers and use of comments, all of which make the code intelligible and uniform from one module to the other.

4.1.3 Hardware used

The VOTE host computer is an ONYX Reality Engine 2, equipped with 2 R4400 CPU, one graphic pipe and one Multi-Channel Option (MCO). The MCO allows a 1280 x 1024 video buffer with 24 bits per pixels (8 bits RGB) to be divided to provides two RGB channels (one for each eye) to a FS5 Scan Converter (Virtual Research Inc.) to which a FS5 CRT HMD is connected. The HMD includes two CRT screens providing an image resolution of 640 by 486 pixels supported by optics providing a binocular FOV of 66 degrees horizontal and 33 degrees vertical, with 50% image overlap.

A RGB to NTSC converter (model CV-223 from Harmonic Research) is fed from the left eye RGB channel to display the trainee's view to the instructor on a regular TV screen.

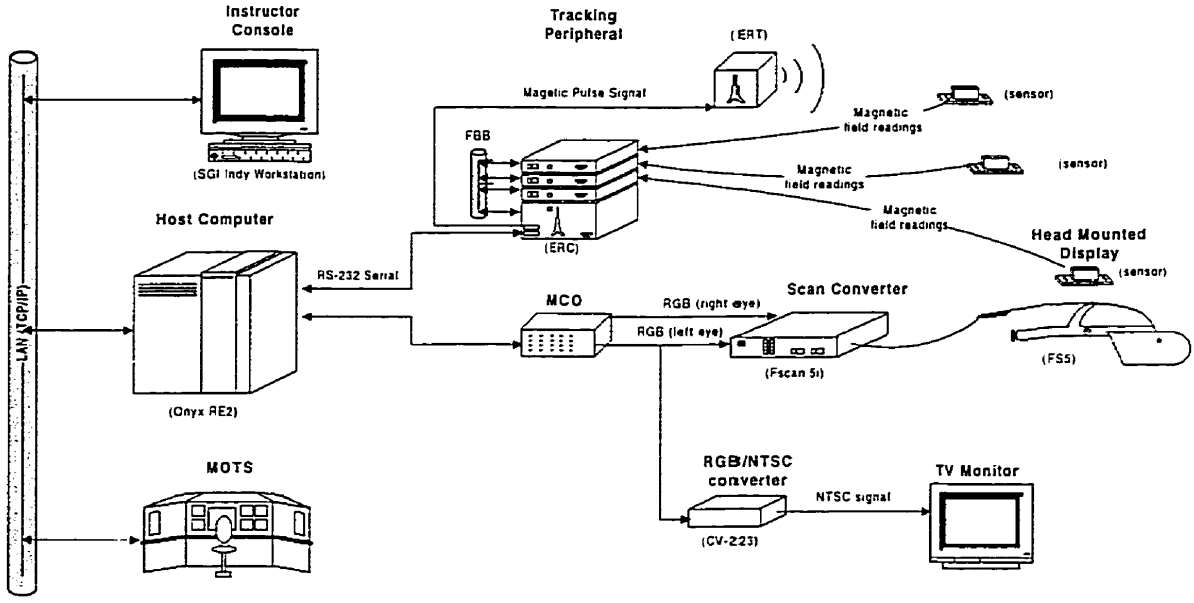


Figure 4.3 : VOTE hardware configuration.

Three Flock of Bird receiver units, each connected via wires to one 6 DOF sensor, are connected as slaves to an Extended Range Controller (ERC) via a Fast Bird Bus (FBB). The ERC is connected to the host computer via a RS232 serial line, and drive an Extended Range Transmitter (ERT) used to send the magnetic pulse used by the Flock of Bird receiver sensors. The sensor of the first receiver unit is mounted on the HMD

structure to provide tracking of the head movement, while the other two are left unused in the current configuration. The configuration of VOTE hardware is shown in figure 4.3.

An SGI Indy workstation, linked to the host computer via the local network, is used as the instructor console to provide display and access to the Instructor GUI using a mouse and keyboard. In the completed system, MOTS will be connected to VOTE via the network.

4.1.4 The Rendering Engine Module

As previously described in section 3.3, the rendering module is implemented using EasyScene API. The static 3D model of the ISS has been imported from MOTS, which uses the same file format as EasyScene. This saved many hours of work and avoided potential problems with file format conversion.

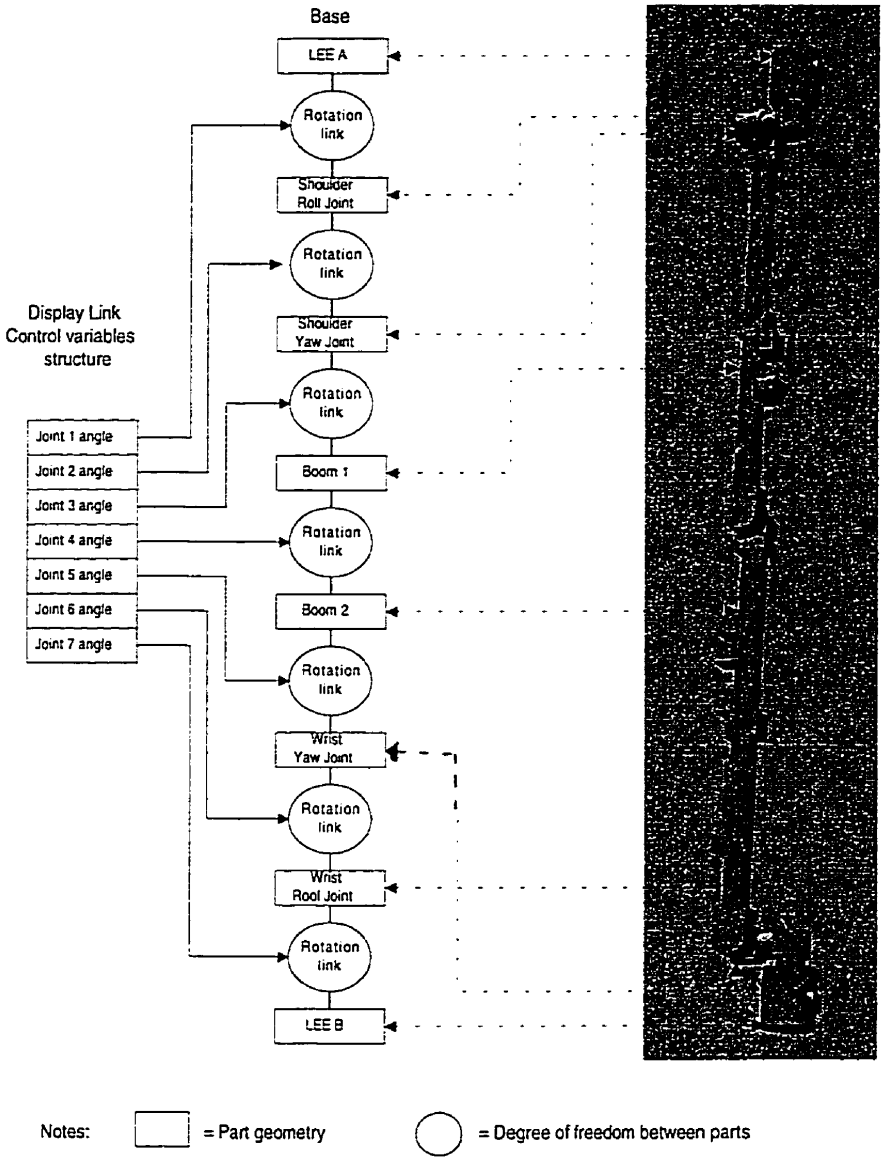


Figure 4.4 : The SSRMS model hierarchy, displays links, control variables and associated 3D model.

The models built for the virtual tools are constructed from primitives using the Designer's Workbench 3D editor. Once the object hierarchy has been completed, the Link Editor within Designer's Workbench is used to define *display links* to objects and between (relational link) objects composing a model [Coryphaeus, 1995]. Scaling, rotation, and translation are examples of display links associated to relationships between objects within a model used for the Virtual Tools in VOTE, while color and scales are examples of display links associated to objects. Each of the display links is mapped to a variable

that is used to control the behavior of the link, such as the angle of rotation, the scale factor, or the translation displacement value. These variables can in turn be mapped to external variables found in a user-defined data structure. Figure 4.4 shows the hierarchical tree, display link and associated 3D model used for providing an animated representation of the SSRMS within VOTE.

Once such a model and display link information has been saved in a graphical model and loaded within EasyScene, API functions are used within a user-defined module to attach a locally defined data structure to the one of the model to be animated. From that point, any change of variable value within the local structure controls the display link defined in the model.

A *user module* is a library of procedures and functions written in C that can be included and used by EasyScene at runtime. For each Virtual Tool, a user module is defined. This user module is then compiled together with EasyScene, thus creating a custom version of the rendering engine. Within VOTE, each user module contains the following :

1- An *initialization procedure,* where the tool model is loaded and default parameters set.
2- A *tool update procedure*, where the tool behavior algorithm is implemented.
3- *Locally defined functions* supporting the tool behavior algorithm.

The *initialization procedure* is executed once by EasyScene upon startup, while the tool update procedure is inserted in the rendering cycle and is executed once for each display update (or complete rendering cycle).

The *tool update procedure* first reads the command status of is associated tool, perform the needed behavior computation, writes the proper values in the local structure attached to the model control variables used for the display links and outputs its parameters (such

as the length for the ruler tool) to the data shared memory to allow tool monitoring by the Instructor GUI.

The *locally defined* functions provide the update procedure with the needed "mathematical support". The functions used to compute the far and near limits of a camera depth of field using the zoom factor and iris aperture are examples of locally defined functions used within the CamView module.

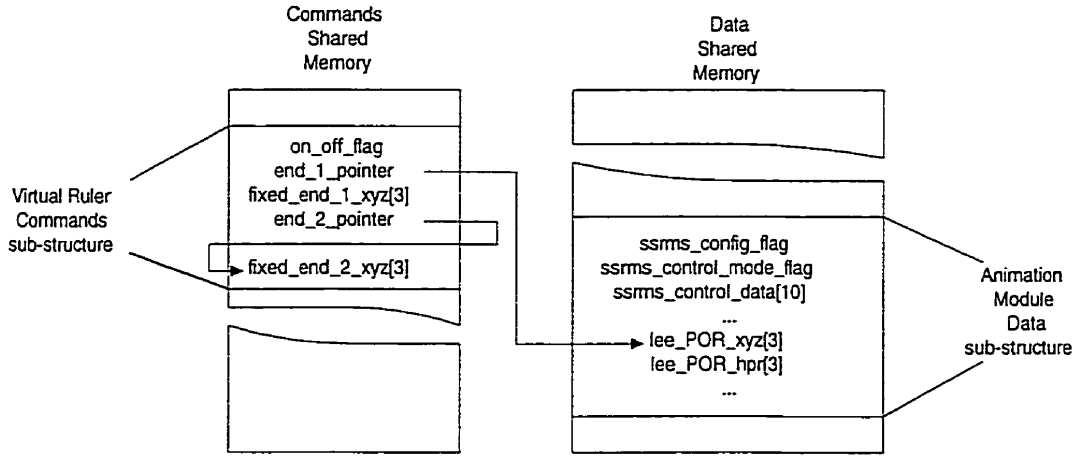


Figure 4.5 : Using pointers to attach the virtual ruler's ends to objects.

Some of the Virtual Tools implemented in VOTE such as the virtual ruler's end or the line Tool, require to be *attached* to a particular location. In order to achieve such attachments, pointers to values instead of values are passed, via the command shared memory, to the Virtual Tools algorithms. For example, if a pointer to the POR position vector computed by the simulation server is passed to the virtual ruler for its end #1, that end will then take the POR position at each frame. This end will then be attached to the end of the SSRMS. Giving the other ruler end (end #2) a fixed position is achieved by passing a pointer which points to a vector in the command shared memory sub-structure associated with the virtual ruler. This strategy is illustrated in figure 4.5. In the same way, the position of the user head could be assigned the POR position instead of the tracking

system data by simply changing the pointer of the head position. This strategy has been used on all of the Virtual Tools.

4.1.5 The Peripherals Module

As mentioned earlier, the only peripheral used in the current implementation is a tracking system. The peripheral hardware used is a Flock of Bird tracking system from Ascension Technology Corporation. The hardware configuration used is a master/slaves that uses a the *Fast Bird Bus* (FBB) to communicate. Master and slave configurations are set on the units using jumpers and dip switches. Figure 4.6 illustrates the hardware configuration used with the tracking system.

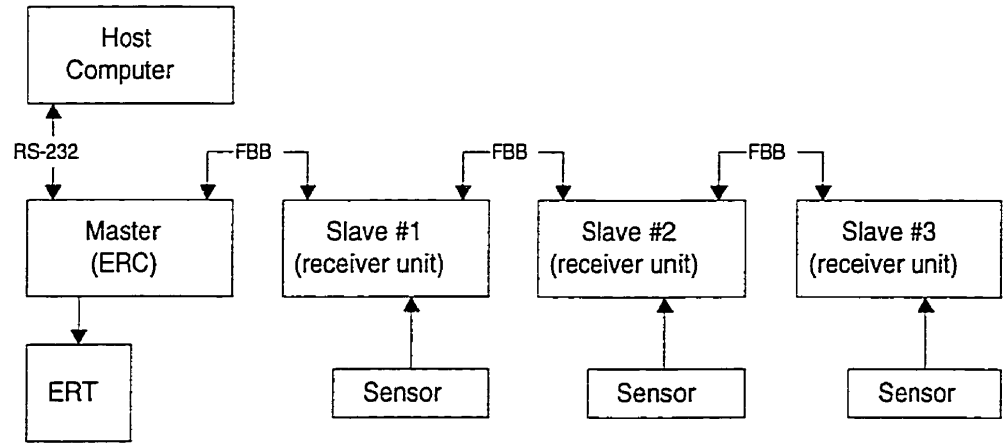


Figure 4.6 : The Flock of Bird hardware configuration.

The master is connected to the host computer via a RS-232 serial line on which commands and data are sent at a maximum rate of 38400 bauds due to limitation of the Onyx serial port hardware. This allow a maximum of 48 position and angle measurements per second per sensor [Ascension, 1996] to be sent to the host computer. This is higher than the anticipated maximum update rate (30 Hz), and thus is not a limitation for the current three sensor configuration.

The tracking server is the only process within the Peripheral module as of now. This process first initializes the tracking system by sending setup commands and parameters to the master unit. These commands set the tracking system configuration (1 master, 3 slaves, 1 ERT) and puts the system into group mode to allow all of the receiver units to receive and execute commands, and allows their output data to be relayed to the host computer via the master unit. The system is then set into *stream mode*. This mode makes the receivers send their output data one after the other continuously, without any polling from the host computer. The format of the sensor data contains both positions (X,Y,Z coordinates) and orientation (Yaw, Pitch, Roll angles) which are sent to the host computer as a set of 12 bytes for each sensor sample (6 words representing coordinates and angles). These bytes are received, parsed and merged to form words. A scale factor is then applied to the coordinate values, and the sensor positions are finally written into the data shared memory by the tracking server process.

While the tracking server process provides low level tracking operations, the *Teleport Tool* within the rendering engine allows the coupling of the sensor position with the generated channel view in EasyScene. This tool has no graphical representation in the VE, but allows trainee navigation, navigation offsets and displacement scaling to be adjusted by the instructor. This allows the trainee head position and displacement in the laboratory to be mapped to the needed position and displacement in the VE.

As briefly described in Chapter 3, two channels are attached to a graphical object representing the user's head in the VE, and are used to model stereoscopic viewing, as shown in figure 4.7.

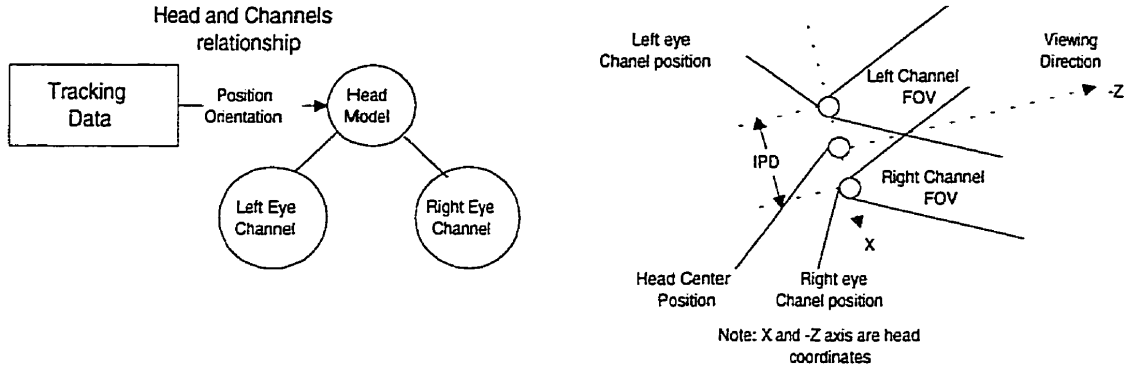


Figure 4.7: The stereoscopic viewing and navigation implementation in the VE.

By translating and rotating the head model within EasyScene to reflect the trainee's position and orientation computed using the tracking system data, the proper stereoscopic view of the world is generated. Adjusting the offsets of the viewpoint's position of each channel relative to the head model center allows the IPD to be adjusted for each user within the simulation.

4.1.6 The Simulation server

The Simulation Server provides the data needed by the SSRMS model's display links to animate the arm using recorded operations performed on MOTS. The Simulation Server also allows control over the simulation by using commands issued by the Instructor GUI through the command shared memory. Figure 4.8 illustrates how the Simulation Server monitors commands and provides interpolated MSS operation data.

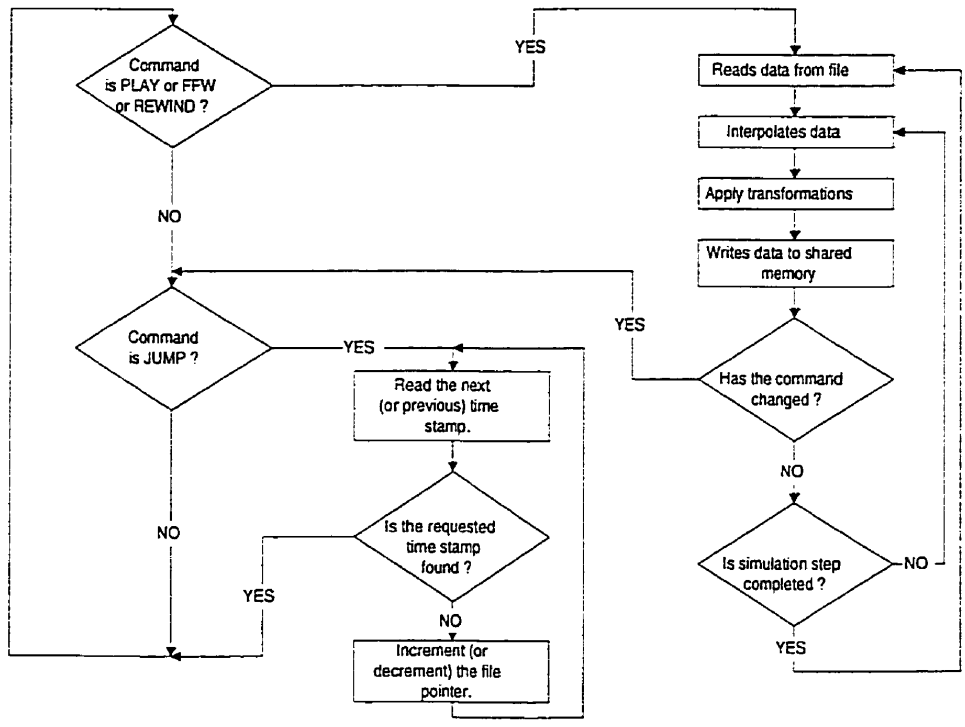


Figure 4.8 : The Simulation Server process.

The command variables are monitored in a loop until a command other then STOP is read. Upon receiving a PLAY, FAST FORWARD or REWIND command, the Simulator enters a second loop in which data is read, interpolated with respect to time, transformations applied (e.g. coordinate frame transformations) and the interpolated data is finally written into the shared memory. This cycle repeats until a new command is received. Upon receipt of a JUMP command, a procedure that uses the current and requested elapsed time, searches the data file containing the operation data. The search ends when the file time stamp which is the closest to the requested elapsed time is found. The simulation interpolation then resumes, using the time stamp found as the interpolation new starting point.

The graphical model of the SSRMS uses a hierarchy that allows each of the arm's components to "follow" the component to which it is attached, thus making the arm "stay together" as the joints are moved. This latter feature means the simulation server process only needs to provide the seven joint angles values to place the LEE at the needed

position and orientation. This is possible since the forward kinematics of the arm are "hidden" in the definition and relative position of the pivot points within the model, definitions that match the ones of the kinematic model used by MOTS.

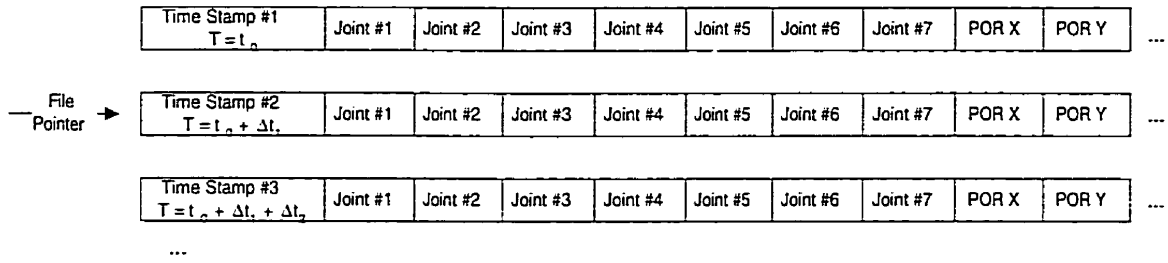


Figure 4.9 : Simulation file data structure.

The sample rate at which the operation data is recorded on MOTS makes it necessary for the simulation server to provide interpolated data, as explained in Chapter 3. The operation data is stored in a file as a sequence of identical structures each representing the values of all the recorded parameters at a given time stamp, as shown in figure 4.9. When in the PLAY, FFW or REWIND mode, two sets of data are used to carry out the interpolation: the start position and the end position. A file pointer points to the start position, and the data set following (for PLAY and FFW) or preceding (for REWIND) is accessed to get the end position data set. These position data are then used many times within the interpolation loop to generate joint and POR values for the time interval in between the two positions, here called a *simulation step*. When the time interval separating the two data sets (the difference between the time stamps) has elapsed, the file pointer is incremented (or decremented for REWIND) to the adjacent data set. This data set becomes the new start position, and the interpolation cycle starts again.

In order to provide a simulation replay that is a true representation of the speed of the operation recorded, the interpolation must be synchronized with a stable time reference. Using the host computer system time as the interpolation variable allows the simulation to closely approximate the state of the MOTS simulation at a particular time, without having the simulation speed vary when the system load increases or the simulation loop

cycle period changes. At each simulation cycle, the current system time is compared to current step's planned completion time, and data interpolation is performed using the ratio of the remaining time to completion over total step time. Such an implementation allows the simulation server to readjust its interpolation factor to compensate for a non constant cycle period. Modifying the total step time using a scale factor within the interpolation allows accelerating (by decreasing the step time) or slowing down (by increasing the step time) the animation.

Interpolating the joint angle values and POR positions in a linear fashion is not a true representation of the SSRMS movement (i.e. the POR coordinates do not necessarily vary linearly in time between two POR positions), but given the relatively short time interval separating the samples and the slow movement of the arm, it has been found to provide an acceptable animation.

As far as the EasyScene rendering engine is concerned, the SSRMS model is a single object whose position and orientation can be controlled, but for which internal components of the hierarchy cannot be accessed. EasyScene can properly render all of the arm components, but the position and orientation of individual components cannot be accessed using the API functions. This leads to the following problem : How can a payload be attached to the LEE to be carried around by the SSRMS ? This leads to the need to provide the position and orientation of the LEE in the "world reference frame" of EasyScene. By doing so, a payload position and orientation can be updated at each frame, making it appear to be physically attached to the LEE as it moves with it.

While MOTS provides the POR position as part of its simulation, such coordinates are given relative to two frames of reference simultaneously : the display frame selected by the operator and the SSRMS current reference frame. As mentioned in chapter two, there are many PDGFs which can be used as a base for the SSRMS, and each has its own frame of reference. In order to get the POR position and orientation in the world coordinates, coordinate frame transformations need to be applied to the POR provided by MOTS.

Each PDGF has an associated 4 by 4 transformation matrix that maps its frame of reference to the one of EasyScene. This transformation matrix is used by the simulation server to transform the coordinates of the POR from the SSRMS base to the EasyScene world coordinates. These coordinates can then be used within EasyScene to attach objects to the SSRMS LEE.

4.1.7 The Instructor GUI module

As the VOTE control program, the Instructor GUI process creates the command and data shared memories, and launches all VOTE' s processes in separate shells using system calls. The use of a separate shell for each process allows easier monitoring of the status and error messages that are printed on screen by the VOTE modules. The Instructor GUI is implemented using the Xmotif C compatible library which provides an object oriented framework to build a graphical user interface for XWindows environments. Buttons, sliders, text entry, and scrollable list are a few examples of the objects, here called *widgets*, available to construct an interface.

Xmotif is event driven. A *callback function* is associated to each widget defined within a window, and it is triggered upon a given action of the widget, such as being pushed (for a button). The callback function can be any user-defined C function, and can include XMotif callbacks as well.

Within the Instructor GUI, widgets callback functions are used to write to VOTE command shared memory. Upon selection of a tool from the Virtual Tool pulldown menu, a window containing the needed widgets is created, and the widget initial state (such as button pushed or not) is set using the command parameters stored in shared memory. On a button activation, an item selection, a slider movement or a text entry, a callback function is triggered. This function formats and writes the proper command parameters in the command shared memory. As the tool update procedure previously

mentioned is executed in the next rendering cycle, the effect of the command on the Virtual Tool is seen in the VE.

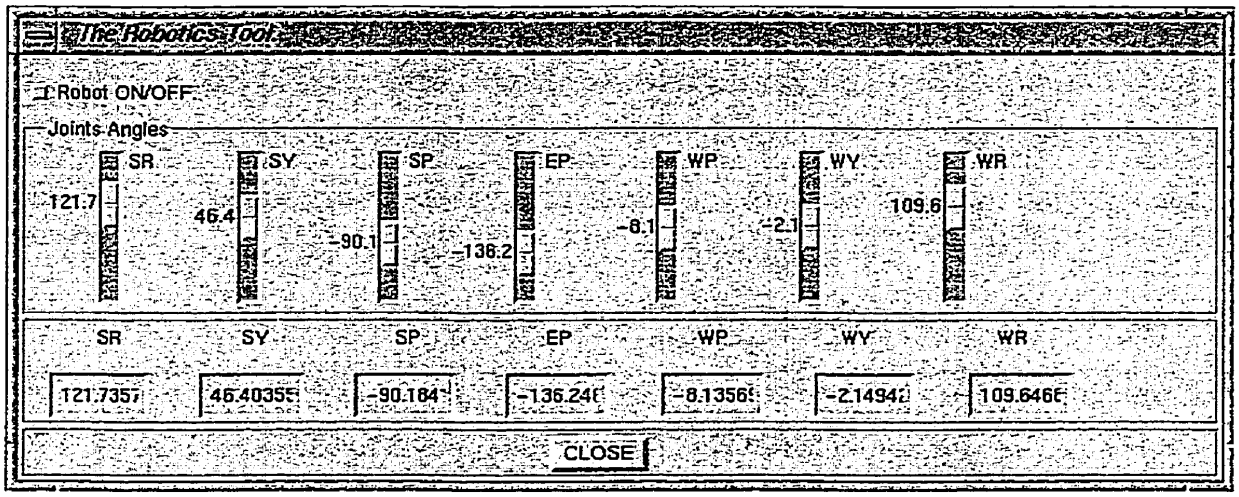


Figure 4.10 : The SSRMS Joints values updates in the Instructor GUI Robotic Tool window.

Continuous monitoring and display of parameters found in the data shared memory is achieved using XMotif *timers*. Timers allow callback functions to be automatically triggered at regular time intervals. In turn, these callback functions are used to update widget parameters, such as the position of the slider showing the simulation elapsed time on the simulation control window shown in figure 3.8, or the SSRMS joint angle sliders and text outputs shown in figure 4.10

4.2 Evaluation Procedures

4.2.1 Virtual Tools Processing Times and Rendering Update Rates.

The time required to process the Virtual Tools behavior within the rendering loop directly impacts the update rate of the visual simulation. Being able to evaluate to what extent including a Virtual Tool within VOTE reduces the frame rate is needed to make the

instructor aware of the necessary tradeoff between update rate and the Virtual Tools being used.

While the update procedures are usually executed once per rendering cycle, in evaluating their performance they were modified to allow continuous execution for 1000 cycles. In order to evaluate the time needed for each of the Virtual Tools to be processed within the EasyScene rendering loop, the host computer system time was sampled at the beginning of the update procedure (see section 4.1.4) and again at the end of the same procedure after the 1000 cycles. The difference between the two samples divided by the number of rendering cycles gives the average time required to go through the virtual tool algorithm.

Virtual Tools	Function Performed	Processing Time (ms) [1]
Line Tool	Update, no point added	0.10 ± 0.01
	Update, points added	0.40 ± 0.01
Axis Tool	Update only	0.10 ± 0.01
	Adding set of axes	0.12 ± 0.01
	Removing set of axes	0.14 ± 0.01
Camview Tool	Update 1 FOV	0.78 ± 0.01
	Update 2 FOV	0.98 ± 0.01
	Update 3 FOV	1.08 ± 0.01
Handcontrollers Tool	THC update	0.02 ± 0.01
	RHC update	0.01 ± 0.01
	THC and RHC update	0.02 ± 0.01 [2]

Notes:

1- The time resolution achieved by the system time function is 10 milliseconds (ms) in the worst case. For all the execution times presented, the error is thus ± 0.01 ms.

2- Processing both handcontrollers is not much more time consuming than processing only one since most of the computations are common to both.

Table 4.1 : Processing times for the Virtual Tools.

Table 4.1 gives measured processing times for some of the Virtual Tools. These measurements were taken when VOTE was fully loaded and running. For some Virtual Tools, processing times for a particular tool operation are given. This is the case for the Line Tool (i.e. when a point is added to the line) and the Axis Tool (when an axis is added and removed from the list).

Ruler's Length, L (m)	Number of Tick Marks	Processing Time (ms)
$0.0 \leq L < 0.5$	1	3.10 ± 0.01
$0.5 \leq L < 1.0$	2	4.90 ± 0.01
$1.0 \leq L < 1.5$	3	6.60 ± 0.01
$1.5 \leq L < 2.0$	4	8.60 ± 0.01
$2.0 \leq L < 2.5$	5	10.50 ± 0.01
$7.0 \leq L < 7.5$	15	25.90 ± 0.01
$9.5 \leq L < 10.0$	20	34.10 ± 0.01

Table 4.2 : Processing for the Virtual Ruler as a function of ruler length.

Table 4.2 presents the processing times for the Virtual Ruler of different lengths. Here the processing time depends on the number of tick marks that have to be displayed on the virtual ruler, which in turn depends on the ruler's length (see section 3.1.2).

The effects of both the 3D model complexity and Virtual Tools in use lead to variable display update rates that range from 6 Hz in a worst case scenario (i.e. all the Virtual Tools used at the same time, with over 5700 fully textured polygons in streoscopic view) to 30 Hz (no Virtual tool used, and less than 100 polygons). For a typical scenario involving the simultaneous use of two or three Virtual Tools and a user FOV limited to one or two modules of the station, display update rates of 12 Hz to 15Hz are achieved. In the worst case scenario, the processing of the Virtual Tools used about 36 % of the

processing time while the culling and drawing of the polygons composing the scene accounted for the balance.

4.2.2 Tracking Update Rate.

The tracking server process update speed has also been evaluated using the time required to perform 1000 measurement loops. Within each loop, the position (i.e. X,Y,Z coordinates) and orientation (i.e. Yaw, Pitch and Roll angles) were measured for the 3 different sensors. The host computer system time was sampled before and after the 1000 cycles, and the difference computed.

Average time needed for 1000 cycles (s)	Average time per measurement cycle. (s/cycle)	Average data update rate per sensor (cycle/s)
22.26 ± 0.01	0.00223	44.8

Table 4.3 : Tracking server process sensor position and orientation update rate.

Table 4.3 shows the results obtained for the tracking server process. These measurements were taken when VOTE was fully loaded and running.

4.2.3 Simulation Server Update Rate.

To evaluate the tracking server process, the simulation server update speed has been observed using the time required to perform 1000 update loops. Within each loop, the seven joint angles were interpolated, and the position (i.e. X,Y,Z coordinates) and orientation (i.e. Yaw, Pitch and Roll angles) of the POR were interpolated and transformed into the world coordinates. Again, the host computer system time was sampled before and after the 1000 cycles, and the difference computed.

Average time needed for 1000 cycles (s)	Average time for a complete simulation data set update. (s)	Average simulation data set update rate. (update/s)
0.27 ± 0.01	0.00027	3703

Table 4.4 : Simulation server update rate.

Table 4.4 shows the results obtained for the simulation server process. These measurements were taken when VOTE was fully loaded and running. These update rates will further be discussed in section 4.2.5.

4.2.4 User Evaluations

The effectiveness of the Virtual Tools has not been assessed in a quantitative fashion since VOTE has not yet been used within MSS training at CSA. However, a few astronauts were presented with some of the Virtual Tools either on site using the HMD with tracking, or at Johnson Space Center using a short documentary video prepared for the MSS Training Group [Allard, 1997b]. Their written comments and "feelings" for each of the Virtual Tools where sent back to CSA via e-mail.

On site demonstrations were provided at different stages of development of VOTE, and led to modifications and the creation of some Virtual Tools. For example, a discussion with Canadian astronaut Chris Hadfield led to the addition of the FOV volume display to the CamView tool. The Axis Tool was modified to use color coded axes after having presented the tool to astronaut Marc Garneau. Following comments made by astronaut Julie Payette about the difficulty in relating a replayed manoeuver to the task timeline as performed on MOTS, the elapsed time slider control was added to the replay control. As a whole, the astronauts saw VOTE as being a very interesting new tool for training, and were looking forward to using the VE for training.

Feedback from the instructors that will use VOTE has been provided only at the very beginning of the VOTE implementation. This is due to the fact that the instructors are presently on training at NASA and have not been able to get a thorough demonstration of the interface and the latest Virtual Tools implemented.

4.2.5 Analysis of Results

The display refresh rate of about 12 Hz usually achieved on VOTE is lower than what is needed to completely avoid motion sickness. It has been observed that the users quickly learn to move their head more slowly to minimize the discomfort caused by the mismatch between their head motion and the displayed image. No nausea or headache problems have been reported by users of the current implementation of VOTE. However, considering the addition of future Virtual Tools, the refresh rate, if not yet presenting major problems, remains a concern. Future upgrade in the hardware is planned, and could decrease the processing time required by the culling and drawing phases.

The Virtual Tools do not usually represent a large amount of the rendering cycle time. The virtual ruler, for long lengths, can require appreciable processing time, and ways to optimize the ruler generation algorithm should be further investigated.

The tracking server process provides about 44 sensor data updates per second, close to the maximum data rate claimed by the manufacturer [Ascension, 1996] when using a RS-232 serial connection at 38 400 bauds. Given the 12 to 15 Hz display refresh rate, the tracking process refreshes the data more often than needed, and does not induce much time lag. In fact, each sensor is sampled approximately three times before its data is used by the rendering engine to modify the displayed graphics.

The simulation server process, just as the tracking server process, provides data at a much higher rate than the display refresh rate. Thus any discontinuity in the SSRMS

motion is due to the inability of the graphics to be generated fast enough, and not to the simulation data provided by the simulation server.

4.3 Future Work

In the future, the VOTE simulation server will be modified to access simulation data from MOTS shared memory in real-time. This enhancement will allow a manoeuver being performed using the RWS controls to be displayed simultaneously in VOTE. Such functionality will allow the astronauts to better relate their hancontroller inputs to the SSRMS movements.

When MOTS will be fully operational, it will be able to use real-time telemetry data sent by the MSS on the ISS to animate the SSRMS within the simulation. As the SSRMS data will be stored and updated within the MOTS database, it could be used within VOTE simultaneously (given that VOTE has access to the MOTS CDB). This would allow 3D viewing of an operation, on the ground. This feature could provide the operation people with a useful tool to help the operator, on-line, if for example, clearances cannot be clearly seen using the ISS cameras.

The addition of a second HMD and associated tracking system would provide the capability to have two users within VOTE at the same time. This latter feature would allow the conduct of EVA-SSRMS joint operation training, much like the type performed by NASA for the Hubble Space Telescope repair described in chapter one. It would also allow a trainee debriefing to be conducted with both the instructor and the trainee in the VE.

Having the instructor and the trainee together in the VE would require an alternative way other than the GUI, for accessing the Virtual Tools functionality. The method currently planned would use voice activated commands to allow both trainee and instructor to access VOTE functions.

Optimization work could improve some of the Virtual Tools, such as the virtual ruler, to minimize the tool's processing time within the EasyScene rendering loop. The use of simplified 3D models could also be used to represent the ISS and the SSRSM if the refresh rate becomes a problem.

More detailed analysis of the process execution times for the entire VOTE system should be carried out to see if other optimizations are possible or necessary.

Other Virtual Tools would need to be implemented in order to help visualize SSRMS manoeuvers. A corridor showing a particular path to be followed and the maximum deviation allowable has been identified as a future improvement to VOTE. A prototype of a virtual tool using grids to allow better relative distance evaluation has been implemented but needs further modification and evaluation by astronauts before being integrated into VOTE.

The development of Virtual Tools using sounds to convey object proximity (i.e. a radar "ping" sound) or SSRMS joint motor torque, would be worth investigating. The integration of an "audio server" process providing audio spatialization capability to VOTE within the Peripheral Module, could support such enhancements.

Chapter 5 Conclusion

This thesis presented a Virtual Operations and Training Environment (VOTE) being developed for the training of astronauts on the Mobile Servicing System (MSS) at the Canadian Space Agency.

An overview of the fields of telerobotics and virtual reality (VR) technology were first presented, followed by a literature review on human factors and benefits associated with the use of VR for training. A description of the MSS and associated in-orbit operations in the International Space Station context was then given, followed by an overview of the MSS Operations and Training Simulator.

The description of VOTE's functionality at the system and module level was then presented. The details of VOTE implementation were also discussed. A description of the evaluation procedures and an analysis of results was also included. Some suggestions for future enhancements to the system were discussed.

The current implementation of VOTE is fully functional and the preliminary results are very satisfying. The system will be incorporated into the training given at the Canadian Space Agency scheduled for some 40 astronauts and 20 mission controllers to be given over the next 24 months.

Glossary

API	**A**pplication **P**rogramming **I**nterface
CDB	**C**ommon **D**ata **B**ase
CPU	**C**entral **P**rocessing **U**nit
CRT	**C**athode **R**ay **T**ube
CSA	**C**anadian **S**pace **A**gency
DOF	**D**egree **O**f **F**reedom
EVA	**E**xtra **V**ehicular **A**ctivity
GUI	**G**raphical **U**ser **I**nterface
ISS	**I**nternational **S**pace **S**tation
IPD	**I**nter-**P**upilary **D**istance
LCD	**L**iquid **C**rystal **D**isplay
LDA	**L**ogistic **D**eployment **A**ssembly
LEE	**L**atching **E**nd **E**ffector
MBS	**M**obile **B**ase **S**ystem
MOTS	**M**SS **O**perations and **T**raining **S**imulator
MSS	**M**obile **S**ervicing **S**ystem
MT	**M**obile **T**ransporter
NASA	**N**ational **A**eronautics and **S**pace **A**dministration
PDGF	**P**ower and **D**ata **G**rapple **F**ixture
POR	**P**oint **O**f **R**esolution
RWS	**R**obotics **W**ork **S**tation
SSRMS	**S**pace **S**tation **R**emote **M**anipulator **S**ystem
SVS	**S**pace **V**ision **S**ystem
VE	**V**irtual **E**nvironment
VOTE	**V**irtual **O**perations and **T**raining **E**nvironment
VR	**V**irtual **R**eality

References

[Allard, 1997a] P. Allard, Z. Joukakelian, "*MSS O&U Training Software Coding Standards*", MSS Training Group, Space Station Program, Saint-Hubert, QC, January 1997.

[Allard, 1997b] P. Allard, S. Gohier, E.P. Greenberg, Z. Joukakelian, G. Parent, C. Ruso, "*Space Station Program Office Training Demo Video*", prepared for June 1997 International Space Station Training Control Board, Houston, TX, by MSS Training Group, Space Station Program, Saint-Hubert, QC, June 1997.

[Adam, 1993], J.A. Adams, "*Virtual Reality is for Real*", IEE Spectrum Magazine, vol.30, no. 10, pp.22-29, October 1993.

[Applewhite, 1991] H. Applewhite, "*Design of acoustic ranging systems*", Technical Report 91-02, Piltdown Incorporated: Beavertown, Oregon, 1991.

[Ascension, 1996] Ascension Technology Corporation, "*The Flock of Bird: Installation and Operation Guide*", Ascension Technology Corporation, Burlington, Vermont, June 1996.

[Backes, 1993] P.G. Backes, M.K. Long, R.D. Steele, *"The modular telerobot task execution system for space telerobotics"*, Proceedings of 1993 IEEE International Conference on Robotics and Automation, Atlanta, GA, pp.524-530, 1993.

[Bailey, 1994], J.M. Bailey, B.G. Witmer, *"Learning and transfer of spatial knowledge in a virtual environment."*, Proceedings of Human Factors and Ergonomics Society 38th Annual Meeting, Nashville, TN, pp.1158-1162, 1994.

[Bliss, 1997] J.P. Bliss, M.A. Guest, *"The effectiveness of virtual reality for administrating spatial navigation training to firefighters."*, Presence: Teleoperators and Virtual Environments, vol. 6, no. 1, pp.73-86, February 1997.

[Brooks, 1992] T.L. Brooks, I. Ince, *"Operator vision aids for telerobotics assembly and servicing in space"*, Proceedings of 1992 IEEE International Conference on Robotics and Automation, Nice, France, pp.886-891, 1992.

[Burdea, 1993] G. Burdea, P. Coiffet, *"Virtual Reality Technology"*, John Wiley and Sons Inc., Paris, 400 pp.1993.

[CAE, 1996], CAE Electronics Ltd., "*MOTS System Specifications Document (SDD)*", doc. No. CD289520.01.8.300, Revision C, July 1996.

[CAE, 1993], CAE Electronics Ltd., "*CAELIB on UNIX Systems*", Release 15, Computer Systems Engineering Department, Saint-Laurent, QC, 1993.

[CAE, 1986], CAE Electronics Ltd., *"Introducing the visual display system that you wear"*, Saint-Laurent, QC, 1986.

[Caldwell, 1994] D.C. Caldwell, A.Wardle, M. Goodwin, *"Tele-presence: Visual, audio and tactile feedback and control of a twin armed mobile robot."*, Proceedings of 1994 IEEE International Conference on Robotics and Automation, San Diego, CA, pp.244-249, 1994.

[Cannon, 1997] D.J. Cannon, G. Thomas, *"Virtual tools for supervisory and collaborative control of robots"*, Presence: Teleoperators and Virtual Environments, vol. 6, no. 1, pp.1-27, February 1997.

[Cannon, 1994] D.J. Cannon, G. Thomas, C. Wang, T. Kesavadas, "*A virtual reality based point-and-direct robotic system with instrumented glove.*", International Journal of Industrial Engineering, vol. 1 no. 2, 139-148, 1994.

[Cater, 1995] J.P. Cater, S.D. Huffman, "*Use of the remote access virtual environment network (RAVEN) for coordinated IVA-EVA astronaut training and evaluation.*", Presence: Teleoperators and Virtual Environments, vol. 4, no. 2, pp.103-109, Spring 1995.

[Chung, 1989], J. Chung, M. Harris, F. Brooks, M.T. Kelly, J.W. Hughes, M. Ouh-Young, C. Cheung, R.L. Holloway, M. Pique, "*Exploring virtual worlds with head-mounted displays, non-holographic 3-dimensional display technologies*", SPIE Proceedings on Non-holographic 3-dimensionnal display technologies, Los Angeles, CA, pp.15-20. January 1989.

[Cooke, 1993] C. Cooke, S.A. Stansfield, *"Interactive graphical model building using Telepresence and virtual reality"*, Proceedings of 1993 IEEE International Conference on Robotics and Automation, part 2, Atlanta, GA, pp.1436-1440, 1993.

[Coryphaeus, 1996], Coryphaeus Software Inc., "*EasyScene Real-time Visual System*", Coryphaeus Software Inc., Los Gatos, CA, April 1996.

[Coryphaeus, 1995], Coryphaeus Software Inc., "*Designer's Workbench 3.1 Reference*", Coryphaeus Software Inc., Los Gatos, CA,1995.

[Delp, 1997] S.L. Delp, P. Loan, C. Basdogan, J.M. Rosen, "*Surgical Simulation: An Emerging Technology for Training in Emergency Medicine*", Presence: Teleoperators and Virtual Environments, vol. 6, no. 2, pp.147-159, April 1997.

[Division, 1993] Division Limited, "*Environment Simulation*", company brochure, Bristol, UK, 1 pp., 1993.

[Drascic, 1993] D. Drascic, J.J. Grodski "*Defense teleoperation and stereoscopic video*", Procedings SPIE vol. 1915, Stereoscopic Display and Applications IV., San Jose, CA, pp.58-69, February 1993.

[Drascic, 1991] D. Drascic, "*Skill acquisition and task performance in teleoperation using monoscopic and stereoscopic display.*", Proceedings of the Human Factors Society 35th Annual Meeting, San Francisco, CA, pp.1367-1371, 1991.

[Ferrara, 1996], S.A. Ferrara, "*PDRS Operation Checklist, STS-77 Flight Supplement*", NASA Mission Operations Directorate System Division, Lyndon B. Johnson Space Center, Houston, TX, February 1996.

[Foley, 1990] J. Foley, A. van Dam, S. Feiner, J. Hughes, "*Computer Graphics: Principles and Practices.*", Addison-Wesley Publishing Company, New York, pp.1175, 1990.

[Foster, 1992] S. Foster, "The Convolvotron Concept", Proceedings of the Cyberarts Conference, pp.93-95, Pasadena, CA, October 1992.

[Fuchs, 1989] H. Fuchs, J. Poulton, J. Eyles, T. Greer, J. Goldfeather, D. Ellsworth, S. Molnar, G. Turk, B. Tebbs, L. Israel, "*Pixel-Plane 5: A Heterogenous Multiprocessor Graphics Systems Using Processor-Enhanced Memories*", Computer Graphics, vol. 23, no. 3, pp.79-88, July 1989.

[Funkhouser, 1992] T. Funkhouser, C. Sequin, S. Teller, "*Management of Large Amounts of Data in Interactive Building Walkthroughs*", Proceedings of the 1992 Symposium on Interactive 3D Graphics, ACM, Cambridge, MA, pp.11-20, 1992.

[Hart, 1987] D. Hart, "*The Encyclopedia of Soviet Spacecraft*", Exeter Books, New York, pp. 63-69, 1987.

[Held, 1991] R.M. Held, N.I. Durlach, "Pictorial communication in virtual and real environments", Philadelphia, Taylor & Francis, pp.233-245, 1991.

[Hendrix, 1996] C. Hendrix, W. Barfield, *"Presence within virtual environments as a function of visual display parameters."*, Presence: Teleoperators and Virtual Environments, vol. 5, no. 3, pp.274–289, Summer 1996.

[Hochberg, 1986] J. Hochberg. "The representation of motion and space in video and cinematic displays" in Handbook of Perception and Human Performance, vol. 1, John Wiley and Sons, New York, 1986.

[Holloway, 1993] R. Holloway, A. Lastra, "Virtual Environment: A survey of the Technology", TR93-033, University of North Carolina at Chapel Hill, 40 pp., September 1993.

[Immega, 1995] G. Immega, K. Antonelli, J. Ko, *"Teleoperation of the KSI tentacle manipulator for hot cell decontamination"*, Proceedings of 1995 IEEE International Conference on Robotics and Automation, Nagoya, Japan, pp.2133-2136, 1995.

[Kalawsky, 1993] R.S Kalawsky, *"The science of virtual reality and virtual environments"*, Addison-Wesley Publishing Company, 405 pp., 1993.

[Kim, 1993] W.S. Kim, *"Graphical operator interface for space telerobotics"*, Proceedings of 1993 IEEE International Conference on Robotics and Automation, Atlanta, GA, pp.761-768, 1993.

[Kreuger, 1991] M. Kreuger, "Virtual Reality II", Addision-Wesley Publishing Company, 286 pp. 1993.

[Latham, 1993] R. Latham, *"What Isn't new in VR"*, Proceedings of VR Systems'93 Conference, New York, pp.225-258, March 1993.

[Levine, 1985] M.D. Levine, *"Vision in man and machine"* McGraw-Hill, New York, 574 pp., 1985.

[Loftin, 1994] R.B. Loftin et al., *"Virtual environments in training: NASA`s Hubble Space Telescope Mission."* Proceedings of 1994 Interservice/Industry Training Systems & Education Conference, Houston, TX, 1994.

[Logan, 1995] A. Logan, *"Training Beyond Reality"*, Proceedings of the 1995 CASI Conference, Montreal, QC, 1995.

[McDowall, 1990] I. McDowall, M. Bolas, S Pieper, S. Fisher, J. Humphries, *"Implementation and integration of a counterbalanced CRT-based stereoscopic display for interactive viewpoint control in virtual environment applications"*, Proceedings of Stereoscopic Display and Applications, Santa Clara, CA, SPIE vol. 1256, 1990.

[McGreevy, 1993] M. W. McGreevy , *"Virtual Reality and Planetary Exploration"*. In. A. Wexelblat , Virtual Reality: Applications and Explorations, Academic Press, Cambridge, MA, pp. 163-197. 1993.

[McKenna, 1992] M. McKenna, D. Zeltzer, *"Three dimensionnal visual display systems for virtual environments."*, Presence: Teleoperators and Virtual Environments, vol. 1, no. 4, pp.421-457, Fall 1992.

[Meyer, 1992] K. Meyer, H. Applewhite, F. Biocca, "A survey of Position Trackers", Piltdown Technical Report 91-05, Piltdown Inc., February 1992.

[Milgram, 1995] P. Milgram, A. Rastogi, J.J. Grodski, *"Telerobotic control using augmented reality"*, Proceedings of the 4th IEEE International Workshop on Robot and Human Communication, Tokyo, Japan, July 1995.

[Miner, 1995] N.E. Miner, S.A. Stansfield, *"An interactive virtual reality simulation system for robot control and operator training"*, Proceedings of 1995 IEEE International Conference on Robotics and Automation, Nagoya, Japan, pp.1428-1435, 1995.

[Pimentel, 1993], K. Pimentel, K. Teixeira, "Virtual Reality: Through the New Looking Glass", Windcrest McGraw-Hill, New York, 301 pp., 1993.

[Pook, 1995] P. K. Pook, D. H. Ballard, *"Remote teleassistance"*, Proceedings of 1995 IEEE International Conference on Robotics and Automation, Nagoya, Japan, pp.944-949, 1995.

[Regan, 1993] E.C. Regan, *"Some side-effects of Immersion Virtual Reality"*, AGARD Proceedings, Virtual Interfaces: Research and Application, Lisbon, Portugal, pp.16.1-16.7, October 1993.

[Regian, 1993] J.W. Regian, W.L. Shebilske, J.M. Monk, *"A preliminary evaluation of virtual reality as a training tool for visual-spatial tasks."*, Armstrong Laboratory (AFMC) Human Ressources Directorate, Technical Training Division, Brooks Air Force Base, TX, May 1993.

[Robertson, 1991], G. Robertson, S. Card, J. Mackinlay, *"The Information Visualizer"*, Proceedings of CHI91.ACM, New Orleans, LA, pp.181-188, 1991.

[Satawa, 1997] R.M. Satawa, R.A. Robb, *"Virtual Endoscopy: Application of 3D Visualization to Medical Diagnosis"*, Presence: Teleoperator and Virtual Environments, vol.6, no.2, pp.179-197, April 1997.

[Sayer, 1992] C. Sayer, R.P. Paul, M. Mintz, *"Operator interaction and teleprogramming for subsea manipulation"*, 4th IARP Workshop on Underwater Robotics, 1992.

[Serway, 1989] R.A. Serway, *"Optique et physique moderne"*, Les Editions HRW Ltee, Montreal, pp.165-167, 1989.

[Silicon Graphics Inc, 1997], Silicon Graphics Inc., *"Cybernet Systems PER-Force Hancontrollers & Joysticks"*, SGI web page.

[Silicon Graphics Inc., 1994a], Silicon Graphics Inc., *"IRIX System Programming Guide"*, D.N. 007-1794-030, March 1994.

[Silicon Graphics Inc., 1994b], Silicon Graphics Inc., "*Onyx Family Product Guide*", ONYX-PROD-GD (1/95), 1994.

[Slavkoff, 1997] E. Slavkoff, "*Articulating Human Hands and Manipulating Objects in Virtual Environments*", M.Eng Thesis, McGill University, submitted June1997.

[Spar, 1995] Spar Aerospace, "*SSRMS system operating procedures*", Spar Aerospace, August 1995.

[StereoGraphics, 1993] StereoGraphics Company, "*CrystalEyes PC*", Company brochure, San Rafael, CA, 2 pp., 1993.

[Stevens, 1993] W. R. Stevens, "*Advanced Programming in the UNIX Environment*", Addison-Wesley, Reading, MA, 744 pp.1993.

[Stredney, 1995] D. Stredney, W. Carlson, J.E. Swan, B. Blostein, "*The Determination of Environmental Accessibility and ADA Compliance Through Virtual Wheelchair Simulation*", Presence : Teleoperators and Virtual Environments, vol. 4, no.3, pp.297-305, Summer 1995.

[STRICOM, 1997], U.S. Army Simulation, Training and Instrumentation Command Web Page, http://www.stricom.army.mil/PRODUCTS/simnet.html.

[Stytz, 1993] M.R. Stytz, E. Block, B. Soltz, "*Providing situation awareness assistance to user of large-scale, dynamic, complex virtual environment*", Presence: Teleoperators and Virtual Environments, vol. 2, no. 4, pp. 297-313, Fall 1993.

[Takahashi, 1992] T. Takahashi, H. Ogata, "*Robotic assembly operation based on Task-level teaching in virtual reality*", Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Nice, France, pp.1083-1088, May 1992.

[Tam, 1996] E.K. Tam, P. Allard, M. Faraj, M. Kaddoura, A. Mourad, H. Liu A.S. Malowany, R.J. Marceau, L. Granger, J. Gagnon, *"WITS: A Reusable Architecture for a VR-Based ITS"*, ITS'96 Workshop, Montreal, QC, 1996.

[Teittinen, 1996] Maryland CS Technical Reports by Marko Teittinen home page http://www.cs.umd.edu/TRs/authors/Marko_Teittinen-no-abs.html

[Verbex, 1990], Verbex Voice Systems, "Grammar Development Manual", Verbex Voice Systems, Edison, NJ, 1990.

[Vertut, 1986], J. Vertut, P. Coiffet, "*Teleoperation and Robotics: Evolution and Development.*", Robot Technology series, Prentice Hall, Englewood Clifs, NJ,vol. 3A, 332 pp., 1986.

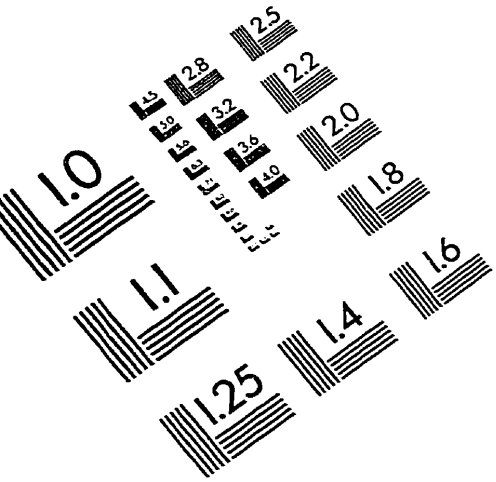
[Wang, 1990] J.F. Wang, V. Chi, H. Fuchs, "*A real-time optical 6D tracker for head-mounted display system*", Computer Graphics, vol.2 no. 24, 1990.

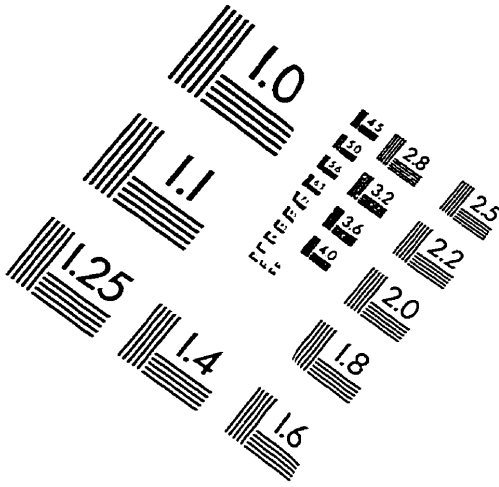
[Wenzel, 1992] E. Wenzel, "Localisation in Virtual Acoustic Display", Presence : Teleoperators and Virtual environments, vol. 1, no. 1, pp.80-107, March 1992.

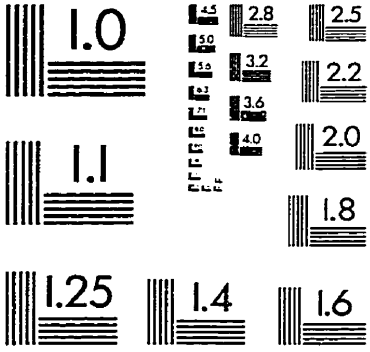
[WTH, 1997] WTH Systems Inc., "*Officer Of the Deck (OOD)*" home page http://www.netaxis.qc.ca/wth/ood/htm.

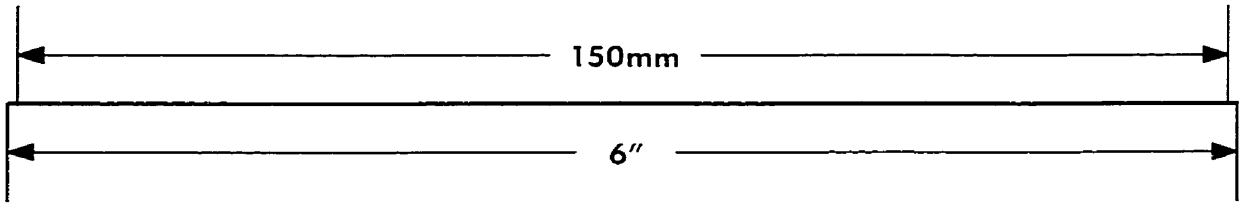
[Zeevie, 1990] Y. Zeevie, O. Hileemrath, "*Single camera Three Dimensionnal head position sensing system*", US Patent 4956794, 1990.

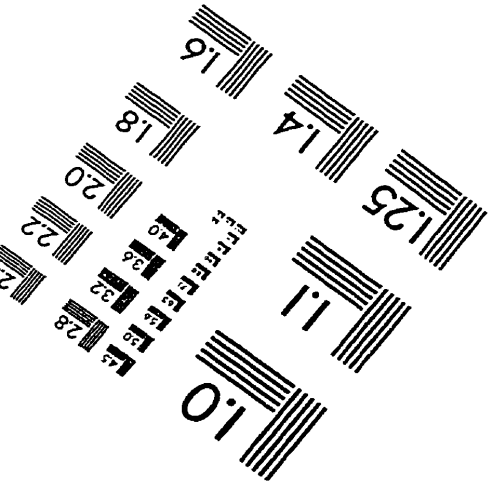
IMAGE EVALUATION
TEST TARGET (QA-3)

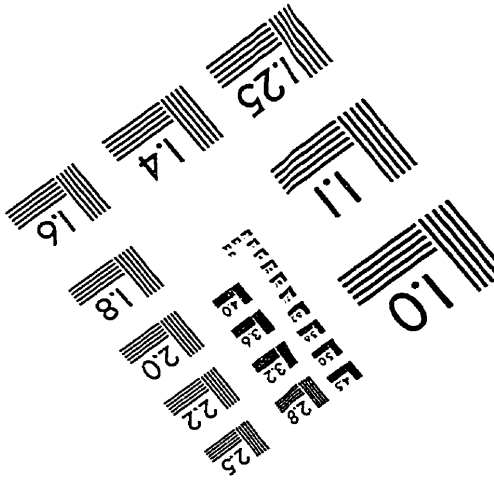












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989