SOUND AND VISION: AUDIOVISUAL ASPECTS OF A VIRTUAL-REALITY PERSONNEL-TRAINING SYSTEM

Copy 1

Alexander Okapuu-von Veh

Department of Electrical Engineering
McGill University, Montréal

March 1996

A Thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements of the degree of
Master of Engineering

ISBN 0-612-12130-5

ABSTRACT

This thesis describes a prototype virtual reality (VR) training system, ESOPE-VR, designed and implemented for Hydro-Québec by graduate students at McGill University and École Polytechnique de Montréal. The project was motivated by the necessity of providing a realistic training environment for substation operators, while ensuring their safety and the network's integrity at all times.

With the simulator, trainees can carry out all the switching operations necessary for their work in absolute safety, while staying in a realistic environment. A speech-recognition system controls the training session, while audio immersion adds a dimension of realism to the virtual world. An expert-system validates the trainee's operations at all times and a steady-state power-flow simulator recalculates network parameters. The automatic conversion of single-line diagrams enables the construction of three-dimensional models of substation equipment.

The present thesis focuses on the speech command, audio, video and network aspects of the system. A survey of current VR applications and an overview of VR technology are followed by a summary of the ESOPE-VR project.

RÉSUMÉ

Cette thèse présente un prototype de système d'entraînement d'opérateurs en réalité virtuelle (RV), ESOPE-RV, conçu et implanté par des étudiants à la maîtrise à l'université McGill et à l'École Polytechnique de Montréal. Le projet est motivé par la nécessité de donner un environnement d'entraînement réaliste pour les opérateurs de poste, tout en assurant leur sécurité personnelle et l'intégrité du réseau en tout temps.

Avec le simulateur ESOPE-RV, les opérateurs peuvent accomplir toutes les manoeuvres nécessaires pour leur travail en sécurité tout en restant dans un environnement réaliste. Un système de reconnaissance de la parole contrôle la simulation, et l'immersion sonore donne une dimension de réalisme au monde virtuel. Un système expert valide les actions de l'opérateur en tout temps et un engin de calcul d'écoulements de puissance recalcule les paramètres du réseau. La conversion automatique des schémas unifilaires permet la construction de modèles tri-dimensionnels d'équipements de postes.

Cette thèse a comme focus les aspects techniques du système, dont ceux de la commande vocale, du son, de vidéo et du réseau. Un survol des applications courantes en RV et un sommaire de la technologie courante en RV sont suivis par une présentation du projet ESOPE-RV.

ACKNOWLEDGEMENTS

Many thanks go to Prof. Alfred S. Malowany, who supervised this Master's Thesis, and to Prof. Richard J. Marceau of École Polytechnique de Montréal for being the expert of positive reinforcement in many late-night phone calls. Without the two of them we'd still be "flying upside down."

It was a pleasure going through the ups and downs of this project with Alain Daigle, Patrick Desbiens, Etienne Garant and Amir Shaikh. Special thanks are due to Etienne for dragging me kicking and screaming from Irene Leszkowicz's Computer Graphics course and putting me in the middle of the ESOPE-VR project.

The vision of Mr. Louis Marquis of Hydro-Québec, along with generous financial and technical support, propelled the project and us to where we are now. The original ESOPE expert system was created by Mr. Robert Gauthier of Hydro-Québec.

Last, but not least, I would like to thank my mother, Margaret von Veh, for putting up with me all these years, and for pushing me along in the right direction, and my father, Ülo Okapuu, for giving me batteries and light bulbs to play with when I was three. (Electrical Engineering has to start somewhere!)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

The following abbreviations are used in this thesis:

A_V	AudioVideo
AIFC	Audio Interchange Format Compressed
ASCII	American Standard Code for Information Interchange
BGS	Background Sound Server
ERV	ESOPE-RV
ESA	European Space Agency
ESO	ESOPE
ESOPE	Expert System Operations Planning Environment
GL	Graphics Language
GUI	Graphical User Interface
HHD	Hand-Held Display
HMD	Head-Mounted Display
IPC	Inter-Process Communications
LCD	Liquid Crystal Display
MOV	ShowMovie Server
MRI	Magnetic Resonance Imaging
NASA	National Aeronautics and Space Administration
NCSA	National Center for Supercomputing Applications
PC	Personal Computer
RE	Reality Engine
RPC	Remote Procedure Call
SGI	Silicon Graphics Inc.
VBX	Verbex
VE	Virtual Environment
VEOS	Virtual Environment Operating System
VR	Virtual Reality
WoW	Window on a World
WTK	WorldToolKit

All trade names mentioned in this thesis are registered trademarks of their respective owners.

CHAPTER ONE

1. Virtual Reality Applications and Technology

1.1. Introduction

This thesis describes an implementation of a virtual reality operator training simulator, ESOPE-VR, designed and implemented for Hydro-Québec by students of McGill University and École Polytechnique de Montréal. As Chapter two will present an overview of the ESOPE-VR system, this chapter focuses on examples of current virtual reality applications and the technology behind them.

The term "virtual reality" can mean various things to different people. For some, virtual reality (VR) means a collection of certain technologies such as a head-mounted display for visualisation, a glove with various sensors as an input device, and some sort of audio feedback. In general, VR is a way for us to visualise, manipulate and interact with computers and complex data [Aukstakanis 1992]. In this case, the term "to visualise" means that a computer generates visual, auditory or other outputs to the user, showing a virtual environment (VE) that exists only in the computer's memory. The user has various ways of interacting with the virtual world, and can manipulate objects within it. A critical test for VR is near-real-time interaction with the VE [Isdale 1994].

One must be careful not to confuse virtual reality with the other "buzzword" that is heard every day: *multimedia*. Jacobson gives a good explanation why this is so:

> "...It is common practice for speakers to allude to virtual worlds as 'the next step beyond multimedia' or 'a type of simulation.' Both claims are wrong.... Most multimedia productions, at their core, are scripted to permit one or a number of outcomes conceived by their authors. The virtual world, being composed around the experience of the participant (...) is subject to in-

tervention by the participant. (...) The idea that virtual worlds are a type of simulation is more difficult to refute, not because it is right, but because, on cursory examination, it *appears* to be right. (...) 'Appearing to be right' is wrong. Nothing in the 'real world,' the material world, is 'right' in the sense of being perfect..." [Jacobson 1993].

We must be careful not to present an "ideal" world to the observer with our application, but must let things occur that would also be present in real life. A flight simulator is a useful tool, but if we cannot simulate engine failure with it, the training benefit to the future pilots decreases.

Of course, one must also be realistic and see that, currently, VR is more of an oxymoron, in that what is portrayed is still far from "real." The term VR suggests performance that is much higher than what current technology can provide [Bryson 1994]. Notwithstanding this fact, however, progress is being made in the right direction, that is, toward something that is indistinguishable from reality. Wilson states that VR may go the same way as artificial intelligence has, since it lacks the ability to make everything "nearly perfect" [Wilson 1993]. According to him, VR should be replaced by the term "virtual environment displays" so that not too much emphasis is placed on "reality."

1.2. Current Virtual Reality Applications

This section presents an overview of several kinds of current applications of virtual reality: commercial, medical, industrial, and military.

1.2.1. Commercial Applications

Many VR applications currently being developed have been talked about in the popular press and the academic literature. Sterbak writes about how Mercedes-Benz will use VR to give its customers a "virtual automobile showroom" in which they can experience the automobile they want to buy before doing so. They are able to change the interior configuration, select various added features, and choose paint colours and upholstery fabrics – and receive a constant update of the price tag [Sterbak 1994].

Telecommuting is a popular word these days: Mannes mentions the "Virtual Office" as a place that improves productivity, gives employees a flexible work schedule, and eliminates time wasted sitting in traffic. He mentions that companies such as Pacific Bell, AT&T, Xerox and American Express, have all made some effort in this direction [Mannes 1995]. A recent study [Caldwell 1996] about telecommuting shows that productivity gains of 10-20% are achievable, and that the number of users in the US should reach 25-30 million people by the year 2000. Osberg writes about a VR-based science camp for children, and mentions some of the advantages of this technology, but also its pitfalls and drawbacks [Osberg 1992]. A financial planning application using WorldToolKit is described in [Coull 1992]. It uses a three-dimensional grid to represent various stocks and different colours to describe each one's activity.

Many of these applications provide worlds that are very similar to our own: Astheimer describes a generalized, distributed VR system designed for architectural purposes (interior design, buildings, cities, landscapes), which can also be used for factory simulation, practicing assembly and disassembly tasks, modeling, historic reconstruction as well as acoustic simulation [Astheimer 1993, 1]. An extremely popular VR game that originated in California, and which is now also in Montreal, is *Le Monde Virtuel*, an interactive environment in which up to ten players can compete at flying spaceships or can engage in robot battles. The widespread popularity despite the steep price tag ($10 for a half hour) shows that such applications are economically feasible!

It is clear that when comparing the applications described here with other ones described in this section, that there is a considerable gap in terms of universality. While research-based systems are quite prevalent, true commercial applications are less so, mainly because of the high cost involved in producing an acceptable product. It is, however, only a matter of time before this situation changes. One can compare the evolution of VR to that of personal computing. Ellis makes the statement that the "virtual environment industry has yet to find its Visicalc" -- a reference to the spreadsheet program that showed a realistic use of PCs to thousands of potential users [Ellis 1994].

1.2.2. Medical Applications

Virtual reality presents advantages to many fields in medicine, and more are being developed. Coull describes its use at a hospital in Virginia for the rehabilitation of physically challenged people [Coull 1992]. By rolling the wheels of a wheelchair, the person can move around in a virtual environment (VE), and using a DataGlove, interact with objects in it. Kleinfeld mentions similar uses of VR at the Bronx Veterans Affairs Medical Center [Kleinfeld 1995]. Several other applications of uses of VR in work with disabled people are mentioned in the Summer 1994 issue of *Presence* [Nemire 1994], [Vanderheiden 1994].

Piantanida describes further applications of VR, mainly in the biomedical sciences, and for several aspects of medicine and surgery ranging from neurosurgery to obstetrics and teaching. He states that, for example, using a semi-transparent display, "it may be possible to superimpose remotely sensed images, e.g., MRI (magnetic resonance imaging) onto – and stereoscopically into – a real image of the patient" [Piantanida 1992]. Peters says that stereoscopic images are in fact regularly used in neurosurgery, albeit not at the level described by Piantanida, and that the driving forces behind this technology are the surgeons who are demanding it [Peters 1995].

Hunter *et al* describe a teleoperated microsurgical robot for eye surgery which incorporates stereo vision as well as force feedback. They discuss the technical implementation of visual feedback in detail, and mention that signal propagation delays present a problem affecting the use of this tool [Hunter 1993]. The modeling aspects of this particular application are discussed in detail by Sagar *et al* [Sagar 1994]. They state that the "creation of a surgical virtual environment to both aid surgeons during operations and provide simulations for training opens up many exciting possibilities. The surgical procedure will be able to be viewed in new ways providing a new level of surgical experience." Other computer models suitable for medical simulation are described by Cover and Ezquerra, who also discuss the benefits and drawbacks of this type of simulation [Cover 1994].

Höhne *et al* describe a 'Virtual Body' model to be used for surgical education and rehearsal of surgical procedures [Höhne 1996]. They show the evolution of computer use in medicine from the first attempts at three dimensional (3D) visualization to simulation of surgical interventions to the extremely complex task of providing interoperative surgical support. They use the concept of an "intelligent volume" to represent data. Each voxel[1] may belong to more than one type, for example structural and functional anatomy. Finally, they present several different features that evolve from this "intelligent volume" idea: virtual anatomy, virtual radiology, virtual surgery as well as virtual endoscopy. The January, 1996, issue of *Computer*, which the Höhne article appears in, focuses entirely on computer applications in surgery.

1.2.3. Industrial Applications

With the relatively recent availability of powerful computers, their use to design new products has become popular as it greatly reduces the time and effort spent. Aircraft design, for example, has evolved rapidly over the last few years, and newer aircraft, such as the Boeing 777 or the Airbus 340, were conceived and modeled largely using computers [Esposito 1993], [Eberl 1994]. Virtual reality gives us ways of viewing things from a perspective that is not available to us in the real world, for example in data visualization [Gershon 1994].

Bryson discusses the use of VR at the NASA Ames Research Center for the visualization of large (1 GB and up), complex data sets [Bryson 1992, 2], [Bryson 1993]. The "virtual windtunnel" is one particular application mentioned, where researchers use

[1] Voxel = volume element (from pixel = picture element)

a virtual environment to study fluid flow around aircraft. Handling the large data sets generated by this kind of application demands the appropriate hardware configuration: Bryson cites the use of an Silicon Graphics Inc. (SGI) Iris380 VGX with 256 MB of memory, along with a Convex supercomputer with 1 GB of memory. As the available computing power increases, newer, larger applications will be created.

Hitchner describes another VR application at NASA Ames Research Center, the Virtual Planetary Exploration Testbed [Hitchner 1992]. Very large terrain data sets from the Viking missions of the 1970s are used to create a virtual world in which the user can travel over certain sections of Mars. He emphasizes the need for "a compelling, immersive experience, but also (...) scientific validity, high fidelity, accurate measurability, and the capability for data enhancement via scientific visualization techniques."

Bagiana and Buc describe the VR activities in the Simulation Facilities Section at the European Space Agency (ESA) [Bagiana 1993], [Buc 1994]. A VR prototype is described that was derived from existing simulation software and then extended to work on a distributed system using SGI workstations (Indigo and Onyx RE^2). Bagiana presents a training tool for astronauts preparing for work in the Columbus space station, which uses various hand-object interactions with the DataGlove: in this case "high-definition graphics, sound and force feedback are necessary to provide the correct cues and therefore avoid 'negative training.'" Buc describes a flight simulator which was developed using the European Space Agency's (ESA) Eurosim simulator and that "reproduces the landing of an unpowered winged vehicle in real time."

Hale, in describing the VR program at NASA's Marshall Space Flight Center, says that a "validated set of VR analytical tools must be developed to enable a more efficient process for the design and development of space systems and operations. Similarly, training and mission support systems must exploit state-of-the art computer-based technologies to maximize training effectiveness and enhance mission support." He continues to describe human factors applications and validation studies and mentions a few possible future applications of VR: it could "provide a timely and safe method to enable the various advantages and disadvantages of reaching and maneuvering in a microgravity environment." Furthermore, it could be used to "permit anthropometric sizing to reflect the dimensions of the on-board crew. This is particularly useful for operations being planned in relatively tight spaces" [Hale 1992].

A recent addition to the list of commercial applications is the use of a virtual environment by Caterpillar Inc. to test drive new construction vehicles [Jones 1995]. Together with the National Center for Supercomputing Applications (NCSA) and using the Sense8 WorldToolKit software (described in greater detail in Chapters two and three), they are able to test various aspects such as vehicle design and visibility from the operator's position in the cab.

1.2.4. Military Applications

One of the driving forces behind technological development, at least until the end of the Cold War, was military spending [Holzer 1994, 1 - Holzer 1995, 2]. There is still a lot of interest in military simulation these days, mainly because of its cost-benefit ratio. As an example: Roos states that flying an Apache helicopter for an hour and a half, and firing all its ammunition in that time costs approximately $335,000, while simulating the exercise would cost a mere $143 [Roos 1995]. Holzer says that the use of simulation enabled the U.S. Navy to fire only thirty Sidewinder missiles in order to test the current version, as opposed to over 300 live firings for the previous one [Holzer 1994, 2].

Future simulators used by the armed forces, such as the $100 million Synthetic Theatre Of War (STOW) simulator, are expected to be used by not only the military, but also by the U.S. Coast Guard, the Drug Enforcement Administration, and by civilian organizations such as the Red Cross. When more detailed modeling and simulation technology becomes available, this simulator could be used, for example, to coordinate a humanitarian or a disaster-relief mission [Holzer 1995, 2].

1.3. Types of VR Systems

While there are many uses for virtual reality, the user-interface is one of the major distinguishing factors between VR systems. Some of the categories of VR applications are Window on a World, Video Mapping, and "Fish-Tank VR," Immersive Systems, and others such as Telepresense, and Mixed Reality.

1.3.1. Non-Immersive Systems

Window on a World (WoW) systems use a conventional computer monitor to present the virtual environment to the user. The idea of a WoW goes back to the very beginnings of Computer Graphics. In 1965, Ivan Sutherland made the statement: "One must look at a display screen as a window through which one beholds a virtual world. The challenge to computer graphics is to make the picture in the window look real, sound real and the objects act real" [Sutherland 1965]. Video Mapping is a variation on WoW. The participant in a video-mapped VR environment watches a monitor that shows his own body's interaction with the world. Another variation of WoW is called "Fish Tank VR", and is used to describe a system where a head-tracker is used along with stereo glasses to give the user a better simulation due to the added effects of modifying the image as based on the head tracker. This sort of system was first introduced at the INTERCHI conference in 1993 [Ware 1993].

1.3.2. Immersive Systems

Non-immersive systems have the drawback that the user's viewpoint is "outside" the virtual world. In immersive systems, which are often equipped with a head-mounted display (HMD), the user is "inside" the world. An example is the "Dive" system, which is a distributed VR system that allows multiple users to interact with each other in a virtual world [Carlsson 1993]. A variation on the immersive systems is to use multiple large projection displays to create a "Cave" or room which the viewer(s) can move around in [Cruz-Neira 1992], [Cruz-Neira 1993], [DeFanti 1993]. The Cave environment is controlled by a SGI Onyx with four processors and three dedicated graphics engines (Reality Engine²). The Cave offers the advantage of more realistic display as the projection plane for the image is an actual wall and not a fixed distance from the user's eyes, as is the case with a HMD [Roy 1995].

1.3.3. Telepresense

A variation on visualising complete computer-generated worlds is telepresence, which combines remote sensors in the real world with the senses of a human operator. This technology is very useful in situations where the action takes place in locations where a human cannot go, mainly for safety reasons, such as deep-sea or volcanic exploration, fire-fighting, bomb-disposal, and others.

Cooke and Stansfield present a system for creating and verifying computer-generated graphical models of remote physical environments [Cooke 1994]. The system uses a robot with twin cameras to view the environment, in this case underground storage tanks for radioactive waste, and uses an SGI Crimson Reality Engine along with a Boom stereoscopic viewer to present the desired images to the user. A Dragon Systems voice recognizer is used to control the robot.

Miner describes a distributed system using VR as a method to interact with robots engaged in industrial cleanup tasks [Miner 1994]. Speaker-independent voice recognition is used as a control input with varying results. Audio feedback is used to relate information regarding the current status of the robot to the user and to provide help about available commands at any point. Reference is made to the use of multimedia information to improve the training aspect of the system.

1.3.4. Mixed Reality

The term *mixed reality* refers to a combination of traditional VR and telepresense: computer-generated images are combined with telepresense inputs or with the user's view of the real world. For example, a surgeon's view of a brain surgery is overlaid with images from earlier CAT scans and real-time ultrasound [Peters 1995], or a fighter pilot sees computer generated maps and data displays inside his helmet visor [Adam 1994], [Furness 1986].

Milgram defines two types of mixed reality: "Augmented Virtuality" is the adding of real objects or images to a virtual environment, while "Augmented Reality" is the addition of virtual objects to the real world [Milgram 1995]. He mentions the use of "Augmented Virtuality" at the ATR Communications Systems Lab in Kyoto for "Virtual Space Teleconferencing." This is a virtual world in which, for example, a person's image is superimposed over a stereo video background, thereby eliminating the need for extensive modeling. At University of Toronto's Ergonomics in Teleoperations and Control Lab, work is being done on an Augmented Reality through Graphic Overlays on Stereovideo (ARGOS) system which is monitor-based, rather than HMD-based. The possibilities here include using virtual objects for collision avoidance for real-world robots. If the virtual and real coordinate systems can be linked, then it is possible for a user to specify objects that the robot must avoid without having to interact with actual physical objects.

Quéau uses the term "televirtuality" to describe the "merging of telecommunications and virtual reality" [Quéau 1993]. This fusion could be useful for a "picturephone conference" which would act as a backup for existing teleconference links, "or even replace them completely, given the economic advantages" of much lower bandwidth needed to transfer information.

1.4. Virtual Reality Technology Overview

This section presents an overview of the existing VR technology, ranging from image generation equipment over control devices to speech recognition. Much of the current technology used for VR can be traced back to developments in vehicle simulation and teleoperation.

1.4.1. Image Generators

One of the most compute-intensive tasks in a VR system is the generation of the images. For the visual (motion) aspect of a VR application to be "acceptable," a minimum frame-rate of 10-15 Hz is required [Bryson 1992, 1], [Göbel 1993]. If stereo vision is required, this implies that 20-30 images need to be recalculated per second. This is not to be confused with the refresh rate of the computer monitor itself, which should ideally be greater than 60Hz. Since fast computer graphics opens a very large range of applications aside from VR, there has been a market demand for hardware acceleration some time. For personal computers (PCs), many new products enter the market every year. Initially, many of these cards were based on the Intel i860 processor, but more specialized chips are being developed by a number of companies, with prices for video cards starting at about $400 US. The most widely used high-level graphics hardware comes from Silicon Graphics, though, who make computers ranging from a few thousand to several hundred thousand dollars. Perhaps the most realistic VR applications today are large-scale flight simulators made by companies such as CAE, Hughes Rediffusion or Thomson-CSF, for which a price-tag of US$12M is not unusual [Taverna 1993]. Approximately 30-40 such simulators were sold around the globe each year [Condom 1993] at the beginning of the 1990s, while now the market has become saturated, and about half that number are made.

1.4.2. Stereo Vision

One of the more important aspects of a VR system is the use of stereoscopic computer graphics. A stereoscopic display is, in effect, an optical system that has as its final component the human brain itself. It functions by giving each eye a slightly different image to look at, the same way we see things in the real world. Hodges states that "an observer's perception of depth in the image arises from cues such as shading, shadowing, occlusion, motion, and linear perspective, as well as structure and size of familiar objects. *Stereoscopic computer graphics* adds the additional depth cue of *stereopsis*" [Hodges 1992]. Stereopsis is the merging in the brain of two slightly offset images into a single 3D image. The basis for this idea was first expressed by Wheatstone, who explained the sense of *stereopsis* by saying that the brain fuses the two retinal images into a single one with "solid seeing" [Wheatstone 1838]. The history of stereoscopic computer graphics can be traced back to 1974, where a dual-trace oscilloscope was used to produce two images at a rate of 300 Hz. In [Hodges 1992], a detailed description of stereoscopic computer graphics with many technical details and some useful definitions is presented, while Lipton gives an overview of one particular application, namely the Crystal**Eyes** shutter glasses, which are used in conjunction with a high-refresh-rate computer monitor [Lipton 1991]. In this case, the right eye is obscured while the image meant for the left eye is displayed, and vice versa. This way, the brain is able to construct a 3D view from the two separate images. An authoritative book on computer graphics in general is [Foley 1990].

Milgram mentions several experiments to study depth cues, for example depth through motion (rotation, or stereoscopic and rotation) [Milgram 1995]. For example, an image of two intermeshed "trees" is shown to a person who is asked to determine which trunk one of the highlighted branches belongs to. With a static, monoscopic image, the success rate is nearly zero. A very high success rate is achieved when a stereoscopic, rotating image is used. Ware *et al* adapted the "Fish Tank" VR system to replicate Milgram's experiment with successful results [Ware 1993].

The most realistic way for provide stereo vision is the use of a head mounted display (HMD). Most use LCDs, while some have small cathode ray tubes (CRTs) or, in the most expensive ones, optical fibres bring the image from where it is generated. The CAE Fibre-Optic Head-Mounted Display uses General Electric's light valves to produce the image [Kalawsky 1993], at a cost of one million dollars! HMDs generally require a head-tracker to determine what kind of image to send the user. Much discussion has been generated about the dangers of using HMDs in the first place. Many people have experienced severe eye-strain, blurred vision, and disorientation while wearing a HMD, and have even had "post-HMD" accidents [Isdale 1994]. The most likely reason for so-called "VR sickness" is that there is a perceivable time-lag between the user's actions and the visual response created, which the brain is incapable of ignoring.

One of the constraints of a HMD is the fact that the user cannot see what is around him. This is one of the factors that will sometimes make users prefer a monitor-based technology. Work at the University of North Carolina to incorporate images from head-mounted cameras into the virtual display is mentioned in [Milgram 1995]. This input has the obvious advantage of eliminating disorientation of the user, but also enables "extra-sensory" inputs such as ultraviolet or infrared images overlaid on the virtual world. Of course, such a system is only useful in a mixed-reality environment, where the user is *supposed* to see his surroundings!

A different approach to VR visualization is presented by Amselem [Amselem 1995]. He describes a Hand Held Display (HHD), based upon a liquid crystal display whose position and orientation are tracked in six dimensions, with the image shown being continuously modified as if it were a window into the virtual world.

It may be reasonable to expect the use of holographic technology as a means of providing a virtual world in the near future.

1.4.3. Manipulation and Control Devices

A key element for interaction with a virtual world, is a means of tracking the position of a real world object, such as a head or hand. For the current project, a 6-degree-of-freedom mouse and/or a head tracker are used, which are both based on the same technology: they use ultrasonic transmitters and microphones to determine the object's position and its orientation by triangulation.

Another common VR device is the glove. A glove has sensors built in to it to determine the flex in the fingers, and a sensor to determine the hand's position and orientation. Gloves range in price and complexity from $50 for a Powerglove to many thousands of dollars for a DataGlove with fiber-optic and magnetic sensors [Sturman 1994]. The Powerglove, originally made by Mattel for the Nintendo game system, has recently emerged as a low-cost VR device, after never having been extremely popular for its original market. Fortunately for the developers of low-cost VR setups, the Powerglove is being manufactured again in an improved version. In the interim, while no new gloves were available, a suggestion on the internet [Isdale 1994] was to purchase these gloves from children who "like money more than toys they never use."

Other devices include trackballs, or joysticks which can be used in a limited way to generate 6-D inputs, or other actual 6-D controllers (made by Polhemus Inc., for example).

Position tracking of arms, legs and bodies is accomplished by various means: some are mechanical, and provide fast, accurate measurements, such as exoskeletons; others are ultrasonic sensors or magnetic trackers.

1.4.4. Audio Output

Another useful technology for VR is audio output. In the case of this project, the audio ports of two SGI workstations are used to provide aural immersion. There are several commercially available tools for audio feedback, starting with the SoundBlaster cards for PCs. High-end systems such as the Convolvotron, use convolution of the audio signal with pairs of filters corresponding to measured impulse responses at the listener's ears to provide an accurate 3D localization of the aural image [Wenzel 1988], [Kalawsky 1994]. These are devices that go beyond quadrophonic sound generation to provide audio signals that the listener can accurately position in space. Sound has also been suggested as a means to convey other information, such as surface roughness. Dragging a virtual hand over sand would make a different sound than dragging it through gravel.

1.4.5. Voice Recognition

Voice recognition is another useful input for a VR simulation, because it helps the users keep their hands free for other tasks. If the user is already wearing the 3-D glasses or a head-mounted display, and a data glove, it is not reasonable to expect him to start typing on a keyboard in order to get things done. The following is summarized from [Simpson 1985], and gives an overview of speech recognition definitions.

1.4.6. Speech Recognition Definitions

Speech recognition systems come in many different forms. Several factors, for example *speaker dependence*, *speaking mode*, and *vocabulary size* can be used to differentiate between various systems.

Speaker dependence refers to the extent to which the system must have data about the voice characteristics of the particular human speaker using it. Speaker-dependent recognition systems can recognize the speech of a particular human speaker only if examples of that person's speech have been provided. Speaker-dependent systems must therefore be trained separately for each speaker who will use them if good (or any) recognition accuracy is to be obtained [Verbex 1990].

Another parameter of speech recognition is the *speaking mode*, that is, the manner in which utterances are spoken to the system. With *isolated-word systems*, which are most widely used, the user must pause briefly between each word when speaking. *Connected-word* systems are able to recognize words with utterances spoken without artificial pauses between words. However, the individual words are spoken with the same intonation pattern that would be used if they were read from a list.

Continuous speech recognition or *continuous speech understanding* adds another dimension, meaning that a system can accomplish tasks using continuous speech input. The Verbex 7000 system used for the current project can recognize short segments of continuous speech of approximately 20 words in length, which is more than adequate for most applications.

The term *enrollment* refers to the training of a speaker-dependent system. Each speaker who will be using the system must train it individually. In the case of Verbex, a simple method of training it is provided [Verbex 1990].

A final parameter that can be used to differentiate between speech recognition systems is *vocabulary size*. Speech recognition systems with a fixed vocabulary must be provided with samples of each word or phrase they are to recognize. They perform acoustic pattern-matching at the word and phrase level and typically handle vocabularies of 100 to 200 words. The Verbex system can also handle approximately this level of vocabulary, provided the words are not in too complicated an order, that is, the grammar definition must not be too convoluted.

The most commonly used measure of *performance* for speech recognition algorithms is the *recognition accuracy*. This simply means the percentage of speech utterances that are correctly recognized by the listener out of a given set of expressions and under a particular set of conditions. The listener in this case is the machine or computer, just as a person would listen to another person speak.

Four classes of errors are categorized: *substitution errors*, where one word is mistaken for another one, *insertion errors*, where a word is inserted but has not actually been spoken, *deletion errors*, where a spoken word does not get recognized, and finally, *recognition errors*, where a legal item in the given vocabulary is detected but not recognized correctly.

Of course, both humans and machines make rejections of words that are in fact correct. For a machine such as Verbex, the user coughing or breathing in a strange way is enough to cause recognition errors. For humans, unfamiliar words or a poor signal to noise (S/N) ratio, such as a noisy working environment, will often cause recognition errors. The main advantage a human speaker has in such a case is the possibility to infer the missing parts from what is understood correctly, a capability that a relatively simple system such as Verbex does not have.

1.4.7. Current Uses of Speech Recognition

Many technologies today lend themselves to interaction with speech recognition. Popular applications are, for example, voice dialing of telephone numbers. A telephone subscriber can store meaningful speech patterns such as "mother" or "pizza" and have the telephone automatically dial the desired number. Vending machines and VCRs can be controlled by verbal commands [Berardinis 1993].

Dictation systems are also becoming popular, using speech input as an alternative to the keyboard. By relying on verbal commands for dictation, people who have suffered repetitive-stress disorders such as carpal tunnel syndrome can continue to do their job successfully. Janah and LaPlante present two low-tech overviews of speech recognition dictation systems as used by the New York Times [Janah 1994], [LaPlante 1994].

Cooke's and Miner's systems, mentioned above, use speech input as a way of controlling robots in industrial cleanup tasks and in underground radioactive storage tanks [Cooke 1994], [Miner 1994].

Voice recognition systems have many advantages if they are used properly. Since the technology commercially available is not at an advanced enough stage to be used by anyone at any time, failures are common. For example, Gosbee presents a "post-mortem" analysis of the failure of a voice recognition system in an urban hospital's emergency department [Gosbee 1993]. He shows that too much time was necessary for training, but that not enough time was available, that the pre-defined vocabulary and templates were frustrating, and a lack of computer skills and time prevented changing the situation. Ambient sounds impeding proper functioning are also mentioned as one of the reasons for the system's failure. He makes several recommendations as to how the system could be improved.

For an in-depth, technical description of current and future trends in voice recognition, please see Roe [Roe 1993].

1.5. Claim of Originality

The author has participated in the development of a type of training simulator that has not, to the best of his knowledge, been developed elsewhere. The addition of a three-dimensional virtual reality interface to the existing commercial software, ESOPE, represents a significant step in the direction of a fully immersive, lifelike training environment. The author's contributions to this project are the following:

• A speaker-dependent voice interface to let the trainee interact with the system by speech commands. This included the integration of an existing speech recognizer with the rest of the project, the development of an appropriate grammar and syntax for this particular application, as well as the implementation of interface programs.

• A multimedia help module to give the user the option of seeing digitized photographs or videos to assist in his training.

• A sound server module which presents an aural "world" consisting of background and action sounds to the user.

• Recording of appropriate digitized speech help and error messages for interfacing with the ESOPE expert system.

• A startup / shutdown script, written in *perl*, to help with the initialization of the system and clean-up after termination of the programs.

In addition to the above, the author also refined the network architecture for the simulator from an initial prototype. This step also included the implementation of a graphical network monitor to display various network data in an easy-to-understand format on the screen.

1.6. Summary

As can be seen from the examples presented at the beginning of the chapter, the potential of virtual reality is considerable, and already much work is being done to exploit its possibilities. Many of the sciences stand to benefit from the use of VR, for example, for modeling and data visualization, and many other non-scientific fields present thousands of potential uses. This chapter presented some of the current applications of VR technology, some commercial, some purely research oriented. In the near future, many more applications will emerge and gain popular acceptance. The next chapter shall focus on the functional implementation of the ESOPE-VR system.

CHAPTER TWO

2. ESOPE-VR System Functional Overview

2.1. Introduction

This chapter presents a functional overview of the virtual reality (VR) operator training simulator developed for Hydro-Québec: ESOPE-VR. The technical details concerning the prototype implementation are covered in Chapter three. The motivation for this project is to provide a realistic training environment for switching station operators, thus ensuring their safety and the network's integrity at all times. The risks associated with the manual operation of switching station equipment demand rigorous personnel instruction, while preserving network integrity implies not changing its topology and exposing the network to system errors for the sake of operator training. With the simulator, operator trainees can carry through all the switching operations necessary for their work in complete safety, while maintaining a high degree of realism. A speech-recognition system enables the user to control the training session, while sound immersion adds a dimension of realism to the virtual world. An expert-system validates the trainee's operations at all stages of the process and a steady-state power-flow simulator recalculates network parameters as necessary. The automatic conversion of single-line diagrams enables the construction of realistic three-dimensional models of switching station equipment.

2.2. Current Personnel Training Practice

Personnel training is an activity that cannot be neglected by today's electrical utility companies; in order to have well-prepared personnel, regular training needs to be done. To ensure a high degree of safety is one of the paramount concerns inherent in job training, and in order to guarantee uninterrupted service to one's customers, it is necessary to limit work on the actual network to an acceptable minimum level. The apparent contradiction between safety and uninterrupted service on the one hand and regular training on the other is usually solved with theoretical instructional tools. These tools range from traditional paper and pencil workbooks over computer-based-training methods to full-scale simulator mockups [Griebenow 1993]. Over the past 15 years, computer-based training systems have experienced rapid evolution [Overbye 1995]. Foley *et al* describe a graphical user interface (GUI) that could be used for training purposes [Foley 1993]. Rajagopal *et al* outline a workstation-based operator training simulator developed for Consolidated Edison [Rajagopal 1994], while Vadari *et al* write about a dispatcher training simulator [Vadari 1995].

There are certain drawbacks associated with all these examples mentioned, such as a low degree of realism and effectiveness or a prohibitive cost. These simulators are useful for certain types of training: where the trainee is being taught a task in which he will not have to move around, the training itself can easily take place at a fixed location as well. However, to properly learn what it means to be a switching station operator implies moving about within the physical dimensions of an actual station, not sitting in front of a computer screen. The operator, in his future job, will have to go from the control room to the switchyard and back again in the course of even a single switching operation. He must be able to recognize that, for example, a circuit breaker represented by a small square on a diagram may be of many different types: depending on the voltage level, it may be a relatively small oil-filled model only two or three meters high, but it may also be an air-blast type roughly 7 meters high. Additionally, different types of the same category of equipment may have widely differing physical appearances.

The actual switching operation can be controlled from within the control building, but can also be performed at the equipment under certain circumstances. The operator is required to visually inspect the equipment before most operations, in order to ensure that everything is in order. The operator usually has no more than a schematic diagram of the switching station as a guide to locate the equipment in the switchyard. An important fact to take into account is that a given operator may not only work at one switching station, but may be responsible for several, each with a different layout, and different types of equipment, depending on when it was built.

The job of a station operator challenges his memory, his ability to deal with stress, as well as his capacity to translate theory into practical work. Various attempts have been made to deal with these issues with conventional graphical user interfaces or mock-ups of actual switching station equipment [Foley 1993], [Rajagopal 1994], [Vadari 1995]. The ESOPE-VR project is an attempt at providing a virtual reality training environment for station operators in order to better address these concerns. As mentioned in Chapter one, virtual reality is being used successfully in several fields such as robotics [Cooke 1994], [Miner 1994] and medicine [Hunter 1993], [Peters 1995], [Sagar 1994].

2.3. Overview of ESOPE-VR

2.3.1. The ESOPE Training System

Currently, Hydro-Québec uses a program known as ESOPE (ESOPE = Expert System Operations Planning Environment) to train its operators. ESOPE incorporates a knowledge-based expert system that makes decisions about the switching operations the trainee is attempting. As well, it provides a steady-state power-flow simulator to recalculate the steady-state network parameters after any changes to the network's topology.

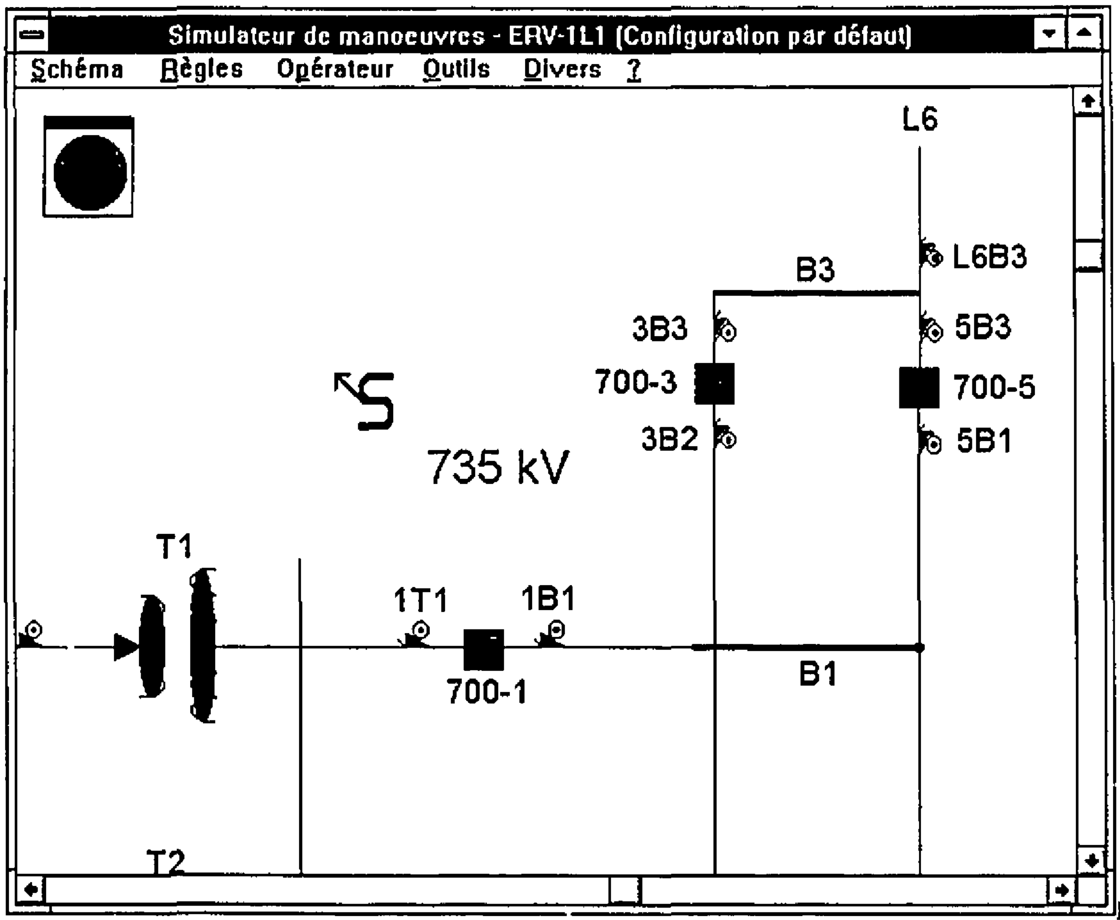

Figure 2.1: The original ESOPE Windows interface.

The ESOPE simulator runs under Windows and provides a GUI representing a single-line diagram[2] of a given switching or transformer station; Figure 2.1 shows a typical view of its screen. In order to complete a given switching operation, the trainee uses the mouse to click on the representations of circuit elements shown on the screen, and can then modify equipment parameters in various pop-up boxes using the mouse or the keyboard. For example, in order to open a circuit breaker, the operator clicks on the

[2] The single-line diagram is a type of schematic commonly used in power systems, representing the three phases of a circuit by one line, three pieces of equipment by a single symbol, and so forth.

circuit breaker symbol, then on the "Open" button within the dialog box and then on the "OK" button. At this point, ESOPE's expert system would validate the attempted switching operation and would either allow the trainee to proceed, or would flag an error condition and provide a brief explanation in a text box.

In ESOPE, the operator's assumed position in the real world can be changed to place him in the control room, the switchyard or at a remote location, with the resulting restrictions on permissible actions. For example, it is not generally possible or permitted to open an air-blast circuit breaker while standing next to the equipment; rather, this would be done from the control room. The operator's position is reflected in the cursor symbol: "S" for switchyard, "C" for control room and "R" for the remote-control centre.

ESOPE has evolved over the past several years, and is now used by several of Hydro-Québec's administrative regions for operator training. It has provided many benefits to its users, as well as to the utility itself, most notably a marked reduction in overall error rate. In Appendix 3 on page 84, a more detailed representation of the single-line diagram used for ESOPE-VR, as well as an actual Hydro-Québec training diagram, are given.

2.3.2. Esope-VR

The aim of the virtual reality training system prototype, ESOPE-VR, is to provide a tool that is cost-effective when compared to some of the currently available simulators, and that is flexible so that an operator can use it to train for any number of work environments using one training setup.

The system was developed by a joint research group from McGill University and École Polytechnique de Montréal, and builds on the functionality of ESOPE while offering a high degree of realism that gives the user a thorough preparation for the actual work of an operator [Okapuu-von Veh 1996]. While ESOPE has been found useful as a training tool, there is little relation in terms of realism between this sort of instruction and the actual work of an operator. The schematic representation provided with the Windows interface is abstract, and it is difficult to relate to any actual equipment represented on the single-line diagram.

With ESOPE-VR, a training tool is provided that places the operator inside a virtual world in which he can interact with the world's objects as if he were at a real switching station. Three dimensional views of switchyard equipment such as transformers, disconnect switches and circuit breakers, as well as a realistic control room with the necessary control panels, let the operator trainee get a feel for the work he will be expected to do, while preventing dangerous exposure to any equipment as well as preserving network integrity.

Enhancements to the virtual world such as action sounds and the hum of transformers in the background make the simulation more realistic, while a speech recognition interface provides a way of interacting with the computer that is more natural for users who are less familiar with computers. A help mode using digitized images and videos guides the trainee through the course of his routine. The ultimate objective is to provide a training environment for switching station operators that will let them perform tasks in complete safety, but with a degree of realism that will thoroughly prepare them for the real world. By offering a rich training experience and by making human-computer interactions as natural as possible, the efficiency of the learning process is increased considerably.

2.4. System Architecture

2.4.1. Visual Simulation

In any VR application, visual simulation is generally the most demanding process in terms of computing power. For ESOPE-VR, a Silicon Graphics (SGI) Indigo workstation is used in conjunction with a 53cm (21-inch) 120Hz monitor to provide the user with a "window on the world" [Sutherland 1965], [Ware 1993].

CrystalEyes liquid crystal display (LCD) shutter glasses let us show the trainee an accurate stereoscopic image. With such glasses, the left eye of the user is obscured while the image meant for the right eye is displayed on the screen, and vice versa. Because the two, slightly offset, images switch every 120th of a second, the human brain is fooled into seeing a 3D view that has depth, rather than just a flat display. The shutter glasses have an advantage over more "immersive" visual systems in that they let the user look at other things around them, such as an adjacent computer monitor. A detailed tutorial on stereoscopic computer graphics can be found in [Hodges 1992].

The functions necessary for the visual rendering of the virtual switching station as well as the interactions with the virtual world are provided by a commercially available software product made for this purpose: WorldToolKit by Sense8. Essentially, WorldToolKit provides libraries of C functions that can be called from within other C code, and that allow for the creation of virtual worlds. User interactions with the virtual world, such as collision detection, as well as animation of objects in the world, such as the opening or closing of a disconnect switch, are also handled in software. The reason for choosing a commercial product rather than writing custom code for this application was one of simplicity -- the numerous functions available made the task of programming much easier.

This software allows for different accuracies of rendering (level of detail), meaning that far-away objects need not be displayed at the same precision as closer ones. The advantage of such a system is that it makes the simulation less compute-intensive, since fewer polygons need to be displayed, and consequently, recalculated. Depending on the available hardware, a low polygon count can offer significant advantages in terms of rendering speed. For example, an initial model of the switchyard had about 15,000 polygons, while a reduced-level-of-detail model contained one fifth of that number, resulting in a significant improvement in visual performance. Despite this fact, however, the rendering efficiency appears to suffer from the large overhead computing load WorldToolKit presents, and that improved performance could be attained with a customized rendering engine.

2.4.2. Electrical Simulation

Simulation of electrical parameters of ESOPE-VR is done by ESOPE's steady-state power flow simulator. ESOPE runs on a PC that is connected to the rest of the system by an Ethernet connection[3]. Any changes in the electrical network being simulated are relayed to ESOPE and updates of voltages, currents and power are then calculated. Presently, the Windows version of ESOPE presents a bottleneck for the rest of the system. Since the PC being used for ESOPE is much slower compared with the SGI workstations, there is a noticeable lag in the simulation every time the network parameters need to be recalculated. When a 120 MHz Pentium is used, some improvement is noticed, but a UNIX version of the program would be preferred.

2.4.3. Switching Operation Validation

Any switching operations that are performed in ESOPE-VR need to be validated if the simulator is to be used for training. In this case, the validation is also done by the ESOPE simulator. For example, if the trainee attempts to open a disconnect switch, the ESOPE-VR process manager will send an appropriate message to ESOPE requesting approval. If ESOPE decides that the user may proceed (as described above), the ESOPE-VR simulator will show the relevant changes in the virtual world. However, if the switching operation may not be done at this point, ESOPE will send back a message forbidding it, and an error message with an appropriate explanation will be provided to the

[3] The porting of ESOPE from the Windows environment to run under UNIX on an SGI workstation, could not be completed in time to be used in this project.

user. The alert reader will notice that this sequence of events implies that the trainee is being taught only "correct" work practices. He may not, for example, open a disconnect switch while magnetizing current is flowing through it, even though this may be a reasonable switching operation under certain circumstances. A mode of operation that would permit such actions is among the future plans for ESOPE.

2.4.4. Network Architecture

As mentioned above, the visual simulation aspect of a VR system is usually the most compute-intensive. In order to give that part of ESOPE-VR as much CPU power as required, it was decided to run it separately on one Silicon Graphics Indigo, while another SGI Indigo takes over the other aspects of the simulation, and the speech recognizer is connected to a PC. In order to solve the problem of finding enough computing power, the solution of putting several workstations in parallel was chosen. The hardware was readily available, and the cost of a more powerful graphics workstation would have been prohibitive. As is, all the necessary processes communicate with each other by means of text messages that are sent back and forth over the computer network. This sort of communication greatly facilitates testing and debugging of the system, as the messages are easily understood by humans as well.

2.4.5. Speech Input

A Verbex 7000 speaker-dependent speech-recognition system is used to control most aspects of the simulation in ESOPE-VR. This method of interaction is useful because it permits the user to free his hands for other tasks, and can help non-computer-literate users interact with the system much more easily. Also, if the user is already wearing the 3D glasses, and is perhaps navigating in the virtual world using the mouse, it is unreasonable to expect him to start typing on the keyboard as well. Several types of speech commands exist, each category being used to control a certain aspect of the program. In the switching operation category are commands that pertain to the training scenario, for example "*Open breaker 700-2*," "*Close disconnect 2B14*," "*Verify open state breaker 120-4*." Another category of verbal commands is used for navigation in the virtual world. Here, the user can change the direction of motion by saying "*Forward*" or "*Backward*," or can move rapidly from one location to another with a command like "*Take me to transformer T-2*." This last command is one of the luxuries of a virtual world – one can do things that are not possible in the real one. (For example, while the ESOPE-VR system is supposed to provide a realistic training environment, it is not necessary to make the trainee walk 20 virtual kilometres in order to accomplish his task.) Finally, speech commands can be used to control the various image and video displays, which are described in the next section.

The reason a speaker-dependent system is used is because it must understand French. Currently, speaker-independent speech-recognizers are available on the market from several vendors, but they all only understand English. Therefore, the Verbex system was used and was trained to correctly recognize French speech.

2.4.6. Digitized Image and Video Displays

One of the advantages of having Silicon Graphics workstations available is to be able to display a large number of digitized pictures and video sequences. Both these forms of output are used in ESOPE-VR to provide realistic images of the apparatus the user is training for. The trainee can use speech commands such as "*Show me a picture of breaker type X*," which causes a stored image of that breaker to be displayed on an adjacent monitor. The user could also ask "*Show me a video of the opening of switch type Y*," and would be shown a short video sequence that can be further controlled by voice commands such as "*forward*," "*reverse*," "*slow motion*," and so forth. Figure 2.2 shows an example of a digitized photograph and a video frame.

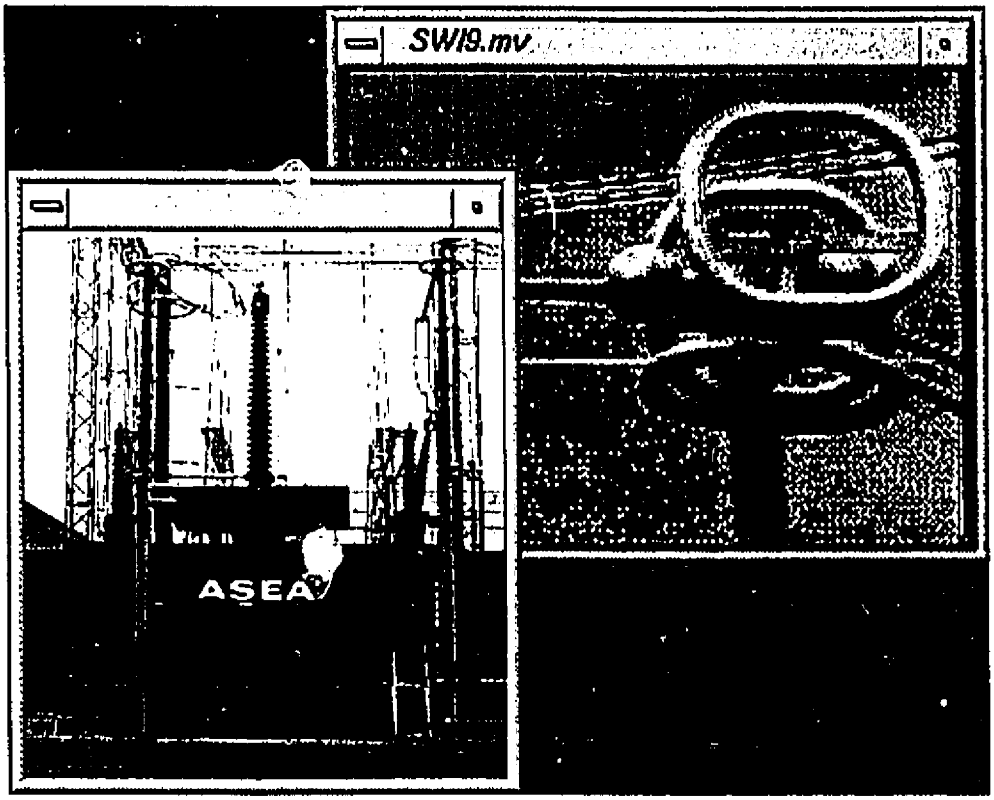

Figure 2.2: Digitized Video and Image Examples.

The pictures displayed in this particular implementation were taken by the researchers working on the ESOPE-VR project and subsequently placed on Kodak PhotoCDs. This media provides a simple way of transferring the images to any graphical software for manipulation and subsequent storage. The videos were recorded by portable video camera and digitized using the video input / output hardware on one of the workstations (IndigoVideo board on an Indigo R3000). These two approaches are reasonably simple and effective, but professional facilities can significantly improve the quality of the video images.

The display of digitized pictures and video sequences is useful as a reinforcement of the educational experience as it shows a completely realistic representation of actual equipment as opposed to the 3D computer model. Hopefully, in the near future, the accuracy of the model and the graphical capabilities of the computer being used will do away with such tools as they will have become unnecessary.

2.4.7. Interactive Help

At any time during training, the user can ask the simulator for advice on the current operation. For example, he can say "*Help about opening a breaker*," or "*Help with switching operation commands*." In either case, context sensitive help is given to the user by means of digitized voice messages. Approximately 45 help messages were recorded.

2.4.8. Aural Immersion

In any simulation, much information can be conveyed by means of visual cues. However, another important dimension of the "virtual" experience is auditory. In ESOPE-VR, a background-sound server provides the necessary "ambience." Two audio channels provide the background hum of transformers, each changing with respect to the distance the user is from the sound sources in the virtual world, and two further channels are available to produce "action sounds" ranging from the clicking of a button to the explosion of an air-gap circuit-breaker being opened. All these sounds are pre-recorded, digitized samples, and can thus be manipulated at will.

It would be desirable to provide an even more realistic aural "image" to the user by means of 3D audio localization technologies. These are devices that go beyond quadrophonic sound generation to provide audio signals that the listener can accurately position in space. One such system is Crystal River's Convolvotron which uses convolution of the audio signal with pairs of filters corresponding to measured impulse responses at the listener's ears [Wenzel 1988], [Kalawsky 1993]. Another possibility is to use raytracing techniques (usually used to generate computer images) to model the movement of sound [Shi 1993].

2.4.9. Automatic Conversion of Single-Line Diagrams to 3D Format

If a real-world switching station is to be constructed in the virtual world, several elements are required. These range from circuit breakers to disconnect switches, from transformers to strain and tubular buses. As well, a realistic control room with panels containing handles, buttons and meters needs to be modeled.

In the case of ESOPE-VR, the control room was modeled using a commercially available 3D modeller: Wavefront's 3Design. The output from this modeller could be read directly into WorldToolKit. In the control room there are two kinds of objects; dynamic objects such as buttons and handles can be manipulated by the user, or by the simulation in the case of meters, while static objects like the panels or the walls need not be changed during the simulation. Figure 2.3 shows a view of the virtual control room.

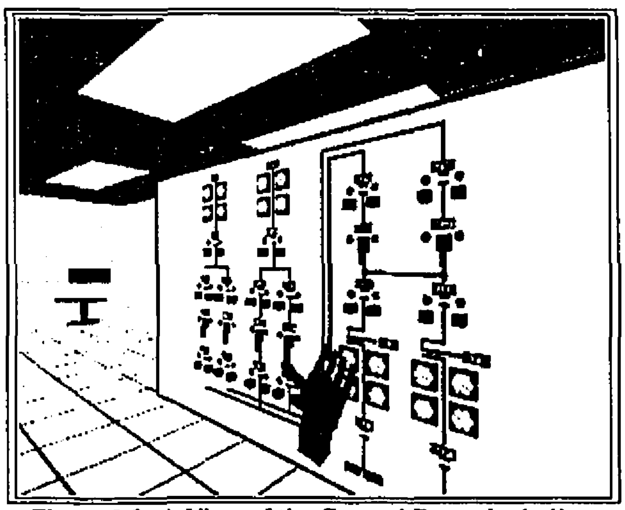

Figure 2.3: A View of the Control Room Including the Operator's Virtual Hand.

The modeling of the switchyard involves a different process. Starting from the single-line diagram used by ESOPE, a single-line-diagram converter program builds a 3D model of the switchyard. As the single-line-diagram is a representation of three phases of a circuit, and since the switching station's physical dimensions must follow certain guidelines with respect to spacing and layout, it is possible to make some reasonable assumptions to create a set of rules by which to construct the 3D model. This procedure does not presently allow for many exceptions. That is, the construction of substations that are restricted in their size by external constraints is not possible yet. The present converter simply builds the stations as large as it needs to in order to meet internal constraints such as the spacing of equipment.

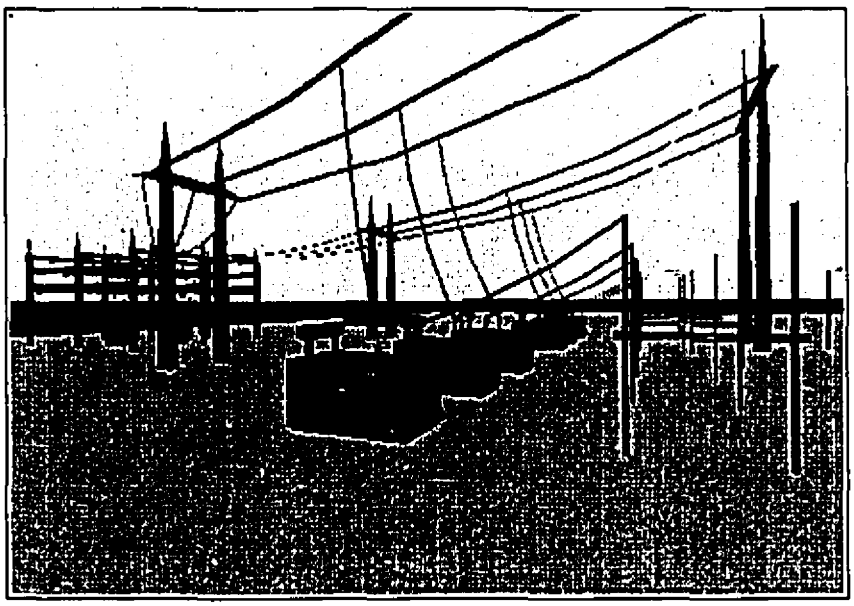

Figure 2.4: A Typical 735-120 kV Station Generated by the Single-Line Diagram to 3D Model Converter.

The placement of each piece of equipment is decided by the converter based on simple industry-specific rules in its database. As well, the necessary connections between the apparatus are provided in the form of busbars, i.e., either rigid or strain-bus. The connections outside of the station are also provided by the converter. The level-of-detail in the 3D model mentioned previously is incorporated at this stage. This process lets the simulation use three different models of the same apparatus which are displayed according to the distance of the person from it in the virtual world. Figure 2.4 shows a typical view of the switchyard.

The single-line-diagram converter is a very valuable tool in that it allows for the conversion of any number of substations, thereby allowing the provision of a training environment that is not location-dependent.

2.5. Summary

This chapter has described the functional aspects of the ESOPE-VR prototype operator training simulator. This system provides a realistic training environment for station operators, and ensures their safety and the electrical network's integrity. In the next chapter, a technical description of this project is given.

CHAPTER THREE

3. Technical Specification of the ESOPE-VR System

3.1. Introduction

This chapter discusses the technical implementation of the ESOPE-VR system. As mentioned in the previous chapter, the system is a client-server based system that is distributed over a variety of platforms. The choice to distribute the processes over several workstations and PCs was made in order to simplify each one's workload to increase the speed of the visual simulation, and to take advantage of the audio and video capabilities of an additional SGI Indigo workstation [SGI 1994].

3.2. Architecture Overview

One main idea behind the ESOPE-VR architecture is to distribute the computing load over several computers. The reason for doing so is twofold: for one, it helps certain processes run more efficiently, most notably the main graphical application which needs as much CPU time as it can get. The second reason is to take advantage of several workstations' audio hardware.

ESOPE-VR consists of several modules, some of which are described in detail in this chapter. The others were mainly the creation of others on the team, and form the focus of their respective M. Eng. / M. Sc. A. theses. Figure 3.1 shows a detailed diagram of the system's architecture.

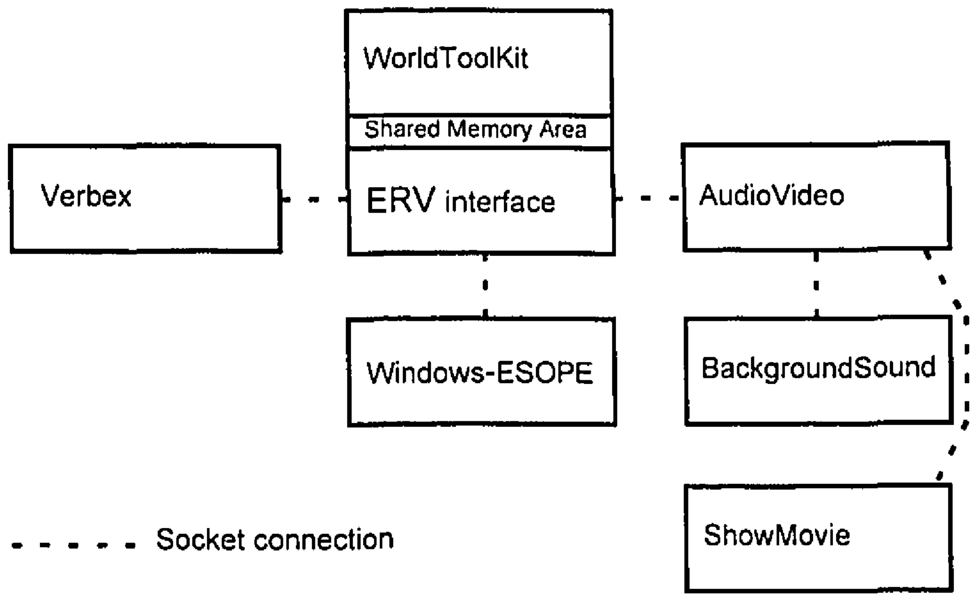

Figure 3.1: TCP-IP Client/Server Architecture.

The boxes in Figure 3.1 represent the following processes (starting at the left):

Name	Process Description	Platform
Verbex	Verbex 7000 speech recognition unit	PC
WorldToolKit	WorldToolKit graphical simulation	SGI
ERV interface	ESOPE-VR interface for WorldToolKit	SGI
Windows-ESOPE	ESOPE electrical simulation module	PC
AudioVideo	AudioVideo server	SGI
BackgroundSound	Background Sound server	SGI
ShowMovie	ShowMovie server	SGI

Table 3.1: Network Processes

The following sections describe the individual modules above and some of their characteristics in more detail.

3.2.1. The WorldToolKit Process

The most visible module of the system is the graphical rendering process. It was constructed using Sense8's WorldToolKit, a library of C functions that can be called from within any other C program. WorldToolKit provides the functionalities required for a VR application such as visual rendering of the virtual world, collision detection within that world, as well as animation of pieces of the world. In ESOPE-VR, the main WorldToolKit process also generates messages to the Windows-based ESOPE module to request that certain switching operations be performed. The messages are passed to the rest of the system through the ERV interface module, described later.

The WorldToolKit process contains the necessary code to be able to use the graphics screen of the workstation in stereoscopic mode. The CrystalEyes stereo glasses, presented in Chapter one, are used to give the user a 3D view of the world being rendered. Because of the stereo mode, no other processes can use the screen of the workstation used by the WorldToolKit process.

Navigation within the virtual world is controlled by various means. The main control for direction of motion comes from one of three different inputs. The simplest one is to use the computer's mouse as a 3D input device, defining forward, backward, left and right as customary, but changing the default directions to up and down when one of the mouse buttons is pressed, and to left or right when another button is pressed. The other means of navigating is by using a Logitech "Flying Mouse". This is an input device that provides 3D capabilities through ultrasonic transmitters and receivers. The major drawback of this device is its extremely poor ergonomics. Since it is much slimmer than a conventional computer mouse it is difficult to grasp comfortably, and holding it in the air, pointed in a particular direction, for extended periods of time is very tiring for the hand. Finally, a Powerglove interface was developed, which greatly simplified navigation and manipulation within the virtual world [Shaikh 1995], [Figueiredo 1993], [Sturman 1994].

As can be seen in Figure 3.1, the WorldToolKit process is not connected directly to the rest of the system, but uses a shared-memory interface to communicate. The reason this is necessary is because of the way the main rendering process works: once every frame, it checks all its sensors and updates the image accordingly. In the case of this project, the frame-update rate is approximately 10 frames per second, meaning 100 ms between frames. (This sort of communications is also known as "polling" – that is, the WorldToolKit process "polls" its inputs to see if there is anything waiting for it.)

Within the rest of the system, however, fewer processes are willing or able to wait for another one to become ready and to read the information that is meant for it. If this were the case, the whole system would come to a halt every 100 ms while the main rendering process sorts itself out and decides to proceed. Since it is not acceptable to wait in this situation, the ERV interface module accepts any incoming messages immediately and places them in a queue in a shared memory area where the WorldToolKit process can read them, raises a flag to indicate that there is some information waiting, and then continues to wait for more incoming messages. Now, the WorldToolKit process can take as much time as necessary to complete its frame-update, and can then react to the incoming information.

The alert reader will notice that there is a possibility of overflow if WorldToolKit does not read the messages quickly enough. While this is true, in practice it will not happen since any action in this particular system is triggered by an input from the user, and it is unnatural for him to stack up commands if the system is delivering error messages or awaiting simulation updates. A ten-message queue is used to gather incoming messages just in case several messages happen to arrive within the same 100 ms period.

3.2.2. The Windows-ESOPE Process

The next important module in the system is the ESOPE process itself. It is responsible for calculating the electrical parameters of the given simulation by a steady-state power-flow program. As well, it provides expert-system support for the trainee by validating any switching operations requested by the WorldToolKit application. An existing version of the ESOPE program was modified to allow it to communicate with the rest of the system via the computer network. Other than the overview of ESOPE given in Chapter two, more extensive information about this part of the system can be found in the other theses about this project.

3.2.3. Speech Input

The Verbex 7000 Voice Input / Output System is used as a speech input system for voice commands. As the system is a speaker-dependent one, it can be trained to understand any language – in this case, French. The advantages of a speech interface are twofold: it frees the user's hands from the keyboard, and it is a more natural way of communicating for someone who is not necessarily accustomed to using computers (as an operator trainee might well be). As the VR interface is supposed to be friendly to the user, it is important to simplify the task of training as much as possible. The Verbex unit uses a grammar file containing the necessary vocabulary for recognition. A hierarchical structure of commands is organized into four categories:

- System
- Navigation
- Operate
- Multimedia

Help is available to the user in each of these categories by means of pre-recorded digitized messages which are played back by the AudioVideo Server, for example, "*Aide ouvrir sectionneur*". The system commands are used to control aspects of the animation such as the type of rendering, zooming in or out, pausing, resetting the program or quitting. Navigation commands include changing the default direction of motion from forward to backward, stopping, and turning left or right. As well, the user can request to "teleport" to a certain location, for example to the control room or the switchyard, or to a specific piece of equipment ("*Téléporter disjoncteur 700-4*"). The switching operation commands include pushing a button specified by the name of the disconnect switch it controls, turning a piece of equipment (disconnect switch, circuit breaker) on or off, turning handles, and verifying the current operating conditions. As well, the operator can specify the current time by speaking it ("*Il est minuit quarante-trois*"). Finally, in multimedia mode, the user can ask to be shown digitized photographs of various types of equipment and short digitized video sequences detailing certain switching operations. The word "multimedia" was selected in this case for its ease of use, not because we are trying to create a multimedia presentation – the beginning of Chapter one gives an explanation why.

The Verbex speech interface is connected by a serial cable to one of the PCs. The reason a PC is used is one of availability: the interface programs for the speech recognizer are all DOS-based. Thus, a first step is to initialize the Verbex from DOS, then the PC is rebooted with Linux. (Linux is a public-domain version of UNIX for PCs which is compatible with the version of UNIX used on the SGI workstations.) The PC is also connected via TCP/IP socket connections to the AudioVideo, WorldToolKit, and ESOPE-VR modules running on the SGI workstations. A program running under Linux monitors the serial port activity and captures any incoming characters. The characters received are concatenated until a carriage return character is received. The string thus received is then parsed to determine where it should be sent, and is finally written to the destination program via the socket connection. The program running on the Linux-PC also uses the concept of hierarchical sub-menus like in the Verbex grammar file. The switch between modes is made anytime the user speaks the name of a category, or immediately after a help request is received.

The speech recognition module provides a comfortable, easy-to-use way of interacting with the VR simulator. It uses a speaker-dependent recognizer to enable the user to interact within a hierarchical command structure.

3.2.4. AudioVideo Server

The AudioVideo server program acts as an interface for the Background Sound and ShowMovie servers and handles all audio and video requests from the speech recognition, ESOPE and WorldToolKit modules. Upon request, it plays help messages, error messages, and action sounds, and displays digitized pictures. Requests for background noises or the display of a movie file, as well as commands that control movie playback, are passed on to the Background Sound server or the ShowMovie server. The AudioVideo server must run on a separate machine from the WorldToolKit animation, as it requires an X-display for itself, and cannot be used in stereo graphics mode. The AudioVideo server has five socket connections. These are detailed in Table 3.3.

Socket	Function
1	Receive messages from Speech Interface
2	Receive messages from ESOPE
3	Receive messages from WorldToolKit
4	Send messages to Background Sound Server
5	Send messages to ShowMovie Server

Table 3.3: Socket Connections to AudioVideo Server

The following sections present in more detail the various categories of requests handled by the audio / video servers.

3.2.5. Speech Output Messages

Requests for help messages are received from the Verbex speech interface. The command is received as a string, and is parsed to see if it contains a help request ("HLP"). If so, the rest of the string is concatenated together without spaces to create the name of the audio file containing the help information. For example, if the string "`VBX A_V HLP OPR CLS BRK`" is received, the name of the file is `OPRCLSBRK.aifc`. (AIFC stands for Audio Interchange File Compressed, a format developed by Apple Computer [Apple 1991].) This filename is concatenated with the default path for the audio files, the name is verified for validity, and if the name is valid, the system command `sfplay` is called to play the file.

Error messages are handled in the same way, except that these requests come from ESOPE via the ERV interface. The error messages from ESOPE are numerical, and so, for simplicity, this standard was adhered to in the naming of the error message files. For example, if the message "`ESO A_V ERR 103`" is received, the name of the file is `ERR103.aifc`.

The Verbex 7000 Speech Recognizer has the ability to translate ASCII strings into phonemes and to speak them through the headphone speaker. However, it only "speaks" English, and has considerable difficulty dealing with French speech. An attempt was made to use "pseudo-French" written text composed of English phonemes that approximate French speech, but this experiment was disappointing. The reason for the failure is twofold: for one, there are many French phonemes that do not even exist in the English language, such as the nasal n in "*longue*", and second, even if the phonemes do exist, it is difficult and tedious to estimate a reasonable-sounding French word or sentence when writing only in English words. For example, one could write the sentence "*les choix sont les suivants*" as "*lay shwa son lay sweevunn*", but even this is a crude approximation, and does not sound very convincing. For this reason, it was decided to use digitized speech stored on the computer's hard disk as a solution for the problem of giving the user speech feedback. Since the application requires only a certain fixed number of messages to be spoken, and does not have random sequences of words to be synthesized, it is simple enough to record these messages once, and then use them on demand. Memory requirements for the various digital media are discussed in the next chapter.

3.2.6. Digitized Images and Videos

Requests to display digitized pictures are dealt with in a manner similar to audio requests. The filename is determined from the message, and the program `xloadimage` is used to display the file. The display of the picture is not controllable via a socket connection, since `xloadimage` is an independent application. Instead, in order to terminate the display, the user must press "q" or "Ctrl-C" while the mouse pointer is over the window displaying the image. However, to keep the screen display from becoming cluttered and confusing, only one picture or movie is displayed at once. If a movie command or another picture request is received while a picture is being displayed, the currently displayed picture is removed by killing the `xloadimage` process. If a picture request arrives while the movie is being displayed, the movie window is iconized, so that it does not interfere with the other window. As mentioned above, other requests are passed on to the ShowMovie server or the Background Sound server.

3.2.7. ShowMovie Server

The ShowMovie server program must also run on a separate machine from the main WorldToolKit animation, since it uses the screen display for itself, and cannot be used in stereo graphics mode. However, the AudioVideo server and the ShowMovie server can run on the same machine, as their displays do not conflict significantly. (They may influence the colour map the display is using, or a new image may destroy a previous one as mentioned above.)

The function of the ShowMovie server is to display SGI movie files on the work-station's screen, and to let the user control the movie's playback. The movie files in this case were converted from VHS videotape to SGI movie format using the SGI SVideo tool kit along with the IndigoVideo board in one of the machines. Initially, the server waits for the command to load a given movie file. Once it has received a valid movie file name, it opens a window and displays the first frame of the movie. It then uses the `select()` function to wait for events from the movie itself, from the X server, and from the socket connection to the AudioVideo server. Upon receipt of a socket event, the received string is parsed to determine the type of command. The command is executed using the audio and video library commands available [SGI 1994], and the program returns to waiting for another event. The ShowMovie commands are summarized in Table 3.4.

Command	Action
Play	Plays the movie in the default direction
Stop	Stops movie playback
Reverse	Plays movie in reverse
Slow-Motion	Plays movie at ¼ of normal speed
Rewind	Resets movie to frame 1
Loop Toggle	Toggles between "play once", "loop forward / backward" and "swing"
Mute	Sets movie audio playback volume to zero
Increase Volume	Increases audio gain by 25 units (out of 255)
Decrease Volume	Decreases audio gain by 25 units
New Movie	Stop current movie's playback, and wait for a new movie name

Table 3.4: ShowMovie Server Command Set

3.2.8. Background Sound Server

The Background Sound server must run on a different machine from the AudioVideo server, as it uses the audio hardware for itself, and would therefore negatively influence the playback of help or error messages or the movie's audio track. It can, however, run on the same machine as the WorldToolKit animation, as the two do not influence each other directly.

It was necessary to split the background sounds from the other action noises and audio messages because the SGI audio hardware imposes certain limitations. For example, while it is possible to play up to four sounds concurrently, it is not possible to control their playback volume individually using high level functions[4]. As well, it is not possible to play a given sound on a given channel individually; rather, one must comply with SGI's way of doing things. When one sound is playing, and it is necessary to change the output gain to play another one at the same time, the first sound's amplitude is also increased, creating a somewhat unreasonable situation. For example, if a the hum

[4] Using low-level programming, almost everything is possible. It was suggested to add up the individual sound samples (16 bit shorts), multiplied by an appropriate gain, and to send them directly to the audio hardware. However, in this particular instance, this was not done.

of a transformer is playing in the background, and a loud bang is to be played as an action noise, the transformer noise must, even for a short time, become as loud as the bang, and then return to its previous level. This situation is clearly unacceptable, and thus it was chosen to split the audio playback to two workstations, where each volume setting is controllable individually.

In this case, one workstation handles the help and error messages and action noises, which all don't influence each other, and the other workstation plays a maximum of two concurrent background sounds. A further limitation imposed by the SGI audio hardware is that a sound file that contains only one channel (mono) is played equally on both left and right channels, while a file containing two channels (stereo) is played with one channel's samples being played on the right output, and the other channel's samples on the left output. To overcome this limitation, and to enable the use of two distinct background sounds, it is necessary to record one of them only on the left channel of a stereo file, and the other one only on the right channel. Then, by regulating the playback volume of either channel, the volume of that particular background noise can be regulated.

This drawback imposes the limitation that the channel a given background sound is to be played on must be pre-determined, and thus, the second sound must exist in a version for the other channel. In order to simplify the choice of output channels, the following limitation on background sound filenames was imposed: they must end with "_L" or "_R", for example XFO3_L and XFO3_R. This way, there is no confusion over what is happening. Both the background noises and the action sounds are played at a volume that is appropriate for the user's position in the simulation. That is, both the Background Sound server and the AudioVideo server receive information about the sound source's location and the user's location in the message requesting the sound. The distance between the two is then calculated, and is used to set the output volume accordingly. In the future, an implementation using three-dimensional sound could use the same data to construct an appropriate sound "image" for the user. Sound intensity should vary as $\frac{1}{D^2}$ where D is the distance from source to listener. In this case, we assume that the sound is at intensity 255 at the source, and dies down from there, so $I = \frac{255}{D^2}$. However, since the "gain" of an SGI audio channel is not linear, one can approximate the appropriate calculations by using $I = \max\left(\frac{255(attenDist - D)}{attenDist}, 0\right)$ as an approximation where the sound intensity decreases linearly to zero at a distance of *attenDist* units. In the case of a background noise, a distance of 50m was used for *attenDist*, while for an action noise (such as a button click), the distance was set at 10m.

Beside the visual rendering of the ESOPE-VR system, the audio and video outputs are an important part of what is necessary to stimulate the user's senses. Aural immersion is achieved with action sounds such as button clicks and explosions and background noises such as transformer hum, which is adjusted in intensity according to the distance from the user to the sound source.

3.2.9. Communications by Message Passing

Since the ESOPE-VR system consists of almost a dozen separate processes running concurrently on several machines, a scheme had to be developed for communicating between these distinct units. What was found to be the simplest and easiest to implement and debug was the use of ASCII[5] strings to send messages from one unit to the other. In order to debug a particular server program, all that is required is to connect to the given machine with a `telnet`[6] connection and to "talk to" the program by means of text messages that are understood both by it and by humans. For client programs, the interaction is a bit more complicated: a "dummy" server program has to be created and

[5] ASCII = American Standard Code for Information Interchange.
[6] telnet is one of several types of connections that can be made between UNIX workstations.

started before starting the client that is to be tested. The server program can then be used to "pass through" information to the client.

Socket connections are used between most of the individual modules to transfer messages from one process to the other. This type of connection was found to be very practical to use, as it allows us to start the processes on any number of different machines. Further, it is a reliable and fast form of communication, and was chosen over more high-level communications protocols such as remote procedure calls (RPC). When two processes run on the same machine, communication is done by means of a shared memory area. This is the simplest choice, and the fastest one, too. Since shared memory is a physical thing (as opposed to socket connections), it cannot be used to communicate from one machine to another.

One important factor is that the processes have to be started in a certain order so that everything will run properly. A script reads the system configuration from a file and then starts the processes on various machines in a pre-defined order. This file is included in Appendix 2 on page 82. The servers need to be started first, and then the clients. In this case, for example, the Background Sound server and the ShowMovie server are started before the AudioVideo server, which in turn, has to be started before the main WorldToolKit application or the speech recognition process.

Once all connections have been made, communications between the modules takes place by text messages containing the name of the sender and the receiver followed by the data being sent. Table 3.2 shows some typical messages.

Source - Destination	Message Meaning	Message Text
Verbex to ESOPE-VR	"operate: close breaker"	VBX ERV OPR CLS BRK
Verbex to AudioVideo	"load movie of switch type 1"	VBX A_V LOA SWI 1
ESOPE to AudioVideo	"Error code 1234"	ESO A_V ERR 1234
WorldToolKit to AudioVideo	"User's position is 23 45 0"	WTK A_V POS 23 45 0
ESOPE to ESOPE-VR	"Voltage line 1 is 123 kV"	ESO ERV VLT 1 123

Table 3.2: Sample of Messages Passed Between Modules

3.2.10. Graphical Progress Monitor

It is often useful to generate debugging messages giving various details about the progress of a running program. In this case, however, there are almost a dozen separate programs running, each generating their messages on the same screen[7], and so it can become complicated to understand what is happening. Therefore, a graphical monitor is used to trace the progress of setting up all the required programs and their subsequent communication.

The program, written using various GL[8] functions, runs on one of the SGI work-stations that is not running the main WorldToolKit animation. A graphical approach to the problem of progress monitoring was found to be useful, mainly because of the amount of information that is to be processed. Each program sends out at least two progress messages per socket connection (of which there are almost two dozen), and it is therefore difficult for a human to read them all and to deduce the fact that everything is running smoothly, or that there has been an error.

A more detailed description of the monitor program's function is found in the next chapter.

[7] All processes are started from a common point using the startup script, so all their outputs go to the same screen. This is described in more detail in the next chapter.
[8] GL = graphics language, a computer graphics standard developed by SGI.

3.3. Summary

This chapter has presented an overview of the technical aspects of the Esope-VR operator training prototype. The system is distributed over a variety of platforms, namely SGI Indigos and PCs, and consists of a main process based on WorldToolKit, as well as an AudioVideo process, a background sound server, a movie server, the original Esope module and a speech recognizer. A script is used to start and shut down the system, while a graphical monitor provides visual feedback on the state of each of the processes.

CHAPTER FOUR

4. Detailed Implementation and Test Results

4.1. Introduction

This section presents an evaluation of the implementation described in the previous chapter and summarizes feedback from a user questionnaire.

4.2. Speech Recognizer

The Verbex 7000 is a speaker-dependent speech recognizer. As such, it needs to be trained by each person who wants to use it. The training procedure is quite straightforward. The following steps need to be taken:

4.2.1. Grammar, Recognizer and Voice Files

• A grammar file has to be created using any ASCII text editor. This file contains the words and phrases that will subsequently need to be recognized. With a speaker-independent recognizer such a step would be either unnecessary or required only to a very limited degree, such as for personal additions to the existing grammar.

• This grammar file needs to be converted to a "recognizer file" by a utility program. In order to recognize words and phrases, a machine format that can be transferred to the Verbex unit has to be created.

• The recognizer file is transferred to the Verbex 7000 by the serial link. As mentioned previously, the Verbex is a DOS-based system as far as the available software is concerned. Therefore, the conversion and transfer are done using two utility programs running under DOS.

• The Verbex 7000 needs to be trained - first the individual words in the grammar file, then the phrases that occur. Training typically requires the user to speak each word three times. The phrases that occur in the grammar are usually of a format such as "*word1 word2 number[1-10] word4 word5*" or something similar. In such a case, it is not necessary to speak all ten possible phrases three times each. Rather, the recognizer will select those phrases that are most likely to cause confusion, and will repeat them several times. All of the parameters such as word training frequency can be selected by the user.

• The "voice file" that is generated by training must be transferred back via the serial link, so that it can be saved and re-used later.

4.2.2. Grammar File Structure

Figure 4.1 gives a state chart representation of the grammar file used for the ESOPE-VR project. There are five main categories of commands represented: base, operate, navigate, system, and multimedia. In each of these categories, there are other commands. The overall structure can be seen as a type of menu system, except that it is voice-based. From each of the categories, help is available by saying "*aide-moi*."

The transitions between the individual states are labeled with the key words to be spoken. In the case of the help commands, the transition back to the corresponding state is made automatically after the help request is recognized. For example, if a user is in the "system" sub-menu and wants help on resetting the simulator, he would first say "aide-moi", then "*aide reset*". The system would give him an appropriate verbal message about how to reset the simulator and would then place him back in the "system" menu.

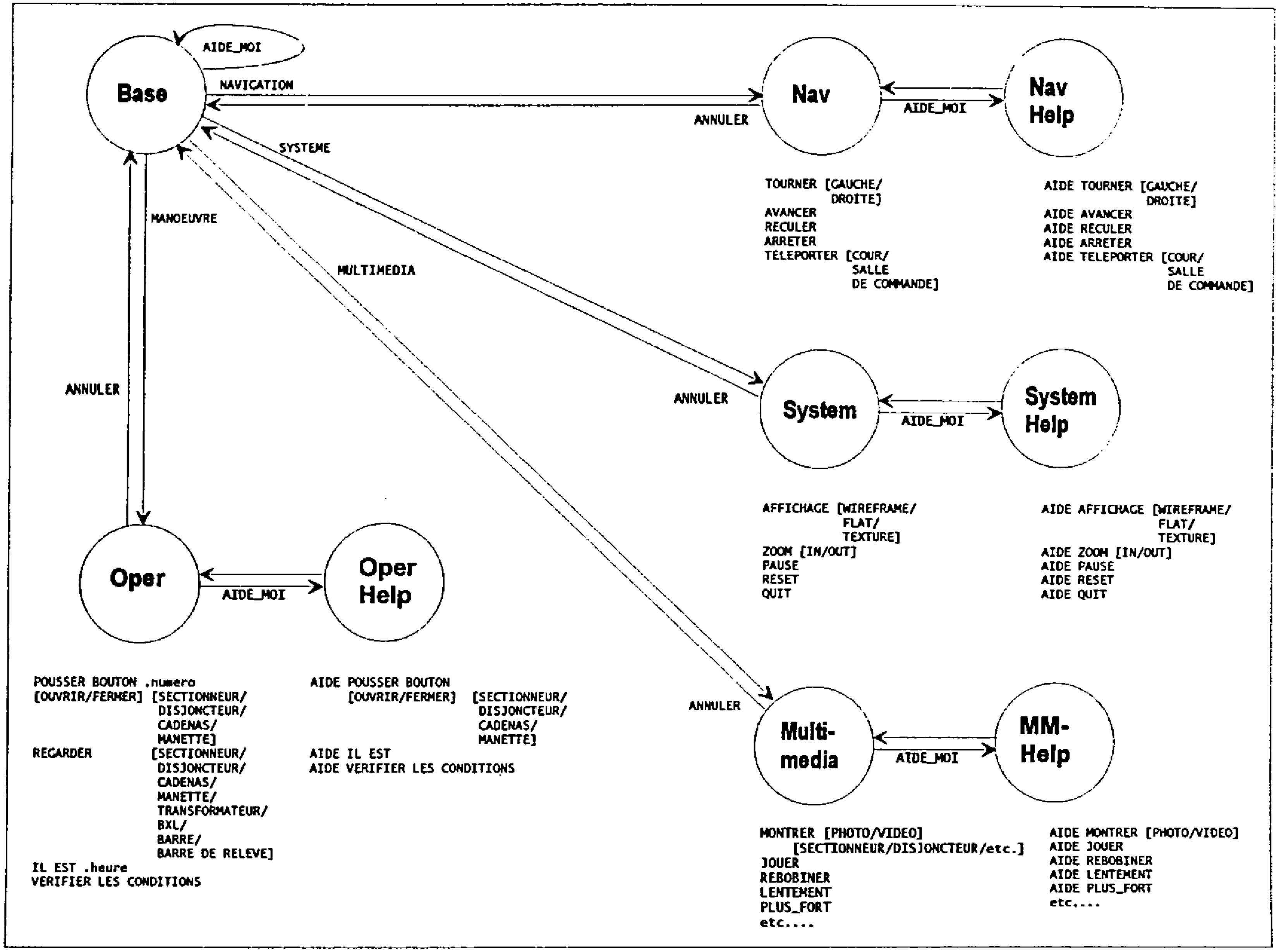

Figure 4.1: Verbex Grammar Flowchart

Note that in order to simplify the diagram, certain transitions are not shown: it is in fact possible to go from any menu to any other by speaking the appropriate keyword. For example, one can switch directly from the "operate" menu to the "navigation" menu by saying "*navigation*" without having to pass through the "base" state.

4.2.3. The Training Process

The method of training the Verbex 7000 speech recognizer is straightforward. The interface between the user and the recognizer is yet another utility program which presents the information on the PCs screen.

First, the user is required to calibrate the recognizer to "silence" – that is, a level of background noise that would typically be present during use of the unit. In the case of ESOPE-VR, we always attempted to have a silent background. It was found subsequently that a noisy environment made accurate recognition very difficult. In this case, as mentioned previously, a keyboard interface that mirrored the Verbex grammar was written to serve as an alternative input mode. This was found to be very useful, particular in demonstrations of the system to groups of people where it is unreasonable to require silence. In a future implementation, the issue of error-prone recognition must be addressed. One suggestion, which should also be tested, is to include a certain amount of controlled, low-level, background noise during the training [Morris 1993]. While it is not clear what the consequences of doing so would be, one thing that was noticed during training was that punctual background sounds during training made subsequent recognition almost impossible. This is due to the noise being included in the machine's representation of the particular word or phrase.

Next, the recognizer prompts the user to speak individual words from the grammar to be trained. The words are run through in alphabetical order. It is sometimes difficult to pronounce the word correctly since one happens to know or does not know the context in which the word is to be spoken. In general, the best approach was found to be to speak each word naturally as a single unit. After all the words have been trained individually, the user must speak the phrases. The recognizer selects the appropriate combinations, as noted above, and warns the user of any misrecognitions. The recognizer can be told to ignore these, and to re-train the phrase in question.

The time needed to fully train a certain grammar varies with the grammar's complexity and the number of words. For example, in an initial try, a grammar was created that enabled the user to speak any time of day from midnight to midnight in one minute increments. Thus, it was necessary to have many different combinations of numbers available. The complexity of this particular grammar was quite high, and therefore the training was unnecessarily long - close to three hours, of which a great part was spent speaking phrases such as "*Il est vingt-deux heures quarante-cinq.*" This task was quickly judged to be much too arduous, and so the possibility of noting the time was eliminated. This cut the training time to a more reasonable half-hour or so. Of course, the training time varied considerably from person to person simply because of the clarity of their pronunciation and the error rate.

The complexity of the particular grammar is measured numerically. A high number (greater than 100) implies a complicated grammatical structure and therefore slower, more error-prone recognition. In the two cases mentioned in the previous paragraph, the complexities were around 215 and 50 respectively.

4.2.4. Verbex Questionnaire

A total of seven people used Verbex over the course of a few months. Five of these were in the context of ESOPE-VR; the two others were undergraduate students who used Verbex for their course projects.

Only two of the people had used Verbex or any speech recognizer before, so for the others, this was "new" technology. In general, the responses about the individual experiences using the speech recognizer were positive. Some of the people spoke to

Verbex in an "unnatural" manner ("radio voice") until they were stopped and told to speak normally. Some people also tried several pronunciations of the same word without being told to do so. It is not clear whether this is helpful or not; probably a consistent pronunciation throughout would be better.

A questionnaire was given to each of the people who used Verbex. The questions and the answers are included in Appendix 4 on page 88.

The responses to the questionnaire are generally positive and informative. In general, after a short acclimatization period, people found the Verbex quite easy to use. Apart from difficulties dealing with background noise levels, and a few misrecognitions, no major problems were encountered. The questionnaire results would indicate that a voice recognizer is quite suitable for virtual reality applications. The suggestions made here should be taken into account in a future implementation.

4.3. Socket Connections

Since all the individual programs making up the ESOPE-VR system communicate by socket connections, it is important to know what sort of performance can be achieved with this technology. A test program was written to determine the speed of the socket connection between two given processes: in order to accurately measure the length of time necessary to transfer a given amount of data, a 50-character string was sent from one process to the other, and then echoed back to the first. This was repeated 1000 times in succession, and the time measured from start to finish. While the time did vary somewhat from one try to the next, the minimum and maximum times observed ranged from 2.2 to 6.3 seconds for all 1000 messages. Thus, the number of messages per second ranged from 317 to 909, or 15900 to 45450 characters per second (1000 messages each way = 2000 messages). Note that this measurement takes all the code overhead into account, but this was also present in the actual application using the socket connections. One should also take into account that these data do not represent the absolute speed of a socket connection, but rather that of a typical connection in this application. One can just as easily transmit 1000 characters at a time, one thousand times, in almost the same time as 50 characters one thousand times. The communications speed achieved by the socket connections is more than adequate for the ESOPE-VR application. Since only relatively small amounts of data need to be sent over the network at any one time, and the speed of the simulation is dependent on the visual rendering of the virtual world, there is no delay introduced by this form of data transfer.

Figure 4.2 shows the function calls required to establish a socket connection. The server first has to create the socket, then bind it to a given address, and then listen for connection requests. When a connection request is sent by a client using the `connect()` system call, the server uses the `accept()` system call to establish the connection. At this point, `read()` and `write()` function calls are used to communicate between the processes. In order for the programs not to block waiting for input, they use the `select()` function to multiplex inputs and outputs.

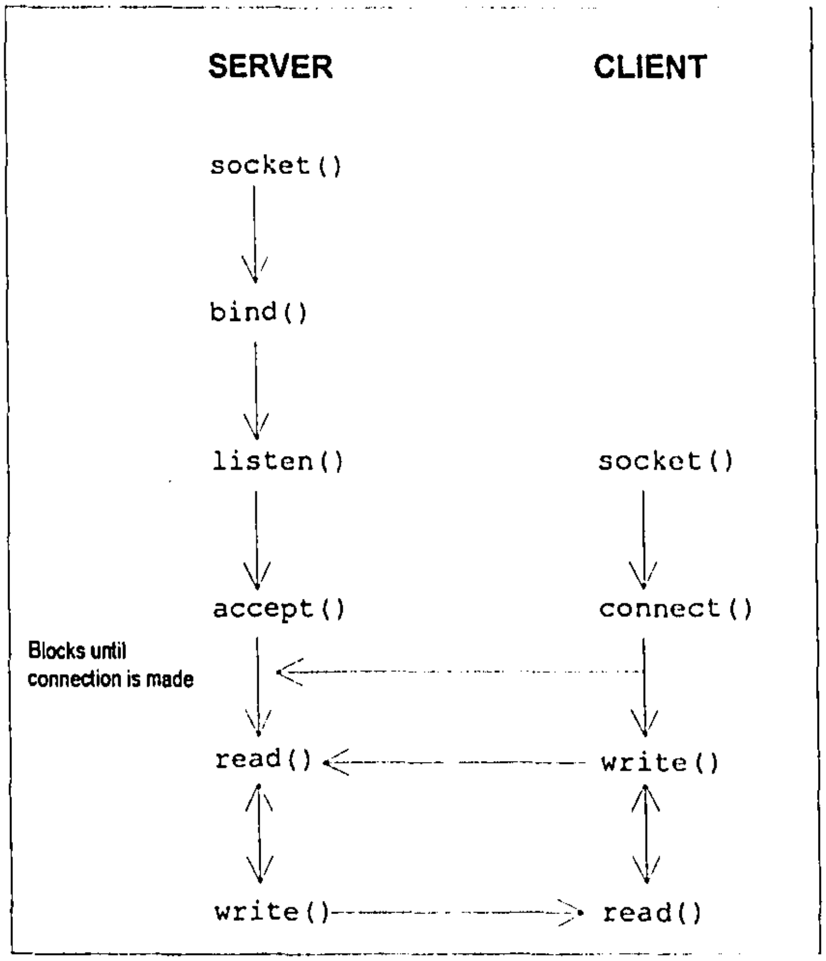

Figure 4.2: Establishing a Socket Connection.

4.4. Audio and Video File Formats and Sizes

For the ESOPE-VR project, digitized audio files were used to relay help and error messages to the trainee. It was decided to use this approach rather than a speech synthesizer because of its high quality output. Only a limited number of messages had to be recorded, rather than synthesizing random word sequences. Speech was digitized using 16-bit samples at 16 kHz using the SGI's audio interface. This frequency was chosen as the best compromise between acceptable quality (8 kHz bandwidth) and memory requirements (32 kB/second at 16kHz). (The total space requirements for audio and video are summarized in Table 4.1.) A total of 44 help messages were recorded, ranging from 2.6 to 21.8 seconds in length, with an average of 6.5 seconds. 38 error messages were also recorded, these being 1.9 to 6.6 seconds long, with an average of 3.6 seconds. Sound effects were also recorded: 12 files ranging from 3.1 to 6.2 seconds were used in the project. These sounds include the background hum of transformers (one per channel) and action noises such as button clicks or motor or arcing sounds.

The images presented to the user were taken with a conventional 35mm camera and subsequently placed on a Kodak Photo-CD. This enabled us to transfer them quite easily from that media to the format used for ESOPE-VR, namely the GIF format. It was decided not to use JPEG compression due to the time necessary to decompress and display each image. For demonstration purposes, seven images were used.

The videos used were recorded onto VHS tape and digitized using the Indigo Video board on one of the SGI workstations. It was decided to use a frame size of 320 by 200 pixels as a reasonable compromise between display speed and image clarity. A frame rate of 8 frames per second (fps) was selected in order to keep the disk space requirements low. Each frame requires approximately 80 kB, consisting of 64 kB (320x200x8 bits/pixel) for the image as well as 16 kB (8-bit samples at 16 kHz) for the soundtrack. The image quality is certainly not the best available, and this aspect would have to be changed for subsequent implementations. It would be interesting to look into the availability of MPEG player hardware, which would allow for a great reduction in disk space due to the compression techniques used. As can be seen in Table 4.1, the SGI movie files take up quite a bit of space. Even with current disk technology, however, there is ample room for scaling up by a factor of 1000.

File Type	Total	Sampling Rate	Storage (kB) per second / frame	Total Storage (kB)
Error messages	38 = 139 s	16 kHz, 16 bit	32 kB	4428
Help messages	44 = 285 s	16 kHz, 16 bit	32 kB	9135
Sound effects	148 s	16 kHz, 16 bit	32 kB	4732
SGI movies	139s = 1112 frames	8 fps @320x200, plus 16 kHz, 16 bit sound	80 kB	89869
Digitized images	7 images	8-bits per pixel	varies	1607

Table 4.1: Disk Space Requirements for Various Digital Media

4.5. Startup / Shutdown Script

A script written in *perl* is used to start up or shut down the system. *Perl* is a script language that combines many functions otherwise found in the UNIX shell, in the C programming language and in various UNIX utilities. It is this combination of capabilities that makes *perl* useful. It is much easier to do so this way than to launch or kill the programs individually. Since there are six processes to be run consecutively in exact order on various machines, having a centralized "starting point" is almost a necessity, and is much simpler than trying to coordinate matters otherwise. The script is included in Appendix 1 on page 76. The script is "pointed to" by two symbolic links called "*start*" and "*term*", respectively. The script determines what name it is being referred to as when it is called, and then acts accordingly, either starting the simulator or cleaning up the running programs. It reads information from a configuration file to determine which processes are meant for which machines. The environment variable `$ESOPE_HOME` is used to determine in which directory the configuration file *config.erv* is located, and must be set on each machine. For each line in *config.erv*, either a remote shell (`rsh`) or just a background shell on the local machine is run. As well, certain command-line arguments can be passed to the script. These are detailed in Table 4.2.

Parameter	Function
-c	Run the simulator with stereo graphics
-d	Debug the *perl* script
-e	Use ESOPE in echo-only mode
-f	Do a fast restart of the simulator
-h	Print the available choices
-k	Use the keyboard interface for Verbex
-l	Use the Logitech 6-D mouse

Table 4.2: Command-Line Parameters for Startup/Shutdown Script

After the simulation has ended, the same script can be used to "clean up" the still running processes. In this case, the script is referred to as "*term*". The script first runs a shell or a remote shell on the specified host to determine the process ID of the running program. It then runs another shell to send the programs a pre-specified kill signal, which causes them to exit gracefully, cleaning up after themselves. Here is an example line from the configuration file:

```
1 lagrande   /bin  audiovideo  5  yes  -a  -b5
```

This line specifies that the program "audiovideo" is to be executed ("yes") with the parameters "-a" and "-b5" on the host "lagrande", and that this program is found in the directory `ESOPE_HOME/bin`. A pause of "5" seconds will follow the start of audio-video, and therefore the launch of the next program.

4.6. Graphical Monitor Program

As mentioned in the previous chapter, a graphical program is used to monitor the progress of system startup as well as subsequent communications between the individual modules. Three colours (white, green and red) are used to show the state of each program that is to be launched. The programs are represented as boxes with their respective names, while the sockets are represented as small squares with lines connecting them. A picture of the graphical monitor window is shown in Figure 4.3. The following is a description of the monitor's use.

Initially, all lines and boxes are shown in white. When a program attempts to make a socket connection, the socket and its link to the other process are coloured in red, and when the connection has been made, the sockets and their connections are shown in green. Finally, when all the connections to a given program have been made, that module is shown in green as well. It is therefore simple to recognize errors that may have occurred simply by seeing that not everything is coloured green, and furthermore, the source of the error is simple to determine, as it can be readily identified on the graphical representation.

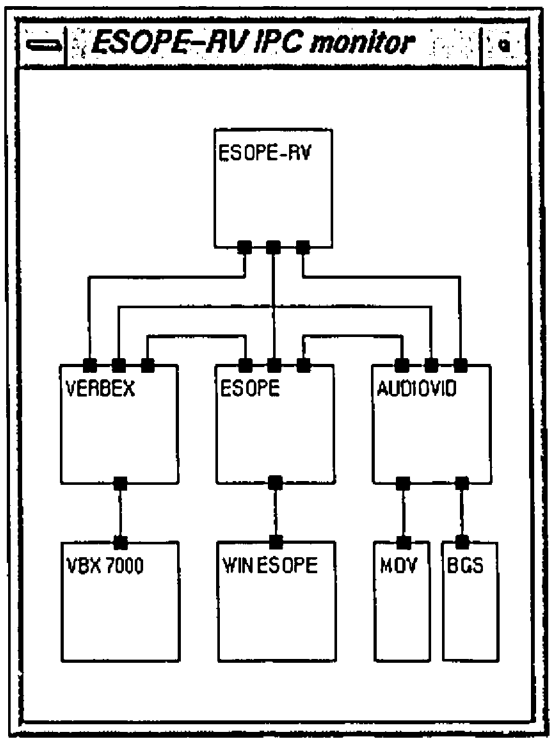

Figure 4.3: Graphical Progress Monitor.

The monitor program actually consists of two separate processes running on the same machine, connected by a shared-memory area. While one process is responsible for the graphical output, the other continually parses the strings it receives through its standard input for certain tokens which signal the activities that are of interest. For example, the token ">>>>" represents the fact that a socket connection has been initiated by one of the programs, and the token "<<<<" means that the connection has been made. The token "=>" represents the sending of some data from one process to the other. Preceding and following these tokens are other strings that uniquely represent the sender and the receiver of the messages, for example "VBX >>>> ERV" or "ESO >>>> A_V". If one of the three tokens mentioned is received, the parsing process places the entire received string in the shared memory area for retrieval by the graphical monitor process. Since the parsing process uses its standard input to receive data, it is a simple task to use the graphical monitor program. The only requirement is that all the programs that are being launched with the startup script write progress-report messages of this type to their standard output. The graphical monitor program is started as follows:

`% monitor`	• the graphical output program
`% sleep 5`	• this lets the above program start and initialize the shared memory
`% start | monitor1`	• "monitor1" is the utility program for the graphical output program, and "start" is the startup script. The symbol "|" represents a pipe - meaning that the output from one program is sent to the input of the other.

The graphical output process runs in a tight loop, continually checking for an incoming message in the shared memory area, but also updating the display as necessary, for example, if the window is temporarily covered, the entire picture is re-drawn when it is uncovered again. The reason a shared memory scheme is used is so that the process does not have to block when waiting for input.

The graphical network monitor was found to be a very useful tool in debugging the ESOPE-VR system. It is much simpler to read a three-colour graphical display than to have to sort through dozens of output messages in order to determine what has gone wrong. On the other hand, it is satisfying to see an entirely green display and to know that everything is working as it should.

4.7. Recommendations for Future Work

The implementation of the ESOPE-VR project has shown that there are several weak points that should be dealt with in subsequent versions. For one, the network communications scheme currently being used is somewhat unstable. It would be necessary to preserve the integrity of the network when one or more of the modules fail for whatever reason. As is, a module that is no longer receiving messages via its socket connection will self-destruct. A better way would be for the module to attempt to re-initialize the socket connection assuming that the process on the other end can also restart itself.

One concept that would be worth looking into is a transparent communications scheme such as that provided by VEOS (Virtual Environment Operating System), in processes communicate with each other independently of where they are located. With such a scheme, no distinction is made between shared memory and socket communications as far as the programs are concerned, these aspects being handled by the communications manager.

It is desirable to transfer the speech recognition interface to a platform that is common with something else in the system. Possible choices would be either a Windows version or a UNIX version so that the program could run in parallel with the ESOPE program or on one of the SGIs. With such a configuration the amount of hardware required would be reduced considerably, and the startup of the system would be simplified.

If a system such as ESOPE-VR should be commercialized, it would be necessary to have it run on a PC platform rather than on SGI workstations. The simple reason for this choice is that PCs are so widespread that there should be no acceptance problem, whereas not everybody can afford to purchase a graphical workstation. It is realistic to assume that in a few years' time, PCs will be able to support all the necessary technology to run a high quality virtual reality system. In the meantime, for further work on SGI workstations, the use of the Performer toolkit would be of interest [Rohlf 1994].

As future work, the use of a fully immersive head-mounted display, as well as touch-sensing and/or force-feedback devices could be considered. These technologies would provide enhancements to the realism of the VR experience. Alternative VR interaction tools such as glove input devices need to be evaluated to determine their suitability in the ESOPE-VR context. Finally, the automatic generation of 3D models should be expanded to allow for conversion of a single-line-diagram into a control-room.

5. Conclusion

The current thesis describes a first realization of providing electrical switching station operator trainees with a new training environment quite unlike any they have used or seen before. The motivation for the project came from the necessity of providing a safe learning environment, while preserving a high degree of realism. The successful integration of a 3D visual interface, speech recognition, aural immersion with verbal feedback, and a decision-support system in a real-time distributed client-server architecture demonstrates that such a system could be used as a powerful training tool.

Chapter one provided an overview of the current applications in virtual reality and the technology behind them. The next chapter contained an overview of the ESOPE-VR system including a look at the existing training software used at Hydro-Québec as well as descriptions of the various features of ESOPE-VR. Chapter three provided technical details for the different modules of the system, while Chapter four presented implementational details and evaluation of the relevant results obtained.

The ESOPE-VR prototype successfully demonstrated the feasibility of using virtual reality techniques to provide a training environment. The task now is to refine the hardware and software at hand in order to improve the system's performance and to search for new, more advanced, ways of interacting with the system.

6. References

The items in this list of references are representative of the current literature dealing with virtual reality, simulation and other related topics. It was suggested to include references to such resources as Usenet newsgroups or World Wide Web sites. However, because of their dynamic nature and their therefore inherent unreliability, it was chosen not to include specific references of this sort here.

[Adam 1994] ADAM, E.C. *Tactical Cockpits: The Coming Revolution.* "IEEE Aerospace and Electronic Systems Magazine" Vol. 9, No. 3, pp. 20-26, Mar 3, 1994.

Adam shows how cockpits have evolved over the last 30 years to become highly complex areas in which the pilot must work. Growing demands on the pilot lead researchers to look for ways to present more information to the pilot more efficiently.

[Amselem 1995] AMSELEM, D. *A Window on Shared Virtual Environments.* "Presence" Vol. 4, No. 2, pp. 130-145, 1995.

[Apple 1991] APPLE COMPUTER INC. *Audio Interchange File Format AIFF-C: A revision to include compressed audio data.* Draft version, 1991.

[Astheimer 1993,1] ASTHEIMER, P., FELGER, W. AND MÜLLER, S. *Virtual Design: A Generic VR System for Industrial Applications.* "Computers and Graphics" Vol. 17, No 6, pp. 671-677, 1993.

Astheimer describes a generalised, distributed VR system designed for architectural purposes (interior design, buildings, cities, landscapes), but also used for factory simulation, assembly/disassembly tasks, modelling, historic reconstruction, acoustic simulation.

[Astheimer 1993,2] ASTHEIMER, P. *What You See Is What You Hear - Acoustics Applied to Virtual Worlds.* "IEEE Symposium on Virtual Reality, San Jose, CA (Oct. 1993)".

[Aukstakanis 1992] AUKSTAKANIS, S., BLATNER, D. *Silicon Mirage: The Art and Science of Virtual Reality.* Peach Pit Press, 1992. ISBN 0-938151-82-7.

The VR-definition comes from here.

[Azuma 1994] AZUMA, R., BISHOP, G. *Improving Static and Dynamic Registration in an Optical See-Through HMD.* "Proceedings of SIGGRAPH 94 (Orlando, Florida, Jul 24-29, 1994).", in *Computer Graphics* Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 197-204.

[Bagiana 1993] BAGIANA, F. *Tomorrow's Space: Journey to the Virtual Worlds.* "Computers and Graphics" Vol. 17, No. 6, pp. 687-690, 1993.

Bagiana describes the VR activities in the Simulation Facilities Section at the European Space Agency (ESA). A VR prototype is described that was derived from existing simulation software and then extended to work on a distributed system using SGI workstations (Indigo + Onyx RE2). A training tool for astronauts in the Columbus space station is presented which uses various hand-object interactions with the DataGlove."High-definition graphic (sic), sound and force feedback are necessary to provide the correct cues and therefore avoid 'negative training.' "

[Berardinis 1993] BERARDINIS, L. *Look What's Talking.* "Machine Design" Aug. 27, 1993.

Berardinis talks about various consumer-electronics items aquiring the ability to speak or to understand speach. He mentions a "noisy revolution" that companies are expecting and gearing up for. A low-tech description of speech recognition follows.

[Bryson 1994] BRYSON, S., FEINER, S., ET AL. *Research Frontiers in Virtual Reality* "Proceedings of SIGGRAPH 94 (Orlando, Florida, Jul 24-29, 1994).", in *Computer Graphics* Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 473-474.

Bryson and four panelists give an overview of current limitations of several aspects of VR and present their ideas on developments in the (near) future.

[Bryson 1992, 1] BRYSON, S. *Survey of Virtual Environment Technologies and Techniques.* "SIGGRAPH Course Notes #9" pp. 1.1-1.35, 1992.

[Bryson 1993] BRYSON, S. *Virtual Reality in Scientific Visualization.* "Computers and Graphics" Vol. 17, No. 6, pp. 679-685, 1993.

Bryson discusses current use of VR at NASA Ames Research Center for visualization of large (1 GB and up), complex data sets. Impressive hardware is described (SGI Iris380 VGX with 256 MB memory, Convex supercomputer with 1 GB memory).

[Bryson 1992, 2] BRYSON, S. *The Virtual Wind Tunnel.* "IEEE Computers Graphics and Applications" Vol. 12, No. 7, pp. 25-34, 1992.

More of the same as in [Bryson 1993]; much overlap, same diagrams/figures.

[Buc 1994] BUC, N., PARIS, D., IZUMI, T. *Simulation of the Landing of a Re-Entry Vehicle Using Eurosim.* "ESA Bulletin" No. 79, pp. 27-33, August 1994.

[Carlsson 1993] CARLSSON, C. AND HAGSAND, O. *DIVE - a Platform for Multi-User Virtual Environments.* "Computers and Graphics" Vol. 17, No. 6, pp. 663-669, 1993.

A multi-user platform for virtual environments is presented which incorporates many users in a distributed system and allows them to interact. Techniques for interaction with objects are discussed including locking mechanisms to prevent inconsistencies in the data sets.

[Condom 1993] CONDOM, P., ROPELEWSKI, R., SUTTON, O. *Simulator Manufacturers dig in for Survival* "Interavia/Aerospace World" pp. 29-32, April 1993.

[Cooke 1994] COOKE, C., STANSFIELD, S. *Interactive Graphical Model Building using Telepresence and Virtual Reality.* "Proceedings - IEEE International Conference on Robotics and Automation." Publ by IEEE, IEEE Service Center, Piscataway, NJ, USA. pt 2 1994. pp 1436-1439.

A system for creating and verifying computer-generated graphical models of remote physical environments is presented. The system uses a robot to view the environment - underground storage tanks for radioactive waste - and an SGI Crimson RE along with a Boom Stereoscopic viewer, and a Dragon Systems voice recognizer.

[Coull 1992] COULL, T. *VR Applications: From Wall Street to Rehabilitation.* "Wescon Conference Record." Vol. 36, 1992. Publ. by Wescon, Los Angeles, CA, pp. 399-402.

Coull presents several applications that use WorldToolKit as a basis for VR.

[Cover 1994] COVER, S.A., EZQUERRA, N.F., *ET AL. Interactively Deformable Models for Surgery Simulation.* "IEEE Computer Graphics and Applications", Vol. 14, No. 11, pp. 68-75, 1994.

Cover presents a modelling technique to be used in surgical simulation. As well, he discusses the benefits and drawbacks of this kind of simulation.

[Cruz-Neira 1992] CRUZ-NEIRA, C., SANDIN, D.J., DEFANTI, T.A., KENYON, R.V., HART, J.C. *The Cave.* "Communications of the ACM." Vol. 35, No. 6, June 1992, pp. 65-72.

[Cruz-Neira 1993] CRUZ-NEIRA, C., SANDIN, D.J., DEFANTI, T.A. *Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE.* "Computer Graphics (Proceedings of SIGGRAPH '93)." ACM SIGGRAPH, August 1993, pp. 135-142.

[DeFanti 1993] DEFANTI, T.A., SANDIN, D.J., CRUZ-NEIRA, C. *A 'Room' with a 'View'.* "IEEE Spectrum" October 1993, pp. 30-33.

[Eberl 1994] EBERL, U. *Erstflug im Computer.* "Daimler-Benz High Tech Report 1/95", Daimler-Benz AG, Stuttgart, Germany 1994, pp. 40-47. (German)

Eberl talks about the use of computers to design new products rather than constructing actual scale models. Testing, for example crash-tests of automobiles, is accomplished by means of finite-element-analysis.

[Ellis 1994] ELLIS, S.R. *What are Virtual Environments?* "IEEE Computer Graphics and Applications", Vol. 14, No. 1, pp. 17-22, 1994.

[Esposito 1992] ESPOSITO, C. *Virtual Reality Research at Boeing.* "Wescon Conference Record." Vol. 36, 1992. Publ. by Wescon, Los Angeles, CA, pp. 397-398.

Esposito briefly presents various uses of VR such as Exterior or Interior design aircraft.

[Figueiredo 1993] FIGUEIREDO, M., BÖHM, K. AND TEIXEIRA, J. *Advanced Interaction Techniques in Virtual Environments.* "Computers and Graphics" Vol. 17, No. 6, pp. 655-661, 1993.

Figuerdo and his colleagues present parallels between "conventional" 2D interface concepts and new 3D ones. They then talk about various hand gestures for interacting with a virtual world.

[Foley 1990] FOLEY, J, VAN DAM, A. *ET AL. Computer Graphics, Principles and Practice.* Second Edition, Addison Wesley, 1990.

"The Bible of computer graphics."

[Foley 1993] FOLEY, M., BOSE, A., MITCHELL, W., FAUSTINI, A. *Object based graphical user interface for power systems.* "IEEE Transactions on Power Systems." Vol. 8, No. 1, pp. 97-104, Feb. 1993.

[Furness 1986] FURNESS, T. *The Super Cockpit and its Human Factors Challenges.* "Proceedings of the Human Factors Society", Nov. 1986.

[Gershon 1994] GERSHON, N.,*ET AL.* *Is Visualization REALLY Necessary? The Role of Visualization in Science, Engineering, and Medicine.* "Proceedings of SIGGRAPH 94 (Orlando, Florida, Jul 24-29, 1994).", in *Computer Graphics* Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 499-500.

Gershon and five panelists all say that visualization is crucial for many current applications in various fields.

[Göbel 1993] GÖBEL, M. *The Virtual Reality Demonstration Centre.* "Computers and Graphics" Vol. 17, No. 6, pp. 627-631, 1993.

[Goel 1995] GOEL, L., WONG, C.K., WEE, G.S. *An Educational Software Package for Reliability Evaluation of Interconnected Power Systems.* Paper 95WM 081-0 PWRS, IEEE Winter Power Meeting, New York, NY, Jan. 1995.

[Gosbee 1993] GOSBEE, J., CLAY, M. *Human Factors Problem Analysis of a Voice-Recognition Computer-Based Medical Record.* "IEEE Symposium on Computer-Based Medical Systems." Publ by IEEE, IEEE Service Center, Piscataway, NJ, 1993. pp 235-240.

A post-mortem analysis of the failure of a voice recognition system in an urban hospital's emergency department shows that too much time was necessary for training, not enough time was given for training, the pre- defined vocabulary and templates were frustrating, and a lack of computer skills and time prevented changing the situation. Ambient sounds impeding proper functioning are mentioned as one of the reasons for the system's failure. Several recommendations are made as to how the system could be improved.

[Griebenow 1993] GRIEBENOW, R., PERDEUS, M., SUDDUTH, A., HEUSER, S., FRAY, R. *Power plant simulators using Westinghouse distributed processing family hardware and software.* "Proc. of the Industrial Computing Conference." S. Triangle Park, NC, USA, pp. 15-24, 1993.

[Hale 1992] HALE, J. *Marshall Space Flight Center's Virtual Reality Application Program.* "Wescon Conference Record." Vol. 36, 1992. Publ. by Wescon, Los Angeles, CA, pp. 382-386.

Hale describes ongoing efforts at NASA's MSFC to use VR for training and mission-support purposes, and mentions the validation studies needed in order to be able to use VR in these ways. The use of VR for in-flight maintenance training and as a substitute for existing microgravity training methods is discussed.

[Hitchner 1992] HITCHNER, L. *The NASA Ames Virtual Planetary Exploration Testbed.* "Wescon Conference Record." Vol. 36, 1992. Publ. by Wescon, Los Angeles, CA, pp. 376-381.

Hitchner describes a VR application that is used as a "virtual exploration" system. Very large terrain data sets are used to create a virtual world in which the user can travel over certain sections of Mars or of other planets.

[Hodges 1992] HODGES, L.F. *Tutorial: Time-Multiplexed Stereoscopic Computer Graphics.* "IEEE Computer Graphics and Applications", Vol. 12, No. 3, pp. 20-30, 1992.

This paper presents an indepth description of stereoscopic vision with many technical details and some useful definitions.

[Höhne 1996] HÖHNE, K.H., PFLESSER, B. ET AL. *A 'Virtual Body' Model for Surgical Education and Rehearsal.* "Computer" Vol. 29, No. 1, pp. 25-31, 1996.

[Holzer 1995, 1] HOLZER, R. *JPATS Ground Trainer Entices Firms.* "Defense News" pp. 21, 24, March 6-12, 1995.

[Holzer 1995, 2] HOLZER, R. *Consolidation Buoys Trainer Industry.* "Defense News" pp. 22, 23, January 9-15, 1995.

[Holzer 1994, 1] HOLZER, R. *Atypical Missions Fuel Modeling Push.* "Defense News" pp. 48, 58, September 5-11, 1994.

[Holzer 1994, 2] HOLZER, R. *U.S. Debates Live Fire vs. Simulation.* "Defense News" pp. 22, 23, Aug 1-7, 1994.

[Holzer 1994, 3] HOLZER, R. *U.S. Army Pursues Unified Network.* "Defense News" pp. 22, 23, May 2-8, 1994.

As can be seen here, there is always something about computer-based simulation in Defense News. These few examples here were extracted during a visit to Oerlikon Aerospace's library.

[Hunter 1993] HUNTER, I. ET AL. *A Teleoperated Microsurgical Robot and Associated Virtual Environment for Eye Surgery.* "Presence" Vol. 2, No. 4, pp. 265-280, 1993.

Hunter and his colleagues describe a teleoperation system incorporating stereo vision, as well as force feedback. The technical implementation of visual feedback is discussed in detail. Signal propagation delays are mentioned as a problem affecting the use of this tool.

[Isdale 1994] ISDALE, J. *What is Virtual Reality? A Homebrew Introduction and Information Resource List.* At `ftp.u.washington.edu` as

`/public/virtual-worlds/papers/whatisvr.txt`.

A general introduction to VR's many aspects: hardware, software, its uses, etc.

[Janah 1994] JANAH, M. *NewsTalk at the New York Times.* "Forbes ASAP", A Supplement to Forbes Magazine, Dec. 5, 1994, pp. 86-88.

The use of voice recognition systems at the New York Times is described in this article.

[Jacobson 1993] JACOBSON, R. *After the "Virtual Reality" Gold Rush: The Virtual Worlds Paradigm.* "Computers and Graphics" Vol. 17, No. 6, pp. 671-677, 1993.

The role of chaos: "...it is common practice for speakers to allude to virtual worlds as 'the next step beyond multimedia' or 'a type of simulation.' Both claims are wrong.... (...) Most multimedia productions, at their core, are scripted to permit one or a number of outcomes conceived by their authors. The virtual world, being composed around the experience of the participant, in some sense is never complete and in every sense is subject to intervention by the participant. (...) The idea that virtual worlds are a type of simulation is more difficult to refute, not because it is right, but because, on cursory examination, it *appears* to be right. (...) 'Appearing to be right' is wrong. Nothing in the 'real world,' the material world, is 'right' in the sense of being perfect...."

[Jones 1995] JONES, C. (ED.) *Virtual Reality Moves Mountains with WorldToolKit* "Silicon Graphics World" Vol. 5, No. 6, p. 7, June 1995.

[Kalawsky 1993] KALAWSKY, R. *The Science of Virtual Reality and Virtual Environments* First Edition, Addison-Wesley, 1993.

[Kleinfeld 1995] KLEINFELD, N.R. *Stepping Through a Computer Screen, Disabled Veterans Savor Freedom.* "New York Times", Mar. 12, 1995, Sec. 2, , p. ?.

Kleinfeld describes the use of VR in a program that enables disabled people to experience many things they usually couldn't.

[LaPlante 1994] LAPLANTE, A. *The Lady and the Dragon* "Forbes ASAP", A Supplement to Forbes Magazine, Dec. 5, 1994, pp. 89-90.

The use of voice recognition in an office environment is presented as an successful alternative input device for people who suffer from carpal tunnel syndrome or similar ailments.

[Lipton 1991] LIPTON, L. *The CrystalEyes Handbook.* Stereographics Corporation, 1991.

[Mannes 1995] MANNES, G. *The Virtual Office.* "Popular Mechanics" March 1995, pp. 62-65, 115.

[Milgram 1995] MILGRAM, P. *Recent Work at UofT Ergonomics in Teleoperations and Control Lab* "CRIM Seminar." CRIM, May 1, 1995.

Prof. Milgram presented a one-hour presentation about current research at the ETC lab of University of Toronto. Information about this is available at `http://vered.rose.utoronto.ca`. He defines two "flavours" of mixed reality: "Augmented Virtuality" is the adding of real to virtual , while "Augmented Reality" is adding virtual to real. He mentions several experiments to study depth cues, for example depth through motion (rotation, or stereoscopic and rotation). At University of Tokyo, a VR setup is used to visualize a power network. A "silk cursor" is an interesting tool that is used for depth-cueing. One can tell where an object is by how much it is occluded."Augmented Virtuality" is used at the ATR Communications Systems Lab in Kyoto for "Virtual Space Teleconferencing." This is a virtual world with a stereo video background, therefore there is no need to model extensively. The ARGOS (Augmented Reality through Graphic Overlays on Stereovideo) system of UofT is Monitor-based, not HMD. It incorporates a "Virtual pointer" and a "virtual tapemeasure." Work with HMD with head-mounted cameras is being done. This has several advantages, especially the one of being easily calibrated, and that it can be extra-sensory (UV or IR, for example).

[Miner 1994] MINER, NADINE E. *Interactive virtual reality simulation system for robot control and operator training.* "Proceedings - IEEE International Conference on Robotics and Automation." Publ. by IEEE, IEEE Service Center, Piscataway, NJ, USA. pt 2 1994. pp. 1428-1435.

A distributed system using VR as a method to interact with robots engaged in industrial cleanup tasks is presented. Speaker-independent voice recognition is used as a control input with varying results. audio feedback is used to relate information regarding the current status of the robot to the user and to provide help about available commands at any point. Reference is made to the use of multimedia information to improve the training aspect of the system.

[Morris 1993] MORRIS, R.B., WHITMORE, M., ADAM, S. *How well does voice interaction work in space?* "IEEE Aerospace and Electronic Systems Magazine." Vol. 8, No. 8, pp. 26-30, Aug. 1993. 199419941994

Morris states that it is useful to train the speech recognition system including simulated background noise.

[Nemire 1994] NEMIRE, K., BURKE, A., JACOBY, R. Human Factors Engineering of a Virtual Laboratory for Students with Physical Disabilities. "Presence" Vol. 3, No. 3, pp. 216-226, 1994.

[Okapuu-v. Veh 1996] OKAPUU-VON VEH, A., MARCEAU, R.J., MALOWANY, A, DESBIENS, P., DAIGLE, A., GARANT, E., GAUTHIER, R., SHAIKH, A., RIZZI, J.C. *Design and Operation of a Virtual Reality Operator-Training System.* Paper 96WM 157-8 PWRS, IEEE Winter Power Meeting, Baltimore, MD, Jan. 1996.

Our most important paper about ESOPE-VR: a well-visited exhibit and an invitation to the summer power meeting in Denver followed the presentation.

[Osberg 1992] OSBERG, K.M. *Virtual Reality and Education: A Look At Both Sides of the Sword.* "HITL Report Abstract", 1992.

[Overbye 1995] OVERBYE, T.J., SAUER, P.W., MARZINZIK, C.M., GROSS, G. *A User-Friendly Simulation Program for Teaching Power System Operations.* Paper 95WM 048-9 PWRS, IEEE Winter Power Meeting, New York, NY, Jan. 1995.

[Peters 1995] PETERS, T.M. *Recent Developments in Medical Imaging.* "IEEE Seminar Series." McGill University, Department of Electrical Engineering, March 9, 1995.

[Piantanida 1992] PIANTANIDA, T. *Practical Applications of Virtual Reality.* "Wescon Conference Record." Vol. 36, 1992. Publ. by Wescon, Los Angeles, CA, pp. 388-396.

Piantanida describes many applications where VR could be used from Education over Architecture, Biomedical Sciences, many aspects of Medicine/Surgery, to Data Visualization and Simulation.

[Quéau 1993] QUÉAU, P. *Televirtuality: The Merging of Telecommunications and Virtual Reality.* "Computers and Graphics" Vol. 17, No. 6, pp. 691-693, 1993.

Quéau describes some initial steps at combining the telephone/communications network and VR to create a virtual world in which several participants can meet.

[Rajagopal 1994] RAJAGOPAL, S., RAFIAN-NAINI, M. *ET AL. Workstation Based Advanced Operator Training Simulator for Consolidated Edison.* "IEEE Transactions on Power Systems." Vol. 9, No. 4, pp. 1980-1986, Nov. 1994.

[Roe 1993] ROE, D., WILPON, J. *Whither Speech Recognition: The Next 25 Years.* "IEEE Communications Magazine" pp. 54-61, Nov. 1993.

Roe gives an in-depth, technical description of current and future trends in voice recognition.

[Rohlf 1994] ROHLF, J., HELMAN, J. *IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D.* "Proceedings of SIGGRAPH 94 (Orlando, Florida, Jul 24-29, 1994).", in *Computer Graphics* Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 381-394.

Performer is an interesting tool for rendering that should be looked into in the future.

[Roos 1995] ROOS, J.G. *Is It For Real? The Rush To Training Simulation* "Armed Forces Journal International", pp. 24-26, January 1995.

[Roy 1995] ROY, T.M., CRUZ-NEIRA, C., DEFANTI, T.A. *Cosmic Worm in the CAVE: Streering a High-Performance Computing Application from a Virtual Environment.* "Presence" Vol. 4, No. 2, pp. 121-129, 1995.

[Sagar 1994] SAGAR, M.A., BULLIVANT, D., MALLINSON, G.D., HUNTER, P., HUNTER, I.W. *A Virtual Environment and Model of the Eye for Surgical Simulation.* "Proceedings of SIGGRAPH 94 (Orlando, Florida, Jul 24-29, 1994).", in *Computer Graphics* Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 205- 212.

More of the same as in [Hunter 1993] with emphasis on modelling.

[SGI 1994] IRIS Media Libraries Programming Guide "Document Number 007-1799-030.", Silicon Graphics, Inc., 1994.

[Shaikh 1995] SHAIKH, A., GARANT, E., OKAPUU-VON VEH, A., MALOWANY, A., DAIGLE, A., DESBIENS, P., RIZZI, J.-C., MARCEAU, R.J., GAUTHIER, R. *Alternative Manipulation Strategies in a Virtual Reality Training System.* Canadian Conference on Electrical and Computer Engineering, Montreal, QC, September 1995.

[Shi 1993] SHI, J., ZHANG, A., ENCARNAÇAO, J., GÖBEL, M. *A Modified Radiosity Algorithm for Integrated Visual and Auditory Rendering.* "Computers and Graphics" Vol. 17, No. 6, pp. 633-642, 1993.

Shi and his colleagues present an algorithm that allows for the calculation sounds based on physical model simulation.

[Simpson 1985] SIMPSON, C.A. *System Design for Speech Recognition and Generation* "Human Factors", April, 1985, pp. 115-141.

[Stansfield 1993] STANSFIELD, S.A. *A Computer-Based Training System Combining Virtual Reality and Multimedia.* "Proc. of Intelligent Computer Aided Training and Virtual Environment TechConf" 1993.

[Sterbak 1994] STERBAK, R. *Platz nehmen im virtuellen Auto.* "Daimler-Benz High Tech Report 1/95", Daimler-Benz AG, Stuttgart, Germany 1994, pp. 48-55. (German)

[Sturman 1994] STURMAN, D.J., ZELTZER, D. *A Survey of Glove-Based Input.* "IEEE Computer Graphics and Applications", Vol. 14, No. 1, pp. 30-39, 1994.

[Sutherland 1965] SUTHERLAND, I. *The Ultimate Display.* Proc. IFIP Congress, 1965, pp. 506-508.

[Taverna 1993] TAVERNA, M. *Managing human error.* "Interavia/Aerospace World", pp. 37-41, April 1993.

[Tusa 1995] TUSA, F *Flying By The Seat Of Their Pants.* "Armed Forces Journal International", p. 31, January, 1995.

[Vadari 1995] VADARI, S.V., MONTSTREAM, M.J., ROSS, H.B.JR. *An Online Dispatcher Training Simulator Function for Real-Time Analysis and Training.* Paper 95WM 116-4 PWRS, IEEE Winter Power Meeting, New York, NY, Jan. 1995.

[Vanderheiden 1994] VANDERHEIDEN, G.C., MENDENHALL, J. Use of a Two-Class Model to Analyze Applications and Barriers to the Use of Virtual Reality by People with Disabilities. "Presence" Vol. 3, No. 3, pp. 193-200, 1994.

[Verbex 1990] Verbex 7000 Grammar Development Manual. Verbex Voice Systems, Edison, NJ, 1990.

[Ware 1993] WARE, C., ARTHUR, K., BOOTH, K.S. *Fish Tank Virtual Reality.* "Proceedings of INTERCHI, 1993, (Amsterdam) The Netherlands, 24-29 Apr. 1993." ACM, New York, 1993, pp. 37-42.

[Wenzel 1988] WENZEL, E.M., WIGHTMAN, F.L., FOSTER, S.H. *A Virtual Acoustic Display for Conveying Three-Dimensional Information* "Proceedings of the Human Factors Society", 32nd Annual Meeting, 1988.

[Wheatstone 1838] WHEATSTONE, C. *On some Remarkable, and Hitherto Unobserved, Phenomena of Binocular Vision (Part the First).* "Philosophical Transactions of the Royal Society of London", 1838, pp. 371-394.

[Wilson 1993] WILSON, J.R. *Virtual reality benefits may prove illusory.* "Interavia/Aerospace World" pp. 34-35, April 1993.

Wilson states that according to some people, VR may meet the same fate as artificial intelligence has. He states that VR lacks the ability to make everything "nearly perfect," and that VR should be replaced by the term "virtual environment displays" so that not too much emphasis is placed on "reality."

APPENDICES

Appendix 1: Startup/Shutdown Script

This is the *perl* script used to start up the Esope-VR system.

```
#!/usr/sbin/perl

#_______ESOPE RV_____________________________________________________
#
#    erv (= Esope-RV)
#
#    Starts up the whole system or kills everything.
#
#    Alexander Okapuu-von Veh, Nov 01, 1994.
#    My very first perl script!  :-)
#____________________________________________________________________

# This script uses the environment variable $ESOPE_HOME to determine
# where the file config.erv is located.  It reads this file to find
# out which executable is to be run/killed on which machine.  For each
# line in the file config.erv, this script either runs a remote shell
# (rsh), if the executable is to be run/killed on another machine, or
# just a shell on the local machine.

# Here is an example line from the file config.erv:
#
# "1   manic       ../bin     audio  5 yes -a -b5"

# This line specifies that the program "audio" is to be executed
# ("yes") with the parameters "-a" and "-b5" on "manic", and that this
# program is found in the directory ../bin, meaning $ESOPE_HOME/../bin
# A pause of "5" seconds will follow the start of audio.

# If this script is invoked as "start", one can use three optional
# parameters: -d, -e and -k.  "-d" means "debug" and implies debugging
# of this script.  The script generates and prints all the necessary
# commands, but does not execute them.  "-e" implies "use dummy
# esope", meaning that the esope program is not to be run in full
# mode, but in echo-only mode. Finally, "-k" implies "keyboard input",
# meaning that the verbex program shouldn't use the speech recognizer,
# but rather the keyboard input.

# If this script is invoked as "term", it runs a remote or a local
# shell (as the case may be) to determine the process ID of the program
# that is running.  This process ID, along with the appropriate
# commands, is used to kill the process.

while ($_ = $ARGV[0], /^-/){
  shift;
  last if /^--$/;
  /^-b/ && ($Logitech = "-b1") && print("Using logitech 6D mouse.\n");
  /^-c/ && ($Stereo = "-c")    && print("Running in stereo graphics
```

```
                                                  mode.\n");
  /^-d/ && ($Debug = 1)          && print("Debugging script only.  No program
                                                  execution.\n");
  /^-e/ && ($NoEsope = "-e")     && print("Esope running in echo-only
                                                  mode.\n");
  /^-f/ && ($Restart = "-f")     && print("Doing fast restart for
                                                  Esope-RV.\n");
  /^-h/ && ($Help = 1);
  /^-k/ && ($Keyboard = "-k")    && print("Keyboard used for Verbex.\n");
  /^-p/ && ($Glove = "-p2")      && print("Using Powerglove.\n");
}

print "\n\n";

if ($Help == 1) {
  print("Options are as follows\n\n");
  print("-b\t Use logitech 6D mouse.\n");
  print("-c\t Use stereo graphics for WorldToolKit.\n");
  print("-d\t Debug this script - don't execute any programs.\n");
  print("-e\t Use Esope in echo-only mode.\n");
  print("-f\t Do a fast restart of Esope-RV.\n");
  print("-h\t Print this help message.\n");
  print("-k\t Use keyboard for Verbex.\n");
  print("-p\t Use Powerglove.\n");
  print "\n\n";
  exit;
}

# affiche le moment du demarrage
$date = `date`;
print "\nDate: $date\n";

# get the host name and the environment variable ESOPE_HOME
chop ($host = `hostname -s`);          # remove trailing '\n'
$esope_home = $ENV{'ESOPE_HOME'};

# Find out from the command line what the user want: start or term.
$WhatToDo = $0;

# check parameters
if ((!($WhatToDo =~ term)) && (!($WhatToDo =~ start))) {
  die "Use 'start' or 'term'....\n";
}

# Customize the config.erv file for the host on which the program is
# launched: This is done creating a temporary softlink.
$cmd = sprintf("ln -sf $esope_home/config.erv.$host $esope_home/config.erv\n");
system $cmd;

print "$esope_home/config.erv \n\nlinked to customized config:\n
$esope_home/config.erv.$host\n\n";

# try to open the configuration file; exit if not possible.
open(CMDFILE, "$esope_home/config.erv") || die "Can't open config.erv\n";
$buf = <CMDFILE>;

print "Command file opened: \n$buf\n";
```

```
# Process configuration file line by line until EOF
while ($CMD = <CMDFILE>) {
  ($ID) = split(' ', $CMD);

  # process file until token "bloc" found
  if ($ID =~ bloc) {
    # now process file until token "fin" found
    # parse line read from configuration file
    while (($CMD = <CMDFILE>) != fin) {
      ($ID, $Machine, $Path, $Name, $Pause, $Exec, $Prm1, $Prm2) =
       split(' ', $CMD);

##############################################################
# $host =~ $Machine i.e. process is to be started on this host
##############################################################
      if ($host =~ $Machine)  {  # we're on the local host
        if ($WhatToDo =~ start) { # command = start programs
          if ($Name =~ esope)
          {
            if ($NoEsope =~ e) {
              $cmd = sprintf("%s/%s/%s %s %s %s %s %s &\n",
                $esope_home, $Path, $Name, $Prm1, $Prm2, $NoEsope, $Glove,
                $Logitech);
            }
            else {
              $cmd = sprintf("%s/%s/%s %s %s %s %s %s &\n",
              $esope_home, $Path, $Name, $Prm1, $Prm2, $Restart, $Glove,
              $Logitech);
            }
          } # if ($Name =~ esope)
          elsif ($Name =~ verbex) {
            $cmd =
              sprintf("winterm -iconic -title Verbex -n Verbex -hold -e
                %s/%s %s %s %s \n", $esope_home, $Path, $Name, $Prm1,
                $Prm2, $Keyboard);
          }
          else {
            $cmd =
              sprintf("%s/%s/%s %s %s %s %s %s %s &\n", $esope_home, $Path,
                $Name, $Prm1, $Prm2, $NoEsope, $Stereo, $Logitech, $Glove);
          }

          if ($Exec =~ yes) {          # if the program is set to exec
            print  $cmd;               # print command to reassure user
            if ($Debug != 1) {
              system $cmd;             # execute the command
              sleep $Pause;            # pause apres le demarrage
            }
          }             # if ($Exec =~ yes)
          else {           # if no execution is necessary
            print "*** SKIP: $Name ***\n";   # print command to reassure user
          }             # else
        } # if ($WhatToDo =~ start)
        elsif ($WhatToDo =~ term) {           # command = kill programs
        # program was executed and can be killed
          if ( ($Exec =~ yes) || ($Exec =~ term) ) {

            # define the command
```

```
          $cmd = sprintf("ps -d | grep %s | \n", $Name);
          print $cmd;
          open(PID, $cmd);                     # read from pipe

          # kill all processes matching the argument
          while ($PID = <PID>) {
            ($pid) = split(' ',$PID);          # get numeric PID

            # define the command
            $cmd = sprintf("kill -USR1 %d \n", $pid);
            print $cmd;      # print command to reassure user
            system $cmd;     # execute the command
          } #  while ($PID = <PID>)
        } # if ($Exec =~ yes)
      } # elsif ($WhatToDo =~ term)
    } # if ($host =~ $Machine)

################################################################
# $host!=$Machine i.e. process is to be started on distant host
################################################################
    else {
      if ($WhatToDo =~ start) {          # command = start programs
        if ($Exec =~ yes) {              # if the program is set to exec
          # define the command
          if ($Machine =~ lagrande) {
            $cmd =
              sprintf("rsh lagrande -n 'setenv DISPLAY :0.0; %s/%s/%s %s %s
                %s %s %s %s &' &\n", $esope_home, $Path, $Name, $Prm1,
                $Prm2, $NoEsope, $Stereo, $Logitech, $Glove);
          }
          else {
            if ($Name =~ verbex) {
              $cmd = sprintf("winterm -iconic -n Verbex -t Verbex -hold -e
                 rsh %s %s/%s/%s %s %s %s \n", $Machine, $esope_home,
                 $Path, $Name, $Prm1, $Prm2, $Keyboard);
            }
            elsif ($Name =~ esope) {
              if ($NoEsope =~ e) {
                $cmd = sprintf("rsh %s -n %s/%s/%s %s %s %s %s &\n",
                $Machine, $esope_home, $Path, $Name, $Prm1, $Prm2,
                $NoEsope, $Restart);
              }
              else {
                $cmd = sprintf("rsh %s -n %s/%s/%s %s %s %s &\n", $Machine,
                   $esope_home, $Path, $Name, $Prm1, $Prm2, $Restart);
              }
            }
            elsif ($Name =~ showmov) {
              $cmd =
              sprintf("rsh %s -n 'setenv DISPLAY :0.0; %s/%s/%s %s %s %s &'
                &\n", $Machine, $esope_home, $Path, $Name, $Prm1, $Prm2,
                $Restart);
            }
            else {
              $cmd =
                 sprintf("rsh %s -n %s/%s/%s %s %s &\n", $Machine,
                 $Esope_home, $Path, $Name, $Prm1, $Prm2);
            }
```

```
      }
      print $cmd;             # print command to reassure user
      if ($Debug != 1) {
        system $cmd;           # execute the command
        sleep $Pause;           # pause apres le demarrage
      }
    }
    else {            # if no execution is necessary
      print "*** SKIP: $Name ***\n";    # print command to reassure user
    }
  }
  elsif ($WhatToDo =~ term) {         # command = kill programs

    print "Seek and destroy $Name \n";
    if ($Machine =~ eddie) {         # Linux requires "ps -ax"
      $cmd = sprintf("rsh %s -n ps -ax | grep %s | \n", $Machine,
        $Name);
    }
    else {                           # IRIX is "ps -d"
      $cmd = sprintf("rsh %s -n ps -d | grep %s | \n", $Machine,
         $Name);
    } # else
    if ( ($Exec =~ yes) || ($Exec =~ term) ) {
      print $cmd;
      open(PID, $cmd);               # read from pipe
      while ($PID = <PID>) {
        ($pid) = split(' ',$PID);  # get numeric PID

        # define the command
        $cmd = sprintf("rsh %s -n kill -USR1 %d \n", $Machine, $pid);
        print $cmd;                  # print command to reassure user
        system $cmd;                 # execute the command
      } # while
    } # if ($Exec =~ yes)


    # This is a bit silly, but we have no way of knowing whether
    # the user started verbex with the -k option or not, so seek
    # and destroy any xterms... :-/
    if ($Name =~ verbex) {
      $cmd = sprintf("ps -d | grep xterm | \n");
      print $cmd;
      open(PID, $cmd);                  # read from pipe

      while ($PID = <PID>) {
        ($pid) = split(' ',$PID);      # get numeric PID

        # define the command
        $cmd = sprintf("kill -9 %d \n", $pid);
        print $cmd;                     # print command to reassure user
        system $cmd;                    # execute the command
      } # while
    } # if ($Name =~ verbex)

   } # elseif ($WhatToDo =~ term)
  } # else
 } # while (($CMD = <CMDFILE>) != fin)
last; # done after "fin"
```

```
  } # if ($CMD =~ bloc)
} # while ($CMD = <CMDFILE>)

close(CMDFILE);
```

Appendix 2: System Configuration File

This is the configuration file for the Esope-VR system.

```
#_______CONFIG.ERV.LAGRANDE et LAFORGE________________________________
#
#       config.erv
#
#       Paramètres de configuration et lancement du système.
#_____________________________________________________________________

# CONFIGURATION DU SYSTÈME:
# ======================
#
# BLOC INFORMATION EMPLACEMENT (SITE):
# ---------------------------------
#
# Le bloc est delimité par les bornes "bloc site" et "fin".  Ces
# dernières sont  utilisées par le module de communication lors de
# l'établissement des liens afin d'extraire l'information concernant
# la localisation des processus.  L'ordre dans lequel les processus
# apparaissent est celui selon lequel ils doivent etre amorcés.
#
# Win-Esope doit etre demarré préalablement sur la plateforme
# Windows, ensuite viennent les serveurs "purs", suivis des
# processus qui peuvent à la fois etre serveur et client dependemment
# du contexte et finalement les clients "purs" sont amorcés.
#
# Le fichier contient un tableau décrivant chacun des processus:
#     - Un identificateur (ID) represente de façon unique chaque
#       processus.
#     - MACHINE est le nom de la machine sur laquelle tournera le
#       programme
#     - PATH represente l'emplacement du programme relativement au
#       repertoire specifié par la variable d'environnement
#       $ESOPE_HOME.
#     - NOM PROCESSUS est le nom de l'executable.
#     - PAUSE specifie un délai a respecter suite au démarrage du programme.
#     - EXEC specifie si l'on doit démarrer ou non le processus à
#       partir de la procedure executée par le script PERL.
#     - PARAMS permet d'ajouter un nombre arbitraire de parametre à
#       passer au programme lors de son lancement.
#
```

```
# ID    MACHINE         PATH      NOM PROCESSUS  PAUSE    EXEC?  PARAMS
# ==    =======         ====      ============   =====    =====  ======

bloc site
1       turtle           .          win-esope     0        no
2       lagrande         .          bgsnd         5        yes
3       laforge          .          showmov       5        yes
4       laforge          .          audiovid      5        yes
5       laforge          .          esope         10       yes
6       lagrande         .          esope-rv      10       yes
7       eddie            .          verbex        5        yes
8       lagrande         .          worldto       5        term
fin

#
# BLOC INFORMATION RELATIONS ENTRE LES PROCESSUS (ROLE):
# -----------------------------------------------
# Definition des relations entre les differents processus.
# Chaque relation n'apparait qu'une fois e.g.: 2 est un client de 3.
# dans le sens X est un client de Y et non X est un serveur de Y.
# Ici aussi, le bloc d'information est defini par les délimiteurs
# "bloc role" et "fin".
#

bloc role
7 client 6 serveur
6 client 5 serveur
6 client 4 serveur
7 client 5 serveur
5 client 4 serveur
7 client 4 serveur
5 client 1 serveur
4 client 3 serveur
4 client 2 serveur
fin

# Example: 4 client 3 serveur means that audiovid has showmov as a
# server.
```

Appendix 3: Single-Line Diagram

The following figures show the single-line diagram used for ESOPE-VR with its corresponding three-dimensional representation. The 120kV ring-bus architecture shown on the left hand side of the single-line diagram is represented by the smaller structures in the right foreground of the image, while the transformers T1 and T2 in the centre, and the 735kV part of the substation on the right of the schematic appear in the background on the left side of the image.

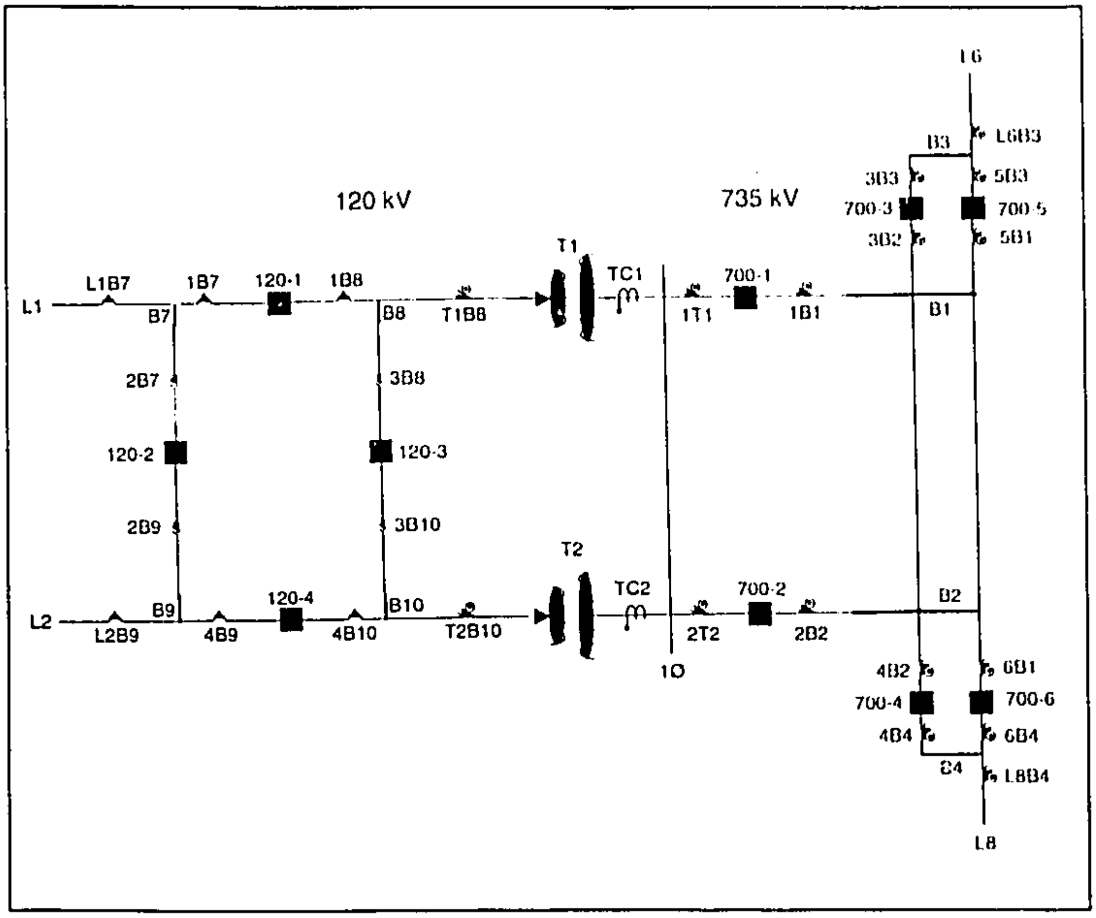

Figure A3.1: Single Line Diagram for Esope-VR

Unfortunately, the print quality does not do justice to the full-colour representation on the computer screen. However, the level of detail is sufficiently high to be able to recognize the appropriate structures. In Figure A3.3 a single line diagram is shown which is currently used at Hydro-Québec for operator training. This gives the reader an idea of what should be simulated in a more advanced VR system.

Figure A3.2: 3D Image Representation of Single Line Diagram

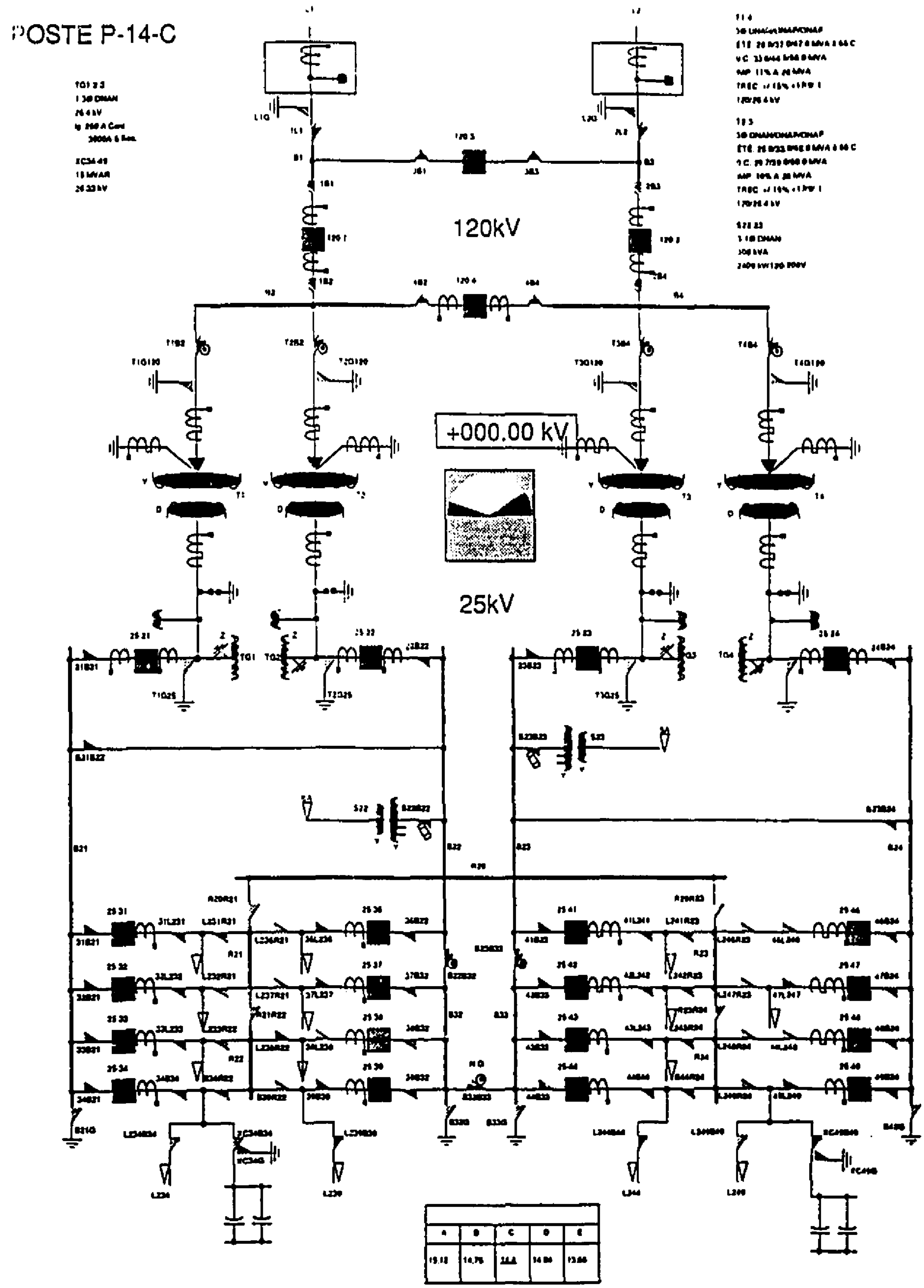

Figure A3.3: Hydro-Québec Single-Line Diagram

Appendix 4: Verbex User Questionnaire and Answers

The following are the questions and answers from the Verbex questionnaire that was completed by the seven people who used the speech recognizer. Please note that some of the answers supplied here were translated from French.

1.	**How easy was it to train the Verbex?**
	It was easy.
	Very easy, even for me.
	Pretty easy.
	The Verbex was very easy to train. The only problem was the time it took to train it. The feature that one could partially train and continue where one left off at a later date proved useful.
	It's very easy and takes about 10 minutes to learn.
2.	**Was the printed training procedure useful?**
	Yes, and it could even be more descriptive.
	Yes, and even necessary because Verbex' user interface is not very explicit.
	Yes. (three responses)
3.	**Were my hints and advice useful?**
	Very.
	Yes, notably the indication on how to distinguish words that are similar in pronunciation (1T1 and 1B1, for example).
	Definitely.
	I did not receive any hints or advice therefore, no comment.
	Yes, they were. Thanks very much for your help.
4.	**How could the training procedure have been improved?**
	One of the fears I had was to lose what had been trained up to now for whatever reason (hardware, accident, spilled coffee). Be more precise about the way to make backups. You could give an explanation about how Verbex actually works (the role of the various files, why it is called "uploading" when it is "downloading." Also explain how to change the voice in Verbex. (?)
	Improve Verbex' user-interface.
	It would reject the bad one even when the user accidentally press the space bar.
	Optimize the training script such that the training time is reduced.
	Maybe add something about single-training and testing?

5.	**Was using the Verbex "natural" for you? If no, did you get used to it after a while? If yes, how long did it take? If no, why not?**
	It's more or less natural. You always have the impression of talking to a machine, which was OK because the actual phrases were not always meant for Verbex. This made Verbex a bit intimidating. After a few minutes of trying, I found out how to talk to Verbex, and my relation to Verbex improved considerably....
	Its use is natural, but it's not easy to use a natural or appropriate voice. You do get used to it though in about 10 minutes.
	No, it was not natural. I did get used to it after a while. At the beginning I tried "too" hard to speak to it in a normal manner. I had a big problem with nasal words like "move".
	The only unnatural part was the finicky microphone positioning required. Yes, but I never was fully used to it. It wasn't an initial "getting used to" overhead but a constant learning process.
	Yes, it's reasonable.
6.	**Approximately how much time did you spend training Verbex? Was this time adequate? Too long?**
	About three hours. I think it's too long. The vocabulary should be cut in half.
	Between one and one half hours and two hours.
	45 minutes. Quite reasonable.
	About 1.5 hours. Too long.
	For a simple grammar file with commands move. direction. number, pick up box. number, place box. number, view. direction. number and stop, it took about 20 minutes to train the Verbex for the first time and several single-trainings followed afterwards.
7.	**During training, was the recognition rate acceptable to you? If no, what did you do about it? Did you want to re-train certain words/phrases? How many? Do you recall having misrecognition errors with certain words? Which ones?**
	Yes, the rate was good (> 80%). My hang-up was "arreter" - 10 to 12 tries each time; one in two times I reached an acceptable rate. For the times where the volume was not acceptable, I started over two to three times on average.
	The rate was acceptable, probably about 80%. I re-trained few words; less than 5. Misrecognition of "T" with "B", and numbers such as "treize" and "seize."
	It could be better. I didn't do anything about it, though. I had problems with one word. It had trouble recognize variable digits in a long command. ie. "pan 4 1 9 right" became "pan 1 4 9 right"....
	Yes, very acceptable. About 2 or 3 words were retrained. I think "sept" had problems.
	The overall recognition rate is quite satisfactory but the Verbex does have problems in recognizing "stop" regardless of the times I trained it.

8.	**When using the Verbex in the ESOPE-VR[9] context, was the recognition rate acceptable to you? Please explain.**
	The rate was acceptable when the operating conditions were good. I didn't have the chance to use it during the demo, but I did try it one evening and the rate was OK.
	N/A - didn't use.
	Yes. It recognize the command most the time. When it does not recognize just repeat the command.
	Very acceptable. The only issue was its intolerance to high levels of ambient noise.
	The recognition rate is acceptable but it's very annoying when the "stop" is not working because it's an important word.
9.	**Was this your first experience with a speech-recognition system? If yes, what is your overall impression of this system as a tool for interacting with a computer? [Please be descriptive; use many adjectives! :-)]**
	It was my first experience with this technology. Even if it has been around for several years, it's little-used by companies and individuals. It's almost certain that this way of controlling computers will be the norm one day. My experience with Verbex was like a short-lived jump to the future and has let me become a bit familiar with the interface of the future.
	First experience. Natural, fast, ease of getting used to it, well suited to the VR context.
	Yes. Cool! except you have to go though the training process.
	No, I've used the Verbex itself before.
	Yes it is. It's easy to use when the grammar file is not very complicated and the way that Verbex outputs recognized words to a computer is convenient for communication. However, if there is a big and complicated grammar file, it's difficult for the user to memorize all the grammar rules. In this respect, it is not as attractive as some other interfaces, for example a menu-driven user interface where all the commands are available as pull-down menus, icons etc.
	It works. That's the main feature. My impression is based purely on the comparison of this system as opposed to another system for interacting with a computer. Compared to a keyboard or mouse driven user interface, the voice interface wins hands down. Its ease of use and learning curve are far superior. I guess the ideal interaction device would be a brain scanner :). The Verbex is the high end speech recognition system of a few years ago. My experience has been limited to lower end systems which do not even compare.

[9] In the case of people who were not part of the ESOPE-VR project, this was changed to "...in your particular context..."

A.	**What are the drawbacks of the Verbex system?**
	Training time was too long. The procedure for changing the user is too long. The unit takes a lot of space. The mike is of low quality (should be noise-canceling). Possibilities for translation errors (you say "seize" and it recognizes "treize").
	A bit long to train it. Bad user-interface, especially if one has to add vocabulary items, and doesn't remember the commands, and the instruction sheet is gone.... Obviously, there's always the problem of background noise that causes the thing to malfunction.
	User dependent.
	Individual users need to be trained.
	One of the drawbacks is mentioned above. Another drawback is that it is user dependent, which is not very efficient. Moreover, it has problems in recognizing some specific words.
B.	**What are the advantages of the Verbex system?**
	More natural than using the mouse. Doesn't cause damage to your wrist as a mouse or keyboard does. The training lets you get familiar with the possible vocabulary items (one advantage of a speaker-dependent system).
	Lets you free your eyes for looking at the virtual environment, and therefore gives you a more interesting immersion experience.
	Each user has his or her voice file, so that people with heavy accent can be recognized by Verbex.
	No comment.
	Eyes free and hands free (for lazy people :)). The response of the Verbex is very fast when it recognizes a word.
C.	**Did you find the Verbex system useful in the ESOPE-VR context?**
	Very. There's still the problem of erroneous translation (see above). There should be a way to put more intelligence into the system for the problem cases in order to avoid such errors.
	Yes, very good.
	Kind of.
	Yes.
	Yes, but I didn't find it irreplaceable.

D.	**How could its use be improved in this context?**
	Make it less sensitive to ambient noise. Make it easier to change the person who is recognized. Here's a crazy suggestion: you enter the manoeuvre mode, and a floating menu appears in front of you on the screen, which indicates the possible choices. You choose one of them by saying its name, and the menu is replaced by another one. This way, the commands would be better-known, and used more. The people who are more "visual" would be able to use the system better.
	The words could be a bit more flexible, and/or accept alternatives ("*ouvre*" = "*ouvrir*" = "*ouvre le*" = "*ouvrir le*"). In the same sense, could we have a system that recognizes keywords rather than exact formulations? This would free the user of the obligation to memorize more-or-less long strings of words.
	Don't know.
	Increased vocabulary, increased functionality. I think another improvement would be to have a certain amount of redundancy in the words chosen to perform a function. For example being able to say "teleporter salle de commande" or "teleporter a la salle de commande," etc.
	It is important to optimize the grammar file. May be we could add some more feedback and provide help information.
E.	**Are there any specific problems with Verbex?**
	The interface is a bit passé (DOS, what's that?!) Verbex takes up a lot of space physically.
	We always need a keyboard interface as a backup!
	I have said earlier, it doesn't like nasal words.
	Yes, the biggest being its incapability of understanding one's voice within a room of high levels of noise.
	We haven't tested Verbex's noise tolerance, it might be a problem.
F.	**Comments? Suggestions?**
	Make the training document more visually attractive with screen shots.
	This is definitely worth a Master's, but have you thought about a PhD.?
	Can't think of any right now.
	Test other more recent voice recognition platforms and compare.
	None.