Training Spatial Knowledge Acquisition Using Virtual Environments

by

Glenn Koh

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 27, 1997

Author .....
Department of Electrical Engineering and Computer Science
May 27, 1997

Certified by .....
Nathaniel I. Durlach
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Training Spatial Knowledge Acquisition Using Virtual Environments

by

Glenn Koh

Submitted to the Department of Electrical Engineering and Computer Science on May 27, 1997, in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

Virtual environments have developed to such a point of sophistication that it is possible to attain an unprecedented sense of immersion in a computer simulation. The overall objective of this work is to apply state-of-the-art virtual reality simulation technologies to investigate issues in spatial knowledge acquisition and training. Subjects were trained to recognize key features of a venue using one of: an immersive virtual environment, a non-immersive virtual environment, an exocentric virtual model of the venue, and a walkthrough of the actual venue. Subjects were subsequently brought to the venue and tested for knowledge acquisition from the various training methods in order to quantify the efficacy of the various training systems.

.

Thesis Supervisor: Nathaniel I. Durlach
Title: Senior Research Scientist

1.0 Introduction

Virtual environments have developed to such a point of sophistication that it is possible to attain an unprecedented sense of immersion in a computer simulation. With the availability of such systems comes a new way of information representation. The overall objective of this work is to apply state-of-the-art virtual reality simulation technologies to investigate issues in spatial knowledge acquisition and training. Virtual environments provide a number of advantages over traditional training methods and it is expected that training with such a system will translate into measurable real-world gains in performance.

The primary objective of this study is to lay the foundation for future investigation in subsequent years. In particular, the first study will show that virtual environments can be used to effectively acquire spatial knowledge of a specific real space. Future work will involve the investigation of environmental features and perceptual stimuli and their role in navigation and wayfinding so that we may understand how to degrade the fidelity of a virtual environment without substantially reducing the amount of knowledge transfer. We will then extend these concepts to training general navigation skills, including map usage (perspective transformation) and landmarking abilities.

We are currently also investigating the transition of this research into the dismounted infantry and/or special operations communities. The requirements of these groups call for deep, highly accurate spatial knowledge of spaces which they have never physically seen before. Research into spatial cognition and virtual environment training methods will only augment and provide further insight into the challenges faced by these groups and provide ways of attacking their specific problems. Because of the interdisciplinary nature of this study, which is based on cognitive science, perceptual psychology, virtual environment design, and real-world training methods,

advancements or insights into any one area will indubitably lead to breakthroughs in the others.

2.0 Navigation Overview

The aim of this project is to establish the efficacy of virtual environments as a training medium for spatial knowledge acquisition. By definition, navigational awareness of an environment is characterized by complete navigational knowledge of the environment. Navigational knowledge can be divided into two different areas, each of which is applicable to different tasks.

This study is primarily interested in determine the knowledge transfer of the spatial characteristics of an environment to a previously ignorant navigator. A successful learning task will then be one which allows the navigator to accurately navigate the environment under constraints placed upon her during the testing phase.

2.1 Route Knowledge

The first type of navigational knowledge is route knowledge. Route knowledge is characterized by the navigator's ability to move from position A to position B along a specific learned route. The navigator may not have any knowledge of the relative spatial locations of A or B and is unaware of alternate routes linking A and B. In general, route knowledge is egocentric.

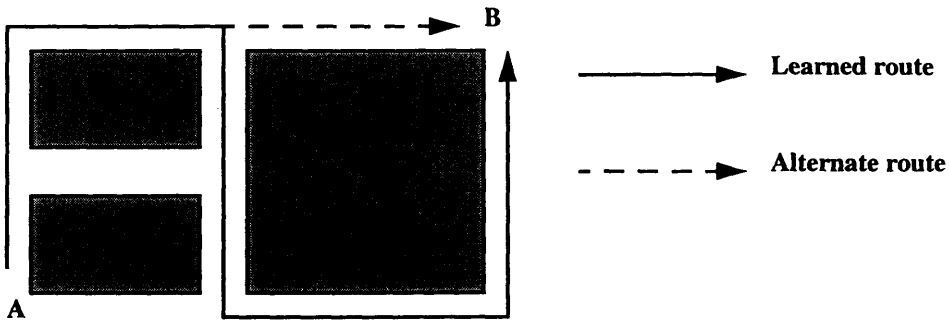


FIGURE 1. Navigator with route knowledge has the ability to move from A to B, but not to take the more efficient alternate route because of lack of configurational knowledge.

While the navigator may be able to move between two points, he may not be able to easily traverse the reverse path or take an exocentric view of the situation to draw or otherwise describe the path to another person. A navigator with strong route knowledge has knowledge of a particular learned path through the environment.

Route knowledge can be trained and tested relatively easily. The navigator can be trained through exocentric or egocentric descriptors (or a combination of the two). Examples of real world exocentric training methods would include drawing a map of a defined path, verbally describing a path to a navigator ("go east two blocks and turn right at the traffic light"), or a similar written list of directions. Exocentric training methods have in a common a lack of immersiveness of the navigator into the environment before her task begins.

Examples of real world egocentric training methods include having the subject traverse a path while being instructed in it and transporting the subject along the path.

These training methods have in common the navigators' ability to see the path from a first-person point of view during the training sequence.

Testing of route knowledge may be accomplished by having the navigator reproduce the learned path independent of the training cues. The knowledge transfer of the training method of the navigator can then be determined as a function of the time required to traverse the path and the accuracy of reproduction of the route.

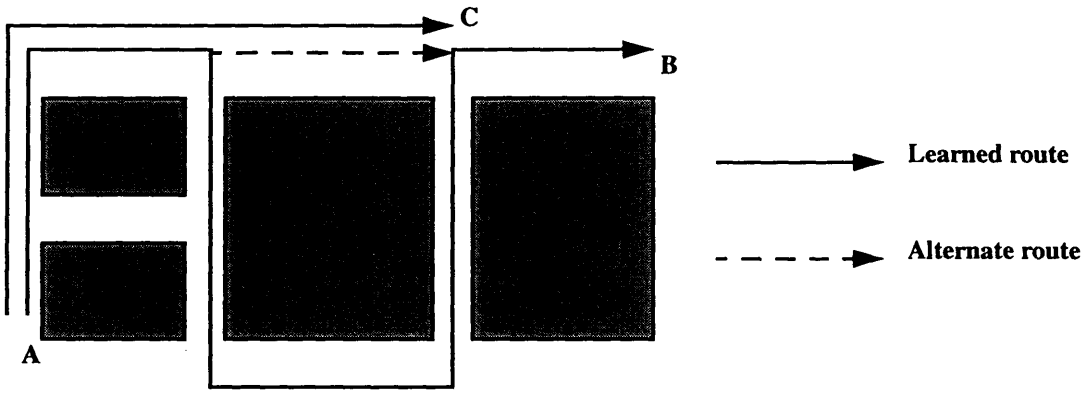


FIGURE 2. Navigator with knowledge of route A-C and route A-B can find optimal route A-C-B even without configurational knowledge of the environment.

A navigator with perfect route knowledge has the ability to traverse accurately all routes between two points within the studied environment. By induction, the navigator can also accurately traverse all routes between points which are formed by the connected graph of nodes indicating traversed points in his learned routes.

2.2 Configurational Knowledge

The second type of navigational knowledge is called configurational (or spatial) knowledge. Configurational knowledge stems from an awareness of the space as a whole. Configurational knowledge is characterized by the navigator's knowledge of the relationships between different locations and her ability to view the location from an exocentric viewpoint. The navigator is able to determine routes between locations in the space which she may not have previously traversed, and is able to describe the physical characteristics of the space.

Configurational knowledge can be trained without any external (experimenter) influences. Subjects may be given a map of the venue or allowed to traverse the environment immersively.

A navigator with a strong configurational knowledge of an environment has:

1. Knowledge of the structure of the environment.
2. The ability to draw a map of the environment.
3. The ability to find the optimum route between two points within the environment.
4. The ability to estimate distance and direction to a particular landmark or location within the environment.

Assessing configurational knowledge acquisition can be more difficult than measuring route memorization accuracy, however. One can measure route knowledge simply by requesting that the subject reproduce the path of travel. Testing configurational knowledge, however, is not as straightforward. Ideally, the tests used to measure configurational knowledge should asses the subjects' knowledge of a space - not the subjects ability to convey that information. Subjects may have different drawing, modeling, or verbal descriptive abilities. Configurational knowledge should be tested by methods which control for these factors.

Two methods which are relatively independent of a subjects spatial descriptive abilities are: a pointing task in which the subject is brought to a position A and asked to estimate direction and distance to a number of other positions B in the venue, and a wayfaring task in which the subject is asked to proceed from location A to location B via some route.

2.2.1 Pointing Task

The pointing task is a derivation of a technique proposed by Siegel (1981) to measure configurational knowledge. Siegel's projective convergence technique requires subjects to estimate the bearing and distance to a number of target landmarks from several different pointing locations. These bearing and distance estimates can be used to develop a measure of the subjects' spatial understanding of the location of the trageted landmarks.

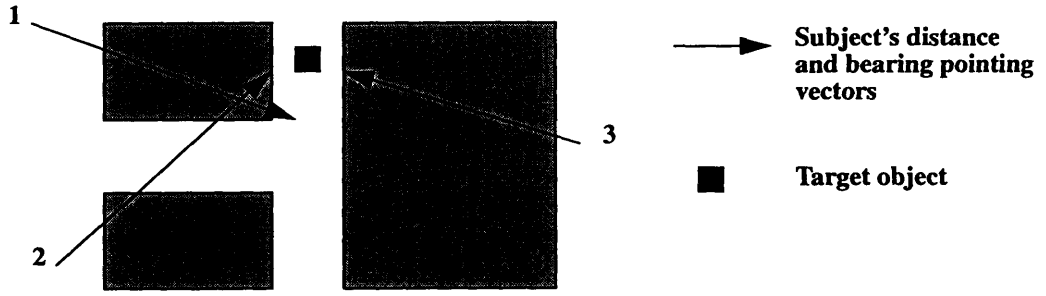


FIGURE 3. Subject is asked to point to target object from three different locations within the venue. The subject's pointing vectors from the three locations may be used to determine his knowledge of location of the target object.

The pointing task measures the subject's ability to reproduce a 3D replica of the space by having her point and estimate the distance to landmark B from position A (position A may not or may not be the subject's current position).

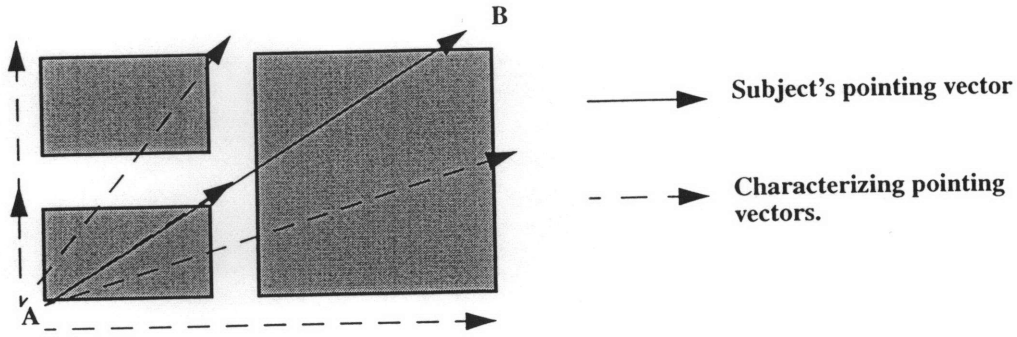


FIGURE 4. Subject pointing from current position A to B. Note that the subject can characterize the space completely if able to point to any intermediate point.

Theoretically, a subject who is able to identify the locations of all points relative to his own or relative to each other has a perfect spatial knowledge of the venue since that subject can completely determine a spatial map of the venue. The accuracy (azimuth and distance) of the subject's performance can be used to determine her knowledge of the space.

2.2.2 Wayfaring Task

The wayfaring task involves the subject moving from two points in the venue which via a route with characteristics explicated by the interviewer at the time of testing. The subject may be asked to find the shortest path between A and B or a path via a number of waypoints, possibly being on another level of the venue. The goal of this test is to assess the subject's spatial and navigational knowledge from her egocentric perspective. Because the paths are not known to the subject during the practice phase of the experiment, the subject is force to piece together the route observations of the

venue derived during the practice phase - it is not possible for the subject to memorize specific routes.

The subjects performance will be assessed based upon a number of variables including:

1. Time taken to complete the task.
2. Number of wrong turns.
3. Distance travelled by the subject.
4. Ability to complete the task.

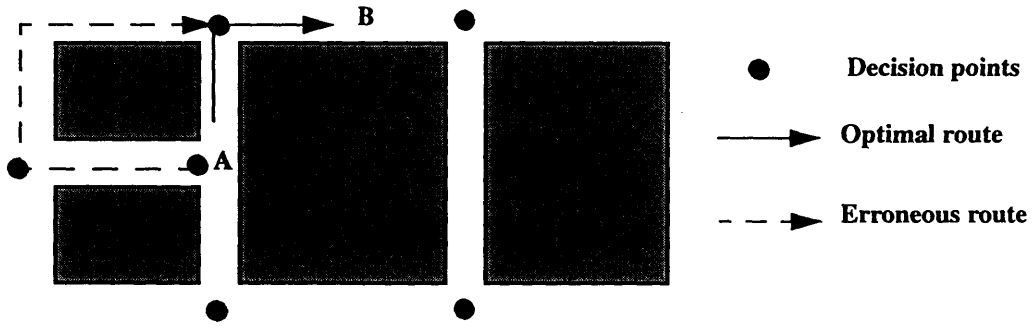


FIGURE 5. An example venue in which there are 6 decision points.

The evaluation of the subject is based upon her ability to correctly navigate through a series of *decision points*. A decision point is any area within the venue in which there is more that one possible exit route. For example, in the above diagram, the navigator starting her walk at point A has 3 available routes.

2.3 Literature Review

Route knowledge training and assessment using virtual environments have been addressed recently in studies by Witmer, Bailey, and Knerr (1995). This study presented optimistic results, stating that "virtual environments can be created that are nearly as effective as real world environments in training participants to follow a specified route." Subjects were trained to follow a specified route through a building and were divided into three conditions: a virtual environment, the actual building, and symbolically trained group which verbally rehearsed directions out loud. All three groups were also presented with study materials including step-by-step instructions on the route and color photographs of landmarks.

Results indicated that subjects who rehearsed in the building made fewer wrong turns than subjects who had rehearsed using the virtual environment, and subjects who rehearsed the route symbolically made the most mistakes. No significant differences in configuration knowledge among the various rehearsal conditions were detected, however.

A recent study by Tate, Sibert, and King (1997) showed that Virtual Environments (VE) can be effective in training shipboard firefighting and navigation in an unfamiliar space. Individuals were given 5 minutes to study DC plates (a collection of isometric views of a ship which detail the ship's structural layout). After this rehearsal period, the Traditional Training group immediately proceeded to the testing phase. The VE Training group proceeded to the VE Rehearsal prior to taking the test. The VE training consisted of three phases: a "magic carpet" walkthrough in which the motion of the subject was controlled by computer, a period in which the subject was free to explore the virtual environment, and a period of training in which the subject was free to explore the virtual environment, but in which simulated smoke was introduced in to the virtual environment, limiting visibility to approximately 3 feet.

Test results showed that there was a measurable improvement in performance in the subjects in the VE group over that of the traditional training group. In the navigation task, the VE group was faster by an average of 30 seconds (1:54 vs. 2:38). As well, all of the control group subjects made at least one wrong turn, while only one VE group subject made any wrong turns.

Pausch, Proffitt, and Williams (1997) examined the relative effectiveness of immersive head mounted displays and stationary displays. There were two tasks in the Pausch, Proffitt, and Williams study. First, subjects were placed in the center of a virtual room and told to look for camouflaged targets in room. In this test, the subjects in the immersive group did not perform significantly better than the desktop users. In the second task, the subjects were asked to search the same virtual room and to determine if the target objects existed. In the second task, subjects using the headmounted display were substantially better at determining when they had searched the entire room. In order to minimize the typical differences in field of view and resolution between desktop and head mounted displays, the stationary display used was a mounted, stationary head mounted display (with head tracking disabled).

Stoakley, Conway, and Pausch (1995) reported on a study on a user interface technique which utilizes a hand-held miniature copy of the virtual environment. Known as the *Worlds in Miniature* (WIM) metaphor, this interface offers an exocentric view of the virtual space within the 3D immersive viewport provided by the head mounted display. Manipulations made to the virtual environment are reflected in the WIM, and vice versa. Stoakley, Conway, and Pausch reports that users quickly adapt to the WIM metaphor. The study noted that important advantages of the technique include: improve visual context, removal of occlusion and line of sight problems, multiple, simultaneous points of view, and multiple scales for selecting and manipulating objects.

Satalich (1995) compared the effectiveness of various methods of training for wayfaring in a building. A virtual model of a building was created. There were four experimental conditions as follows: self exploration of the virtual environment (subjects were free to explore the space in any manner), active guided exploration of the VE (subjects moved along a predetermined path using the control apparatus), passive guided (subjects were moved along a guided path at constant speed), and a control group which did not explore the environment but studied a map of the virtual building. As well, half of the virtual environment trained subject groups were provided with maps of the virtual building.

All subjects were tested in the virtual environment and asked to perform wayfaring and distance estimation tasks. By all measures used, people who had the virtual experience either performed equivalently or worse than the control group. Satalich discusses several possibilities including limited time in the virtual environment (30 minutes) and that the VR experience may have been a distraction because it was unfamiliar.

2.4 Summary

In summary, route knowledge and memorization is the ability to reproduce a particular learned path through an environment. Configurational knowledge is knowledge of the structure of the environment.

Route knowledge seems to be less of an indicator of the navigator's spatial knowledge of a venue than configurational knowledge. It is possible for a subject to perform perfectly on a route navigation task by memorizing the predefined path at the various decision points. For example, a subject attempting to learn the route highlighted in **Figure 1** can simply memorize the sequence "right, right, left, left, right" and be able to reproduce the designated path without any spatial knowledge of the venue.

Configurational knowledge transfer lends itself more easily to real world applications such as special forces or rescue operations in which the best route may not be well defined in advance. In a firefighting or other emergency operation, for example, knowledge of the optimal route to a target area is desirable, but such knowledge is insufficient if a change in the environment engenders a change is the optimal route. If a particular memorized route is blocked or is rendered inaccessible, it becomes imperative that an alternate route may be selected. A solution to this problem through route learning is to simply memorize many such paths to the target. Such a solution is obviously impractical when the location of the target is not exactly known, a large number of decision points exist in each of these routes, or if there is a strong possibility of blocked paths.

Configurational knowledge of the space allows the navigator to select a route to the target even when the optimal route is blocked. Because the navigator knows the structure of the venue, she is able to construct an alternate route if there is a minor change to the topology of the venue, and is theoretically able to find a new best path if new options for travel become available (e.g. a hole in the wall or a new opened door).

3.0 Experiment #1

3.1 Motivation

The aim of this experiment is to establish the efficacy of virtual environments as a training medium for spatial knowledge acquisition. The primary goal of this experiment is to determine the extent of which configurational knowledge may be acquired by subjects exploring a high-resolution photorealistic virtual environment. Subjects were assessed during the testing phase by a series of pointing tasks in which they were asked to point to and estimate the distance to landmarks in the real-world environment.

Establishing the level of efficacy of high-resolution virtual environment as a training tool will serve as a benchmark by which factors affecting learning in a virtual environment may be analyzed.

The secondary aim of this project is to determine the relative effectiveness of several types of computer aided training systems. The literature on virtual environment navigation training introduces several questions which have not been resolved. The study by Witmer, Bailey, and Knerr (1995) suggests that virtual environments may be used as a training aid for route knowledge, but produced inconclusive results with regards to configurational knowledge training. Tate, Sibert, and King (1997) demonstrated that training in an immersive virtual environment is a useful supplement to studying 2 dimensional charts of a given venue with regards to navigational ability. Satalich (1995) examined various forms of immersive virtual environment training and found little difference in the results of those who had trained using the virtual environment when compared with the control group which trained only with maps.

None of the above studies have shown the effectiveness of immersive virtual environments as a compelling training tool. Both the Witmer, Bailey, and Knerr

(1995) and the Tate, Sibert, and King (1997) studies utilized virtual environment training as a supplement to traditional training methods. The Satalich (1995) study compares map and VE training and showed little advantage in training using the VE.

The goal of this study is to compare the efficacy of different forms of computer aided training in order to determine which is the most effective. Four experimental training conditions were developed: immersive Virtual Environment (VE), non-immersive Virtual Environment (NVE), exocentric Virtual Environment (Model), and real world (RW). Subjects are randomly assigned to the four training groups and then asked to perform the same testing tasks. This study will allow for a determination of the effectiveness of an immersive based virtual environment system versus real world training, as well as an assessment on the effectiveness of two other types of computer training systems.

3.2 Apparatus and Environment

The virtual environment walkthroughs in this study were run on modified Easyscene software from Coryphaeus running on an SGI Onyx RE/2 The Onyx was equipped with two R4400 processors operating at 150 Mhz and 128 MB of RAM. The Easyscene software was modified to allow for a first person walkthrough through a developed model with collision detection and joystick support added. Modifications were also made to enable support for the headmounted display and adaptation of multiple viewpoints and control methods for the immersive, non-immersive, and exocentric experimental conditions.

The head mounted display was a Virtual Research VR4. The head mounted display had a horizontal resolution of 350 lines and a vertical resolution of 230 lines and a 60 degree field of view. Weight of the headmounted display unit was 33 ounces. The head mounted display was not stereo enabled for this study. Fred Books (1992) stated

that "stereo noticeably adds to the illusion of presence, but it makes much less difference in architectural modeling than in most applications. Architectural models have lots of parallel lines and planes, known to be perpendicular... Obscuration, perspective, and the kinetic depth effect provide so strong a set of cues that stereo is almost frosting."

The orientation of the HMD was determined by an attached Polhemus 3Space Fastrak sensor which provided orientation and position information to the walkthrough software. The pitch, roll, and yaw information is received by the walkthrough software and translated to a properly corresponding image in the HMD. The positional information is discarded. All positional translation is accomplished through the use of a joystick for navigation.

Subjects in the non-immersive virtual environment condition used a 21 inch monitor running in a 640x480 graphics mode.

Both VE subjects used a joystick, which moved the subject forward when it was pushed forward, backward when it was pushed backward, and rotated the subject in the VE left and right when it was pushed in the associated direction. Subjects were also able to navigate by facing or turning their head and looking at their desired heading, as the orientation of the HMD also determines the orientation of the subject in the VE.

A tripod with attached pointer and protractor was used to determine pointing heading during the testing phase.

A wheelchair was used to transport the blindfolded subjects between the four pointing locations.

The environment chosen for the study the was the 7th floor of MIT Building 36. This location was chosen because of its proximity to the virtual environment simulator and its easy accessibility.

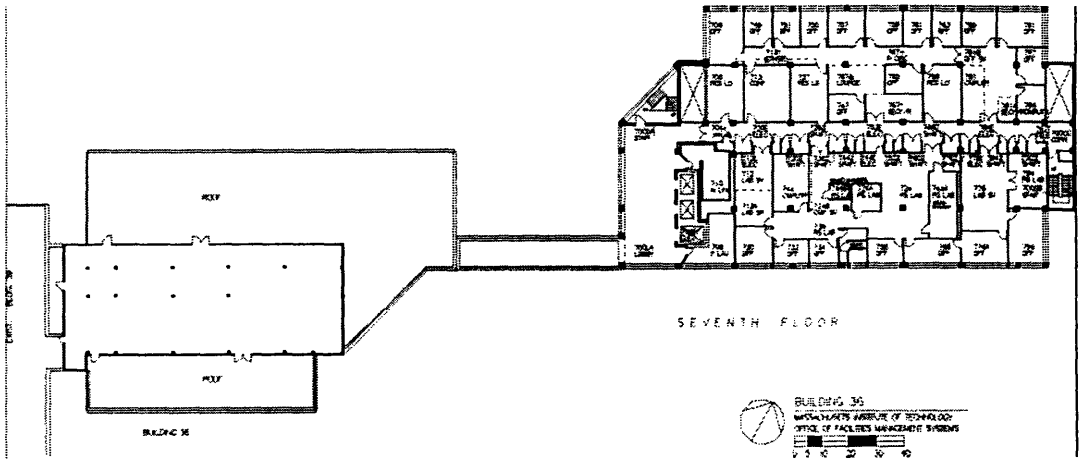


FIGURE 6. Blueprint for the 7th floor of MIT Building 36.

An architectural model was constructed using Coryphaeus' Designers Workbench software from blueprints of MIT Building 36 obtained through the MIT Office of Facilities Management. Accessible areas of the virtual environment were modeled with open doorways. Any closed doors in the virtual environment signified inaccessible areas irrelevant to the experiment. Motion of the subject was confined to the X and Y plane. Subjects were unable to go up or down stairs or utilize the elevators. Each room in the model was populated with objects created from within Designers Workbench. All objects were fully texture mapped from within Designer's Workbench. The actual textures were obtained by photography with both a Nikon N6006 camera with film scanned at 1280x1024 by Konica and a Kodak DC20 Digital Science digital camera. Texture maps were edited using Kodak PhotoEasy software and Adobe Photoshop, and imported into Designer's Workbench for application to the Building 36 model. Textures mapped objects were used as landmarks in the training conditions and pointing task.

3.3 Experimental Methods

The experiment focused upon the relative efficacy of training methods for spatial knowledge acquisition. Three computer modeled conditions were used in conjunction with a control group of subjects trained in actual 7th floor of Building 36.

Subjects were informed of the goal of the experiment, and told that they were to observe as carefully as possible the environment during the training phase, and that they were to be asked to identify certain significant objects during the testing phase. Subjects were told to note the relative spatial configuration of the venue, and not specifically memorize characteristics of individual objects. As this study attempts to focus upon configurational knowledge acquisition, the targeted objects were chosen to be very prominent.

Subjects were asked to read the COUHES agreement on human subject experimentation to sign. Following this, they were introduced to the training phase of the experiment. Subjects were randomly assigned to one of the four training conditions: immersive virtual environment (using a headmounted display to explore a computer model of the venue), non-immersive virtual environment (using a computer monitor to explore a computer model of the venue), non-immersive virtual model (manipulation of a computer model of the venue), and control (exploring the actual venue).

After the specified training time, subjects were brought to the 7th floor of MIT Building 36 and entered the testing phase. Subjects in the testing phase were asked to point to a series of objects from their current location using a tripod mounted pointer. The subjects were also asked to estimate the straight line distance between their current location and the targeted object. When the series of pointing and rangefinding tasks was completed, the subjects were placed in a wheelchair, blindfolded, and rolled

to a new pointing location. Subjects were blindfolded and wheeled in order to isolate them as much as possible from learning contamination acquired during movement from one pointing location to the next.

Upon the completion of the entire series of pointing and rangefinding tasks, the subjects were asked to fill out two surveys on the virtual environment and immersion.

3.4 Training Phase

Subjects were informed during the training phase of the goals of the experiment and were told that they were to fully explore the training environment. Subjects were informed of the nature of the task which they were to perform in the testing phase and were informed to fully appreciate major environmental details during the training phase.

The four training methods used were as follows:

1. Real World Environment
2. Egocentric Immersive Virtual Environment
3. Egocentric Non-Immersive Virtual Environment
4. Exocentric Virtual Model

3.4.1 Real World Environment

Subjects in the real world training environment were allowed 10 minutes to explore the experimental environment (MIT Building 37, 7th Floor). Subjects were allowed to explore the venue for 10 minutes in any manner that they chose, but were informed by the experimenter of the areas which are relevant to the study and of off-limits areas (privately occupied regions of the building).

3.4.2 Egocentric Immersive and Non-Immersive Virtual Environments

The Egocentric Immersive and Non-Immersive training groups operated a first person walkthrough simulation of the experimental venue. The group in the immersive condition utilized the Virtual Research headmounted display and the non-immersive group utilized a 21 inch monitor to visualize the rendered virtual environment. Both groups utilized a joystick as the primary control method. The joystick was configured for both groups such that a forward push on the joystick moved the subject forward, a backward push moved the subject backwards, and left and right pushes rotated the subject in place in that direction. Subjects in the immersive virtual environment were also instructed in the use of the headmounted display.

TABLE 1. Joystick and headmounted display control modes

	Joystick	**Headmounted Display**
Up	Moves forward	Pitch up
Down	Moves backward	Pitch down
Left	Rotates left	Yaw left, Rotates left
Right	Rotates left	Yaw Right, Rotates right
Tilt Left		Roll left
Tilt Right		Roll right

Subjects in the two egocentric virtual training environments were allowed to familiarize themselves with the operation of their respective training systems by exploring a simple training virtual environment.

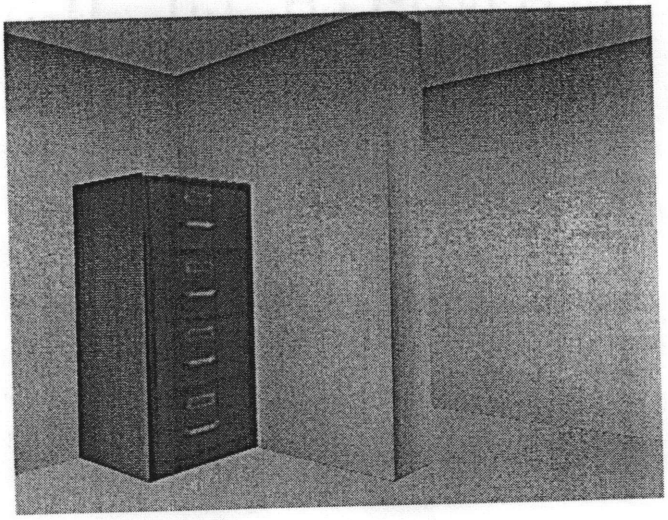


FIGURE 7. Training environment used to familiarize subjects with the virtual environment control system.

The training model consisted of a simple loop with only a file cabinet as a distinguishing feature. This model was topologically independent of the actual experimental area and served only to allow the subjects to become experienced mechanics of joystick operation and simple navigation of a virtual environment. The subjects in these two groups were allowed as much time as necessary to familiarize

themselves with and become comfortable with the operation of the walkthrough software and apparatus.

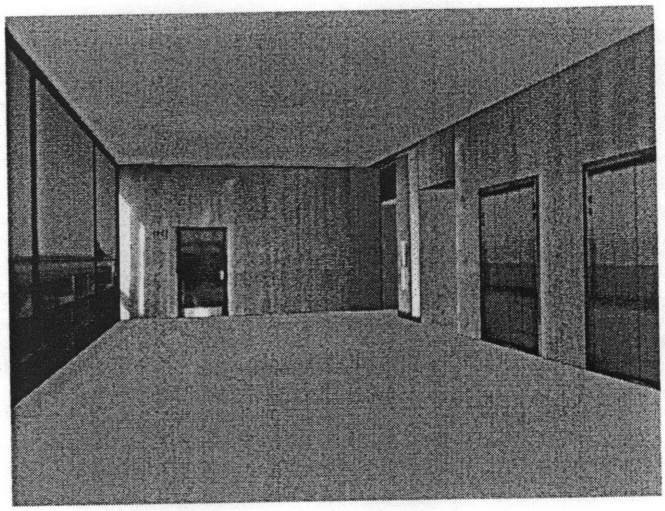


FIGURE 8. Starting point of the Building 36 virtual model.

The subjects in the egocentric virtual environment groups were introduced to the virtual model of Building 36 after having indicated their familiarity and demonstrated competence in the operation of the simulator while exploring the training environment. Subjects were allowed to freely explore the Building 36 virtual model for 10 minutes after an initial tour of the model was performed by the experimenter.

3.4.3 Exocentric Virtual Model

The exocentric virtual model condition is a three dimensional equivalent to a map. By utilizing 3D visualizing software, subjects are able to manipulate a model of the Building 36 environment using a mouse and monitor. Subjects were able to rotate and view the model from any vantage point as well as zoom in closely to examine specific details. The model used in the exocentric virtual model condition is almost exactly identical to the model utilized in the two virtual walkthrough conditions. The only

differences are the point of view change and the removal of the ceiling in order to facilitate the usage of the virtual model as intended.

Subjects in the virtual training model were allowed to familiarize themselves with the operation of the 3D visualizer by manipulating a simple training model. Manipulation of the virtual model was accomplished as follows:

TABLE 2. Model manipulation controls

Portion of Screen Clicked	Middle Mouse Button	Right Mouse Button
Top	Zooms In	Translates Up
Bottom	Zooms Out	Translates Down
Left	Rotates Left	
Right	Rotates Right	

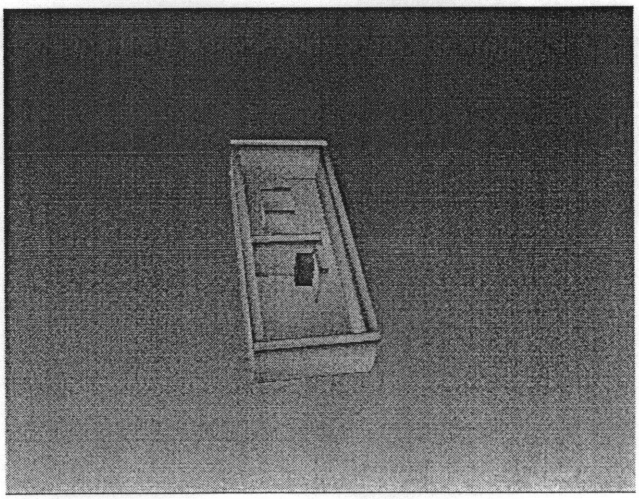


FIGURE 9. Training model used to familiarize subjects with the virtual model manipulation system.

The training environment was independent of the Building 36 model and was utilized only to familiarize the subjects with the manipulation of the virtual model. Subjects

were allowed as much time as necessary to familiarize themselves with and become comfortable with the operation of the training model.

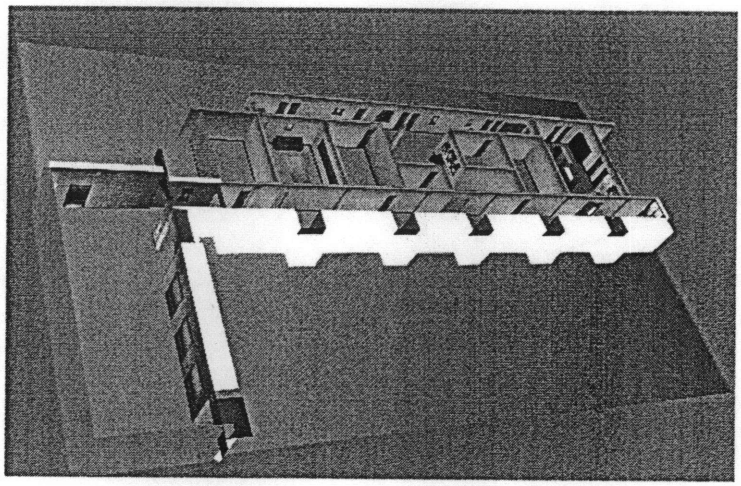


FIGURE 10. Building 36 virtual model used in training session.

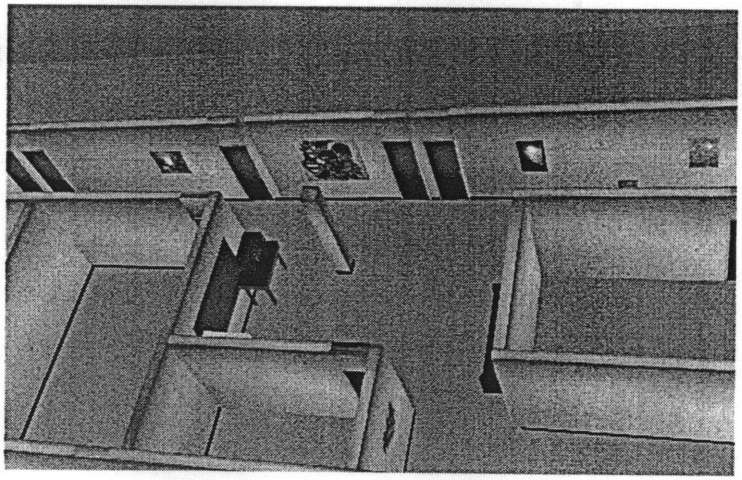


FIGURE 11. Zoomed and rotated view of the Building 36 virtual model.

The subjects in the exocentric virtual model group was introduced to the virtual model of Building 36 after having demonstrated competence in the operation of the 3D

visualizer. Subjects were allowed to freely manipulate the Building 36 model for 10 minutes. Subjects informed that they were free to ask the experimenter the names or descriptions of objects which may be unclear within the virtual model.

3.5 Testing Phase

The subjects from the four conditions were tested in the actual venue The primary task which was to be performed by the subjects was a *pointing task* in which the subjects were asked to point the tripod mounted pointing device to various landmarks within the building and asked to estimate the distance from their pointing location to the landmark (pointing target).

Landmarks were generally obscured by walls so that the subject would be unable to use line of sight as a cue to targeting the landmark. Subjects were also asked to point to certain landmarks which were in their line of site. The objective of these targets are two-fold: they provide a means of calibrating distance estimations from the subjects and also provide a way of estimating the minimum bearing error for different subjects.

Subjects were brought to the pointing locations marked in the figure in the order shown, and were asked to point to and estimate the distance to several objects from each pointing location. The locations were chosen to minimize the amount of

knowledge of the environment gained by the subjects while performing the pointing task in each of the pointing locations.

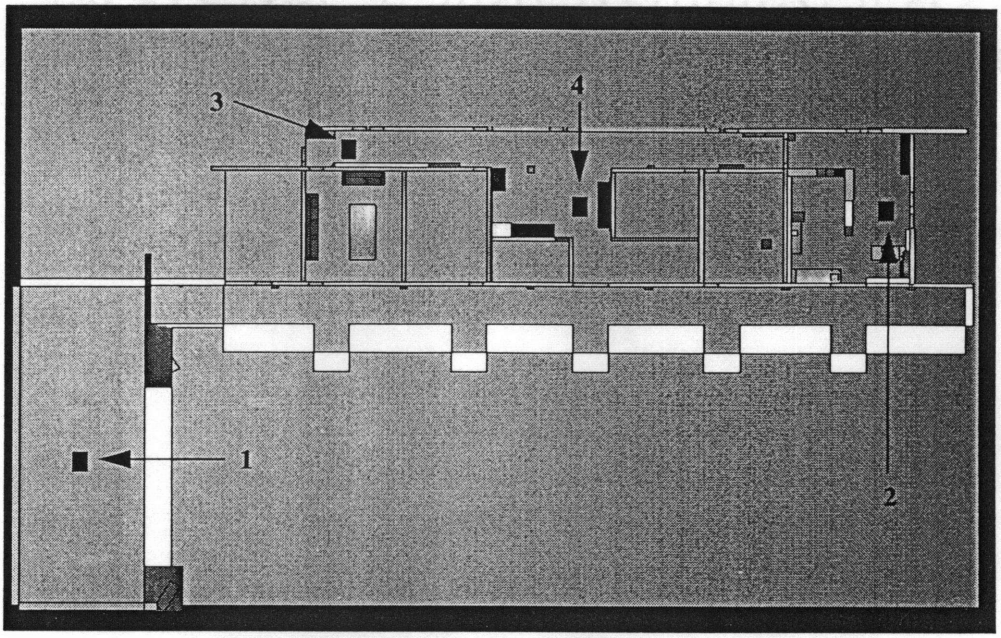


FIGURE 12. Overhead view of Building 36 7th floor floorplan with pointing locations labelled.

The subjects were blindfolded and transported to the successive pointing locations on a wheelchair in order to minimize spatial knowledge acquisition during the testing phase.

The following landmarks were used as targets from each of the pointing locations:

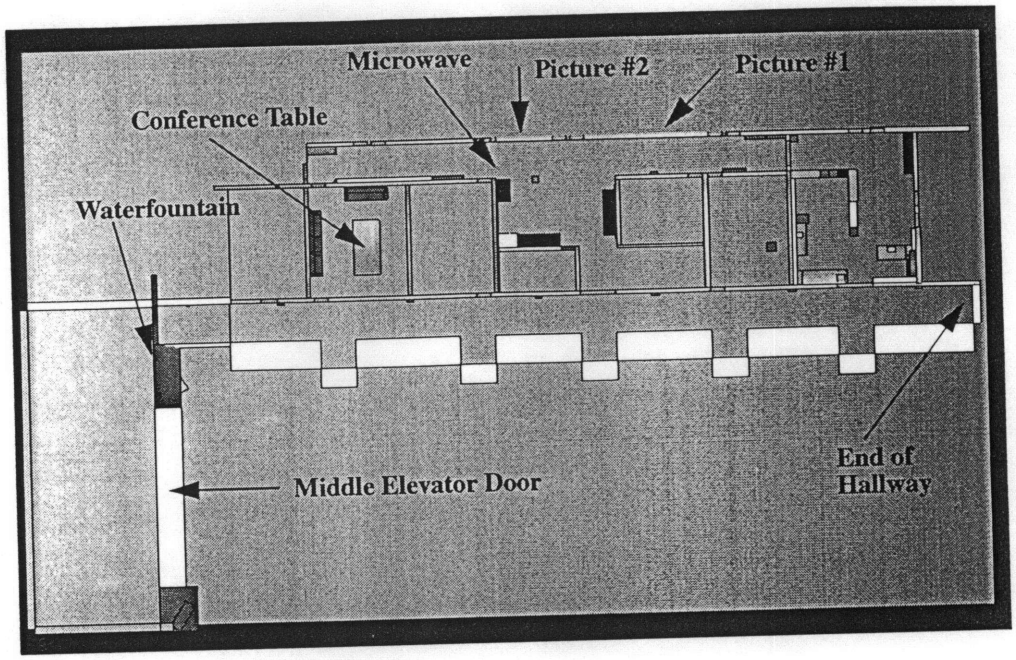


FIGURE 13. Landmarks used as pointing targets.

TABLE 3. Pointing targets from various locations.

	Water Fountain	Micro-wave	Conf. Table	End of Hallway	Middle Elevator	Picture #1	Picture #2
1.					■		
2.	■		■				
3.			■			■	■
4.	■	■					

Targets labelled **Picture #1** and **Picture #2** were shown to the subjects in the form of a photograph of the picture to be targeted. All four experimental conditions were asked to point to the same target objects in the same sequence.

3.6 Observations

A measure of the angular error is given in **3.6.1**. The non-immersive Virtual Environment (NVE) and Model groups produced the best results, with averaged mean errors (confidence intervals) of 10.07 (1.76) and 10.11 (1.94) degrees respectively. The Real World (RW) and Virtual Environment (VE) groups had error scores of 11.92 (1.58) and 12.18 (1.69) respectively. The analysis of variance found an Obtained F of 1.22 and a Criterion F of 2.37.

Analysis of distance estimation is given in **3.6.2**. The Model and NVE conditions fared best with averaged mean errors (confidence intervals) of 19.46 (4.5) ft. and 19.36 (4.97) ft. followed by the VE and RW conditions at 24.76 (4.1) ft. and 33.61 (4.06) ft. respectively. Analysis of variance found an Obtained F of 7.02 and a Criterion F of 2.37.

The subjects' bearing and distance estimates were combined with their initial pointing location to recreate the cartesian coordinates of the subjects' perceived location of the targets. The magnitude of the error vector between these points and the actual location of the targets are detailed in **3.6.3**. The Model and NVE conditions fared best with averaged mean errors (confidence intervals) of 25.49 (5.15) ft. and 27.48 (5.69) ft. followed by the VE and RW conditions at 33.14 (4.77) ft. and 40.29 (4.64) ft. respectively. Analysis of variance found an Obtained F of 5.25 and a Criterion F of 2.37.

It is important to note that the summaries presented above and in section **3.6** as a whole are based upon experimental data collected on pointing tasks from the four locations to a number of major landmarks. These landmarks were the waterfountain, microwave, conference table, end of hallway, and middle elevator, and are identified in **Figure 13**. The pointing locations are identified in **Figure 12**. The two pictures were not used for the purposes of this study as location of these two landmarks are more memory dependent than the other -- more obvious -- landmarks.

3.6.1 Angular Error

This test was an indicator of the ability of the subjects to point to a given location.

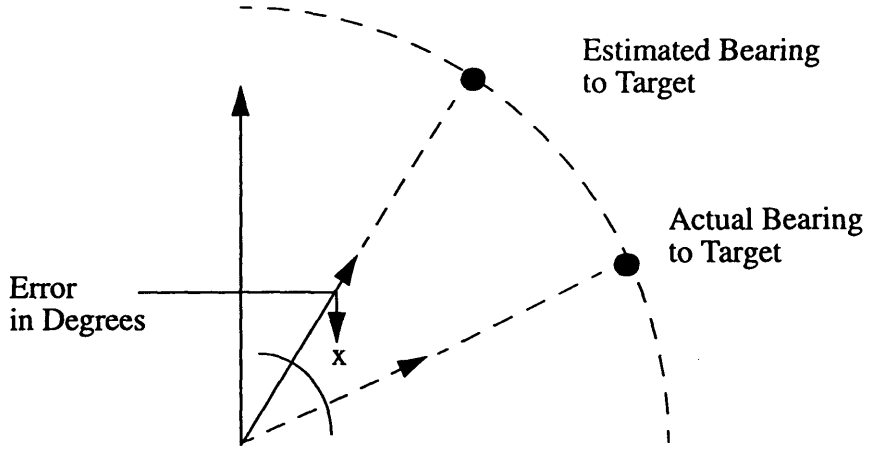


FIGURE 14. The error calculated by taking the magnitude of the difference between the actual bearing to the target and the estimated bearing to the target.

Table 4 shows the magnitude of the difference between the actual bearing and the predicted bearing, averaged over the subjects in each condition.

TABLE 4. Pointing Test Mean Bearing Error (deg.)

	VE	RW	NVE	Model
Microwave 1	7.4	11.7	8.9	2.9
Conf. Table 1	13.3	12.1	8.3	11.8
End of Hallway 1	20.9	16.6	20.8	12.4
Microwave 2	13.1	6.7	8.4	6.6
End of Hallway 2	26.3	16.3	17.6	9.8
M. Elevator 2	10.7	18.1	13.1	11.7
W. Fountain 3	11.8	14.4	9.1	11.8
Microwave 3	9.2	10.6	5.2	15.3
End of Hallway 3	8.2	6.6	7.1	9.8
M. Elevator 3	7.1	7.6	5.6	9.2
Conf. Table 4	6.1	6.4	8.3	5.1
End of Hallway 4	17.0	16.0	17.6	11.6
M. Elevator 4	4.4	11.4	7.4	13.2

Table 5 gives the standard deviation for each set of subjects across a given Target/ Location and Condition.

TABLE 5. Pointing Test Error Standard Deviation

	VE	RW	NVE	Model
Microwave 1	4.4	6.4	4.7	4.6
Conf. Table 1	5.4	6.6	4.9	4.9
End of Hallway 1	10.8	6.7	4.8	5.9
Microwave 2	12.7	6.7	7.8	3.9
End of Hallway 2	27.3	9.2	7.8	8.6
M. Elevator 2	6	9.6	7	6.6
W. Fountain 3	8.3	6.4	7.1	7.6
Microwave 3	9.2	9	4.4	13.4
End of Hallway 3	4.2	4.7	3	8.8
M. Elevator 3	3.3	4.9	3.0	6.4
Conf. Table 4	4.2	5.3	6.2	6.7
End of Hallway 4	6.6	7.3	12.0	11.6
M. Elevator 4	5.9	8.9	5.8	8.9

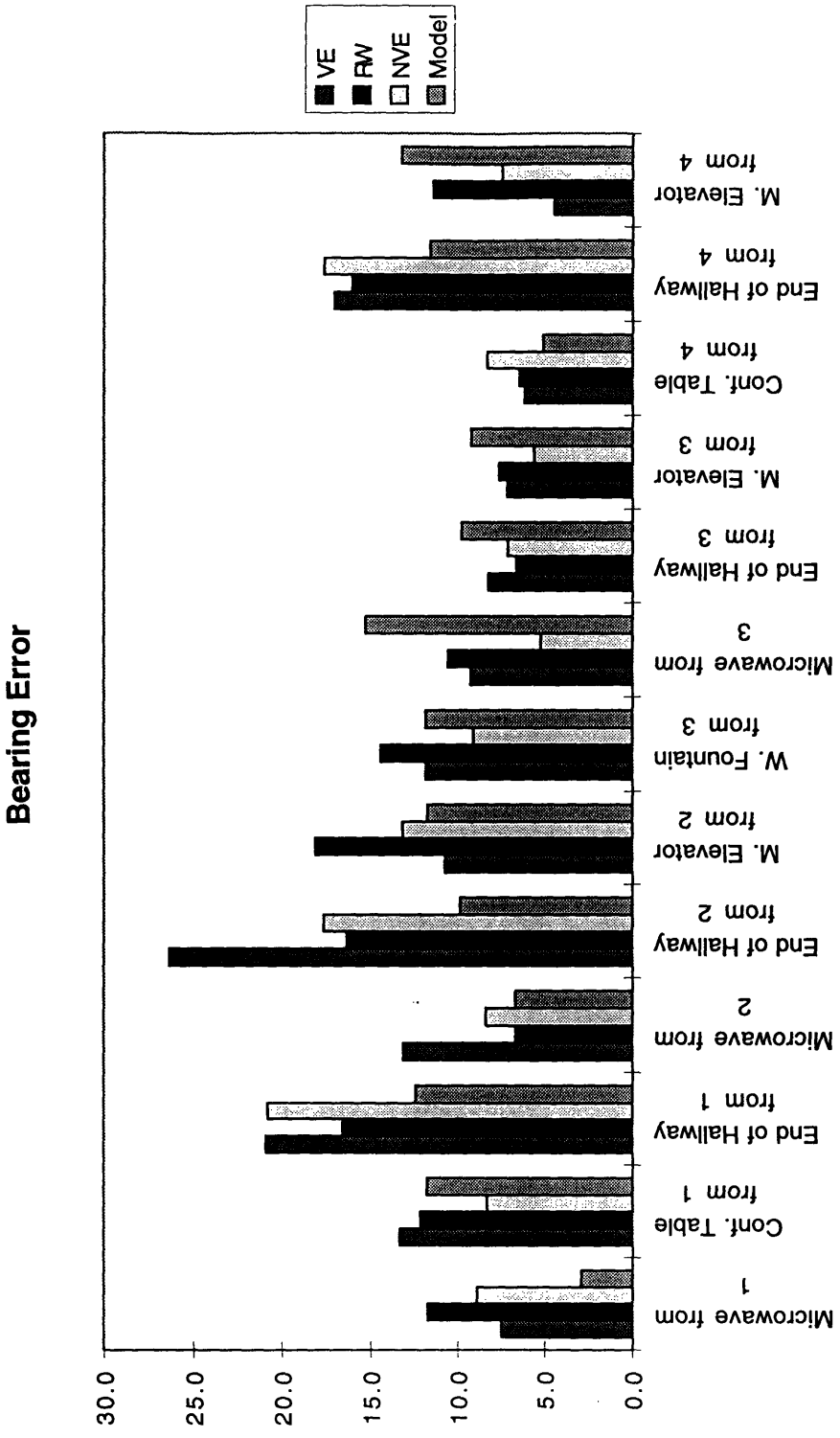


FIGURE 15. Bearing Error averaged over subjects in each Location/Target pair and Experimental Condition.

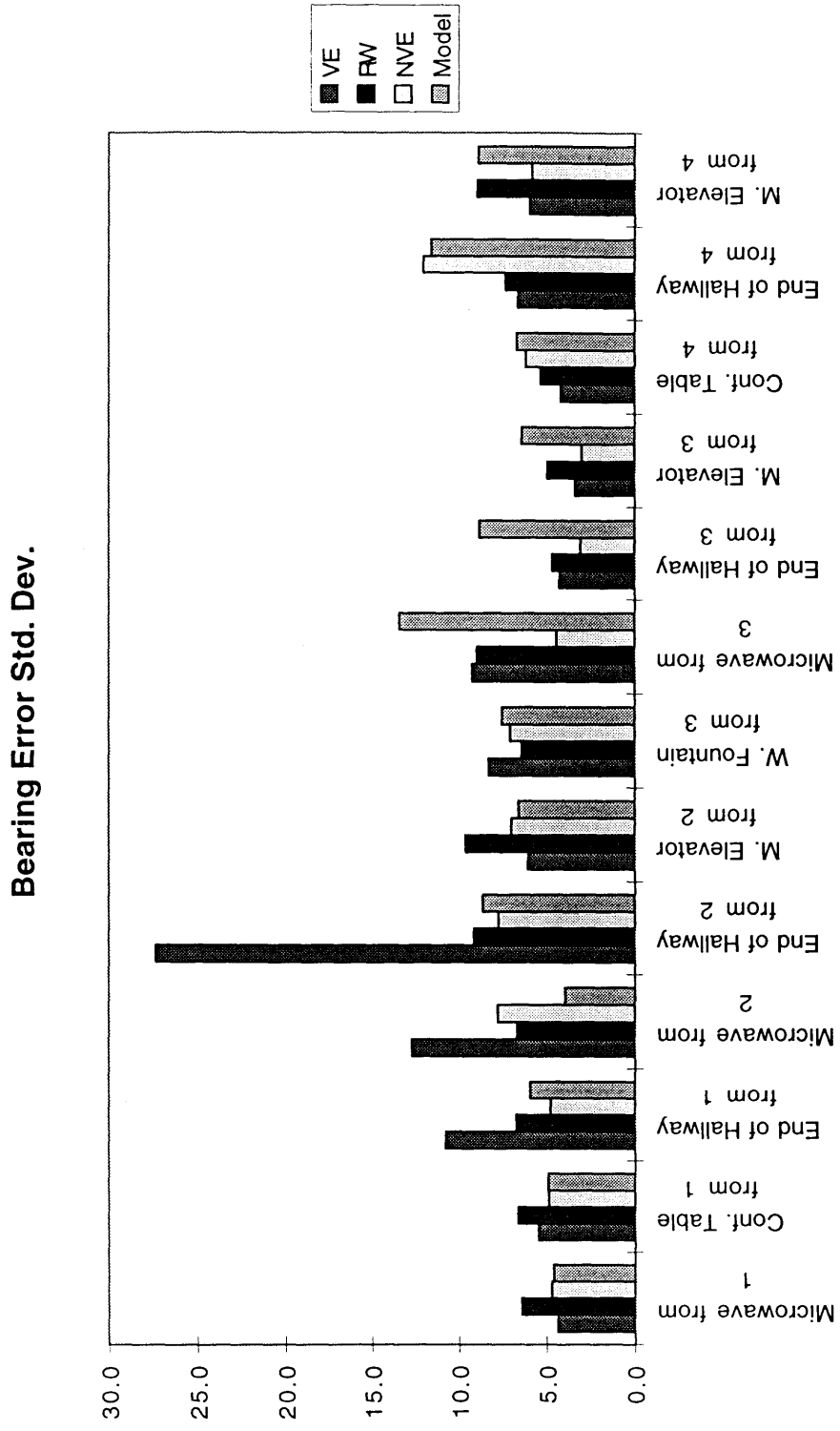


FIGURE 16. Standard Deviation of the Bearing Error for subjects in each Location/Target pair and Experimental Condition.

An *analysis of variance* (ANOVA) was performed to determine the statistical significance of the observations. ANOVA involves a consideration of two different types of variance: the population variance (which characterizes the scores within a given condition) and the variance between groups. The determination of the significance of a training conditions score is directly related to its variance relative to the variance of the other individuals in the study.

Results for the ANOVA on the mean bearing error at a confidence level of .95 resulted in the following confidence intervals:

TABLE 6. ANOVA for Mean Bearing Error

Condition	Mean Error	Confidence Interval
VE	12.18	1.69
RW	11.92	1.58
NVE	10.11	1.94
Model	10.07	1.76

TABLE 7. ANOVA Summary Table for Mean Bearing Error

Source	Degrees of Freedom	Sum of Squares	Mean Square	Obtained F	Criterion F
Between	4	406	101.5	1.22	2.37
Within	425	35472.21	83.46		

Note that the Obtained F value of 1.22 is smaller than the Criterion F value of 2.37. This suggests that we should accept the null hypothesis that the training conditions have no effect on performance in the bearing task.

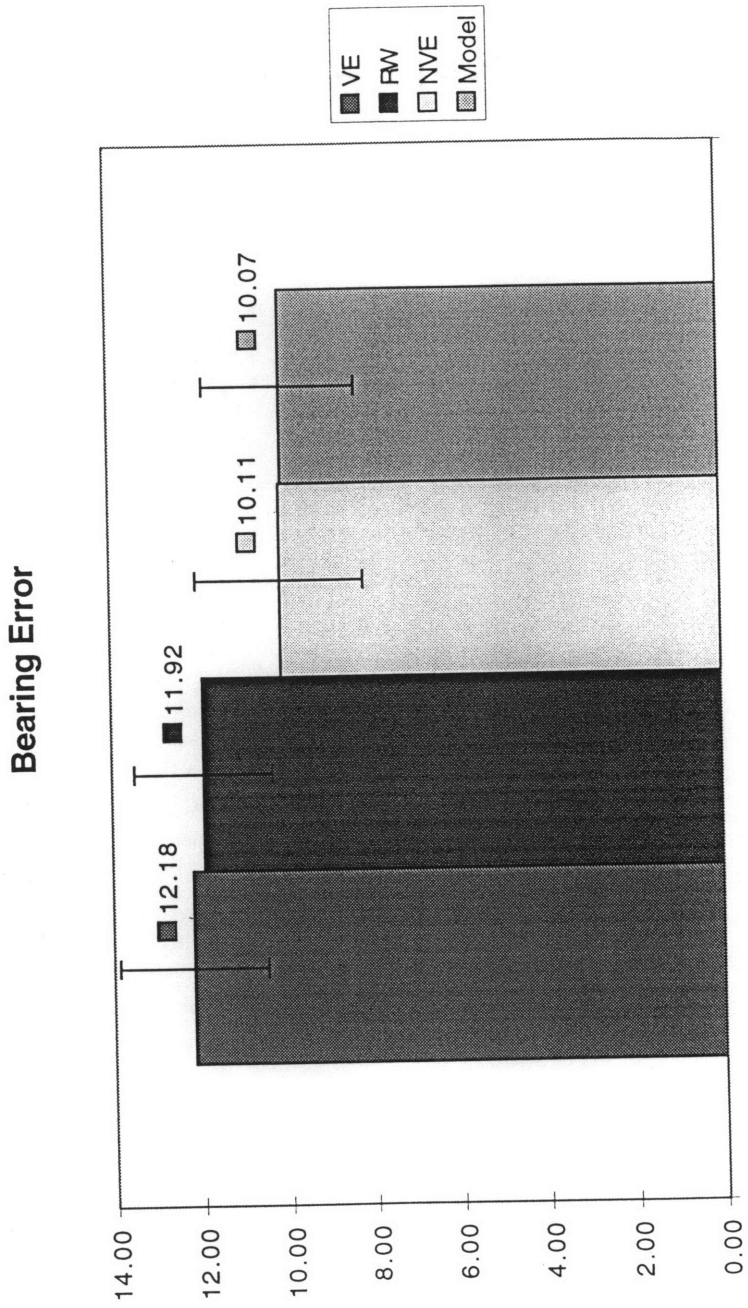


FIGURE 17. Mean Bearing Error for all Location/Target pairs. The Error Bars indicate the confidence interval as derived from the ANOVA test on the data set.

3.6.2 Distance Estimation Error

The difference between the subjects' estimation of the distance from their pointing location to the target object are given and the correct value for all pairs of pointing locations are given in **Table 8.**

TABLE 8. Pointing Test Mean Distance Error (ft.)

	VE	RW	NVE	Model
Microwave 1	22.3	38.4	26.0	19.7
Conf. Table 1	14.1	25.2	20	21.2
End of Hallway 1	43	100.3	32.6	23.9
Microwave 2	23.6	29.2	21.9	23.6
End of Hallway 2	8.4	13.5	4.6	8.4
M. Elevator 2	35.1	66.3	34.7	15.2
W. Fountain 3	14	14	16.6	34.7
Microwave 3	13.3	9.8	8.4	18.9
End of Hallway 3	32.9	69.5	13.8	66.0
M. Elevator 3	16.1	16.5	17	30.7
Conf. Table 4	20.5	26	24.6	9.9
End of Hallway 4	21.8	47.1	20.5	51.7
M. Elevator 4	21.0	27.9	24.4	16.3

Table 9 gives the standard deviation for each set of subjects across a given Target/ Location and Condition.

TABLE 9. Pointing Test Standard Deviation

	VE	RW	NVE	Model
Microwave 1	22.3	38.4	26.0	19.7
Conf. Table 1	14.1	25.2	20	21.2
End of Hallway 1	43	100.3	32.6	23.9
Microwave 2	23.6	29.2	21.9	23.6
End of Hallway 2	8.4	13.5	4.6	8.4
M. Elevator 2	35.1	66.3	34.7	15.2
W. Fountain 3	14	14	16.6	34.7
Microwave 3	13.3	9.8	8.4	18.9
End of Hallway 3	32.9	69.5	13.8	66.0

TABLE 9. Pointing Test Standard Deviation

	VE	RW	NVE	Model
M. Elevator 3	16.1	16.5	17	30.7
Conf. Table 4	20.5	26	24.6	9.9
End of Hallway 4	21.8	47.1	20.5	51.7
M. Elevator 4	21.0	27.9	24.4	16.3

Results for the ANOVA on the mean bearing error at a confidence level of .95 resulted in the following confidence intervals:

TABLE 10. ANOVA for Mean Distance Error

Condition	Mean Error	Confidence Interval
VE	24.76	4.1
RW	33.61	4.06
NVE	19.36	4.97
Model	19.46	4.5

TABLE 11. ANOVA Summary Table for Mean Distance Error

Source	Degrees of Freedom	Sum of Squares	Mean Square	Obtained F	Criterion F
Between	4	16598.65	4149.66	7.02	2.37
Within	473	279629.28	591.18		

Note that the Obtained F value of 7.02 is considerably larger than the Criterion F value of 2.37, indicating that we should reject the null hypothesis that there is no correlation between the training conditions with regards to the distance estimation task.

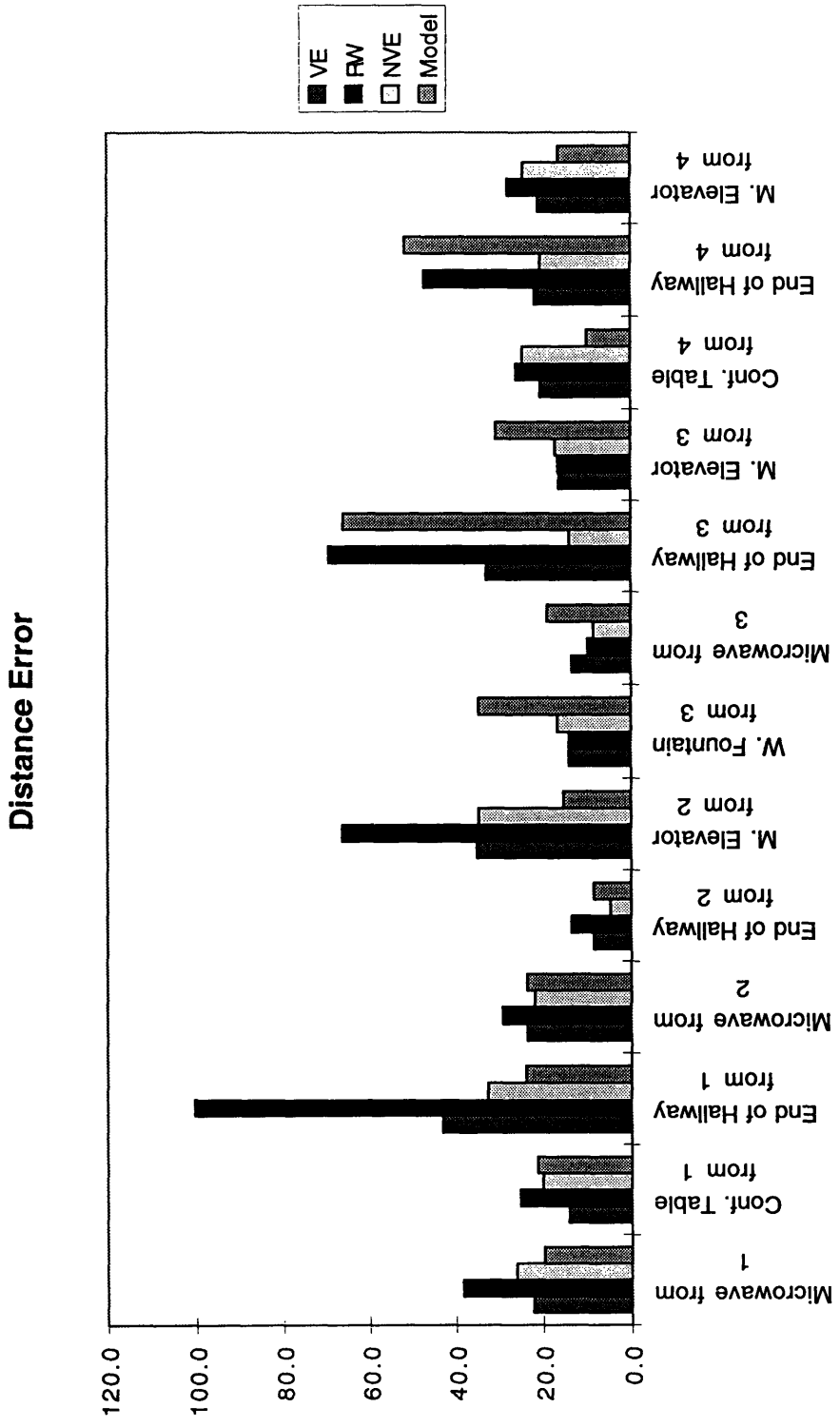


FIGURE 18. Distance Estimation Error averaged over subjects in each Location/Target pair and Experimental Condition.

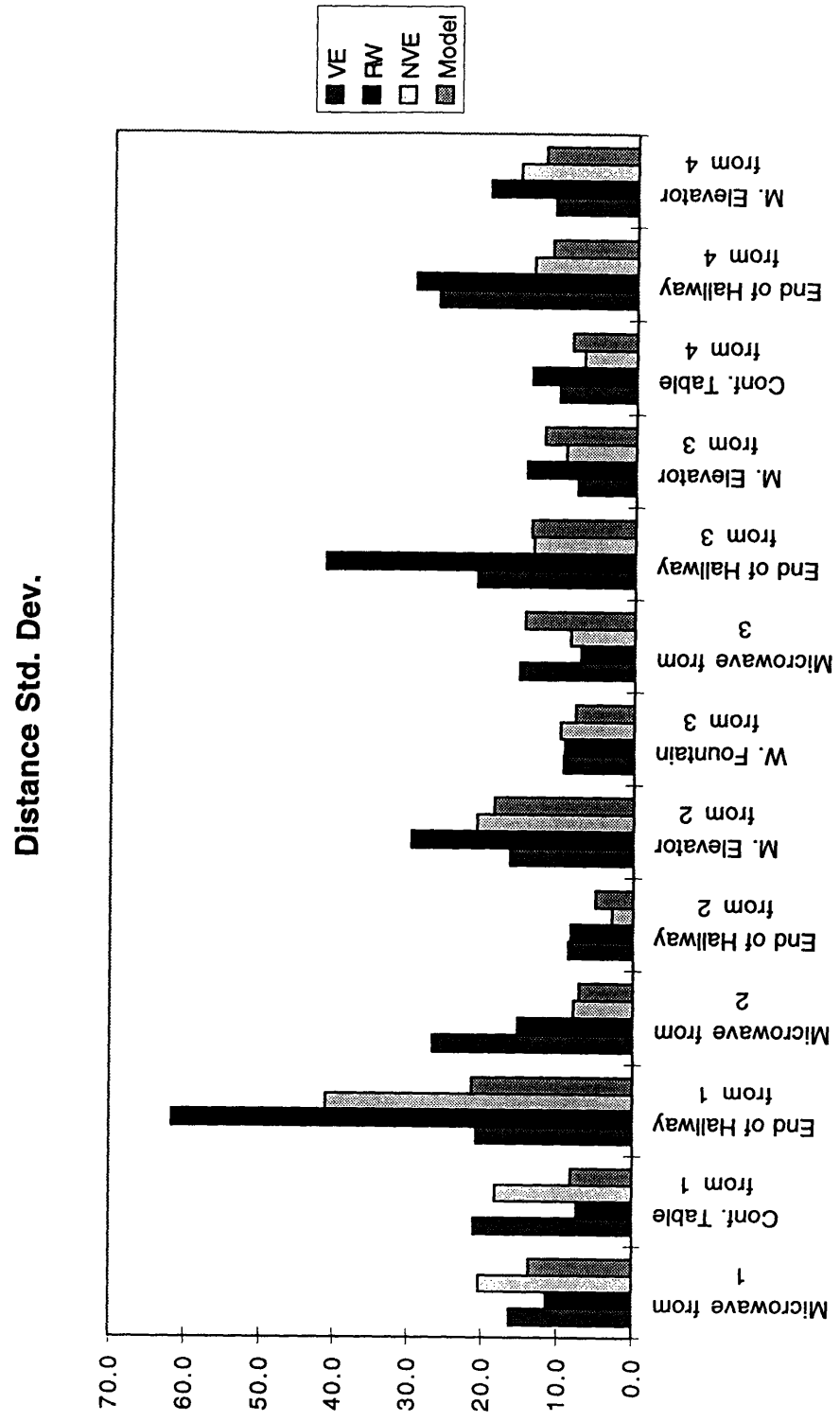


FIGURE 19. Standard Deviation of the Distance Estimation Error for subjects in each Location/Target pair and Experimental Condition.

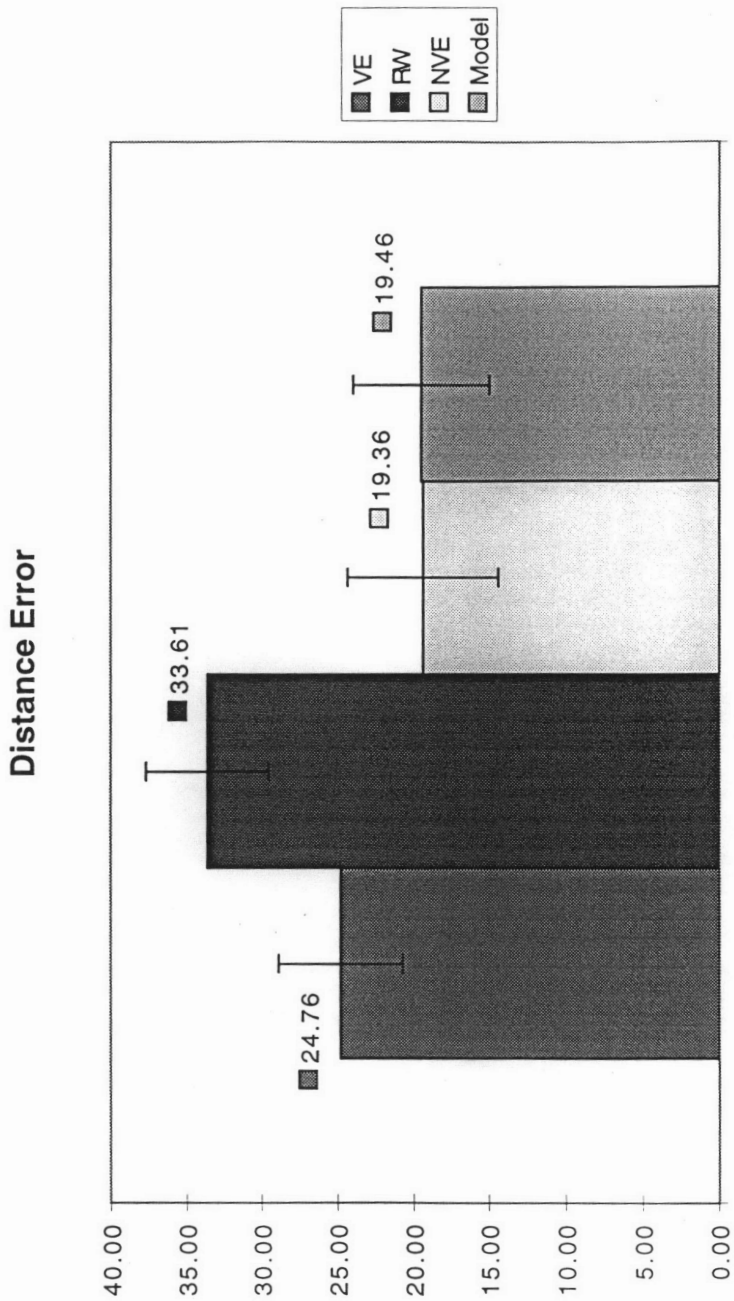


FIGURE 20. Mean Distance Estimation Error for all Location/Target pairs. The Error Bars indicate the confidence interval as derived from the ANOVA test on the data set.

3.6.3 Cartesian Displacement

The bearing and distance estimates of each subject from each of the pointing and target pair locations were used to generate an XY plot of the subject's estimated location of the target object.

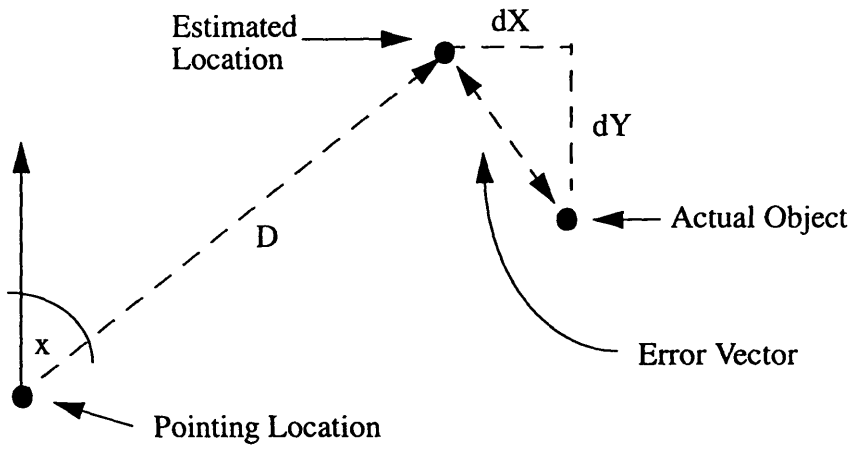


FIGURE 21. The subject's estimate location of the target is determined using his bearing and distance estimates. The error vector is the magnitude of the distance between the estimated location of the target object and the actual location of the object.

The magnitude of the error vector is used as an indicator of the ability of the subject to point to a given object. Error values from each pair of pointing location and target were found for each subject, and then averaged over all the subjects in their respective experimental conditions.

The results are displayed in **Table 12**, below.

TABLE 12. Magnitude of Error Vectors (ft.)

	VE	RW	NVE	Model
Microwave 1	36.3	36.6	37.2	22.1
Conf. Table 1	23.3	25.3	23.2	14.6
End of Hallway 1	70.0	106.1	69.7	54.0
Microwave 2	40.1	31.9	24.1	22.6
End of Hallway 2	14.8	14.5	8.3	7.6
M. Elevator 2	51.9	75	47.3	46.2
W. Fountain 3	18.6	19.1	19.0	17.0
Microwave 3	22.6	12.6	9.2	18.4
End of Hallway 3	49.0	65.5	21.8	26.8
M. Elevator 3	18.6	23.3	18.3	27.0
Conf. Table 4	24.8	26.6	26.0	19.3
End of Hallway 4	34.8	47.2	32.0	20.6
M. Elevator 4	28.6	37.1	27.4	35.2

By combining the bearing and distance tasks, these results should give an indication of the subjects spatial awareness. It is interesting to note that the Real World (RW) training group performed the most poorly, while the non-immersive Virtual Environment (NVE) and Model conditions performed similarly well. Subjects in the RW and VE conditions performed especially poorly in the longer distance estimations (End of Hallway from 1, Middle Elevator from 2, and End of Hallway from 3).

TABLE 13. Standard Deviation of the Magnitude of Error Vectors

	VE	RW	NVE	Model
Microwave 1	21.0	20.8	20.2	21.1
Conf. Table 1	18.6	19.7	19.2	18.7
End of Hallway 1	33.0	33.6	34.7	26.9
Microwave 2	37.3	39.9	39.2	40.4
End of Hallway 2	14.5	14.9	14.7	14.1
M. Elevator 2	24.5	25.7	30.7	29.9
W. Fountain 3	9.0	9.3	9.5	9.1
Microwave 3	17.1	17.9	17.9	17.3

TABLE 13. Standard Deviation of the Magnitude of Error Vectors

	VE	RW	NVE	Model
End of Hallway 3	32.8	34.8	34.6	32.9
M. Elevator 3	6.8	7.2	7.8	6.6
Conf. Table 4	10.4	8.8	9.1	10.9
End of Hallway 4	25.4	27.1	27.1	26.8
M. Elevator 4	20.2	20.6	21.8	19.3

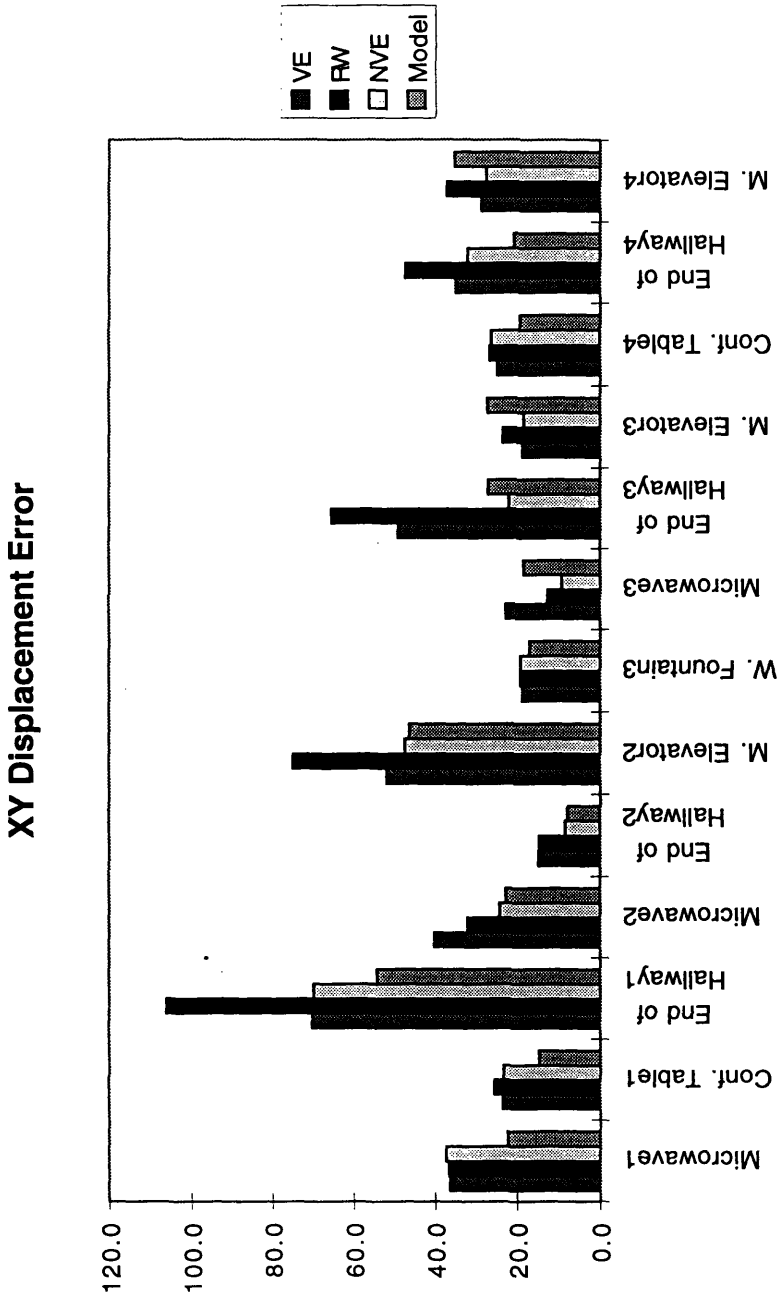


FIGURE 22. Magnitude of Error Vectors averaged over subjects in each Location/Target pair and Experimental Condition.

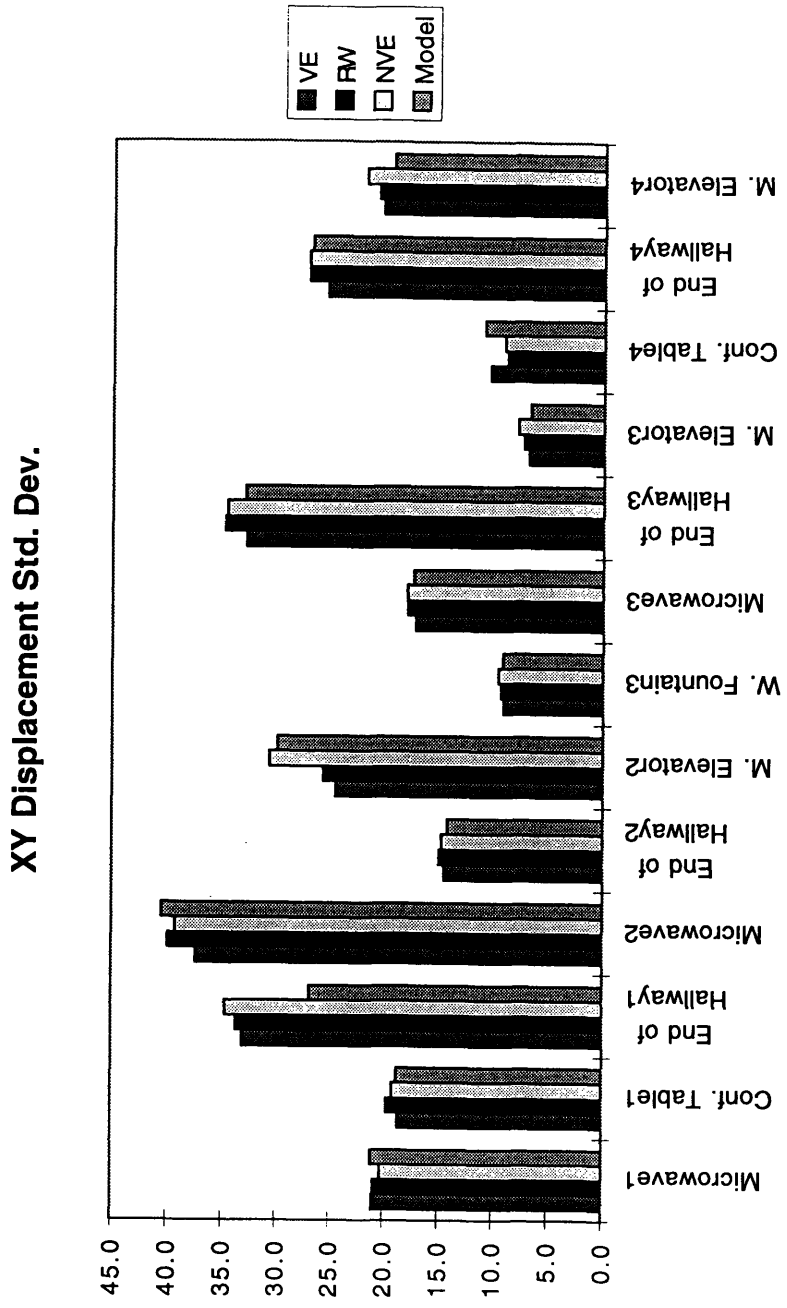


FIGURE 23. Standard Deviation of Error Vectors averaged over subjects in each Location/ Target pair and Experimental Conditions.

Results for the ANOVA on the mean bearing error at a confidence level of .95 resulted in the following confidence intervals:

TABLE 14. ANOVA for Mean Displacement Error

Condition	Mean Error	Confidence Interval
VE	33.14	4.77
RW	40.29	4.64
NVE	27.48	5.69
Model	25.49	5.15

Figure 16 plots the mean bearing estimation error for each condition along with their associated confidence interval.

TABLE 15. ANOVA Summary Table for Mean Displacement Error

Source	Degrees of Freedom	Sum of Squares	Mean Square	Obtained F	Criterion F
Between	4	15.057.36	3764.34	5.25	2.37
Within	434	311124.59	716.88		

Note that the Obtained F value of 5.25 is considerably larger than the Criterion F value of 2.37, indicating that we should reject the null hypothesis that there is no correlation between the training conditions with regards to the distance estimation task.

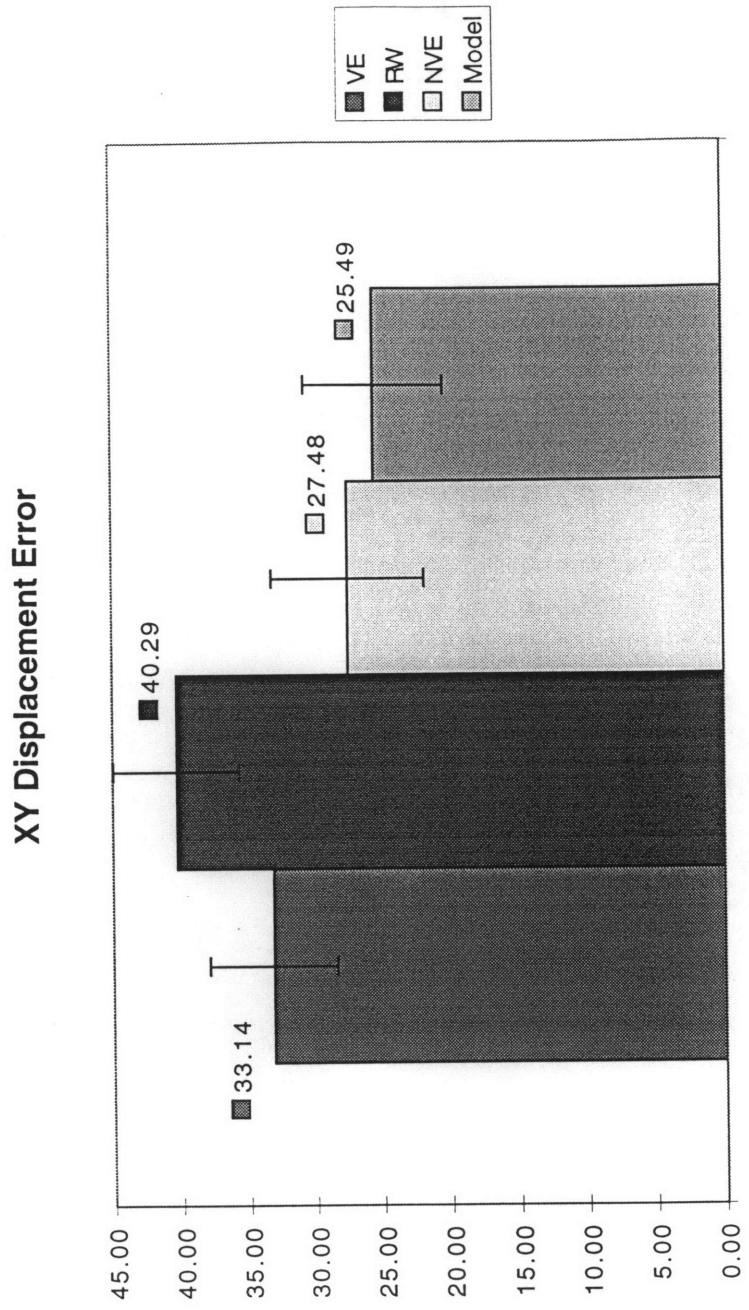


FIGURE 24. Mean Displacement Error for all Location/Target pairs. The Error Bars indicate the confidence interval as derived from the ANOVA test on the data set.

3.7 Discussion

Subjects which gave bearing estimates of more than 90 degrees offset from the actual target were considered to be disoriented. Two subjects exhibited disorientation in the testing phase of the experiment. One of these subjects was in the Model condition, and was disoriented during every pointing task. The responses of this subject were not included in the following analyses. A second subject in the Virtual Environment condition was disoriented in one pointing location (location 4). The responses of this subject to the pointing and distance estimation task from location 4 were discounted, but the rest of his results were used in the analyses. There was no metric for disqualification due to any lack of ability of the subjects to estimate distance to a subject.

At the first pointing location, subjects were required to point to an object in their line of sight. This bearing was used to calibrate the subsequent results in order to minimize for inaccuracies due to the calibration of the pointing device.

With an Obtained F of 1.22 and a Criterion F of 2.37, the ANOVA for bearing error in **3.6.1** indicates that there is not enough evidence to conclude that any of the training conditions have an effect on the performance of the bearing test. This does not indicate that there are no differences in the training methods with regards to the bearing test, just that the variance between the different training conditions is not sufficiently larger than the variance within the training conditions.

ANOVA analysis for mean distance error in **3.6.2** results in an Obtained F of 7.02 and a Criterion F of 2.37 suggesting that the null hypothesis is false and that there is some correlation between the training conditions and performance of the distance estimation task. With regards to the distance estimation task, the Real World training condition

fared particularly poorly, with the non-immersive Virtual Environment and Model training conditions fairing very well.

The cartesian localization analysis of the pointing and distance estimation tasks outlined in **3.6.3** is the metric which probably comes closest to the idealized spatial awareness test - how well the subjects gained knowledge of the spatial locations of the target landmarks. Analysis of variance results in an Obtained F of 5.25 and a Criterion F of 2.37, again indicating positive correlation between the training method and test performance.

That the Model condition gave the best results is perhaps not surprising. Certainly it has been noted that maps can be a very good method of training spatial knowledge, perhaps giving even better results than training in an immersive virtual environment (Satalich 1995). The Model training condition provides a WIM (Stoakley, Conway, Pausch, 1995) representation of the experimental venue, which has many advantages over a map. One of these advantages is that the training Model is the exact model used in the Virtual Environment (VE) and Non-Immersive Virtual Environment (NVE) walkthroughs. Subjects training in the Model condition had access to the same photorealistic rendering of the virtual venue as did the egocentric virtual environment groups but from an alternate and differently manipulable point of view.

That is surprising is the relative ineffectiveness of the Real World training condition when compared to the VE and NVE conditions. There are several plausible explanations of this. While the computer training conditions were modeled with photorealistic textures, it was impossible include *all* possible detail into the computer model. The RW subjects were thus faced with a more cluttered version of the experimental venue than the computer trained subjects. As this experiment was focusing upon spatial knowledge of the venue and not a test of memory, the target objects were chosen to be major landmarks. The RW subjects may have been trying to

gain a higher resolution sense of the venue than was relevant to the experiment, focusing on smaller details but losing sense of the overall spatial configuration of the venue.

Another plausible explanation for the subpar performance of the Real World subjects is lack of interest during the training period. Subjects in the computer training conditions were likely to have been more absorbed in the training due to the novelty of the virtual model. Perhaps the subjects in the RW condition were less persistent in their exploration and surveying of the venue than their colleagues in the computer conditions.

The VE and NVE walkthroughs were also speed limited; subjects controlled their speed using a joystick up to a maximum amount. Although subjects in the NVE and VE conditions expressed a desire to move faster, this speed limitation may have been advantageous in the subjects' spatial knowledge acquisition of the venue. The relatively slow rate of travel may have allowed the subjects to more closely observe the venue when travelling from point to point. As well, because the rate was low and limited, subjects tended to move at the maximum limited speed. Moving at a constant rate may have enabled the subjects to acquire a greater sense of distance when moving around in the virtual condition as compared to the subjects in the Real World condition whose movement rate was not constant.

The Non-Immersive Virtual Environment condition also proved to be a very successful training system -- ranking closer to the Model condition than the Immersive Virtual Environment and Real World conditions. In comparison to the RW condition, the NVE condition has the possible advantages outlined above. Perhaps the main difference between the non-immersive and the immersive virtual environment conditions is the headtracker device. Because the headmounted display in the VE condition is tracked to the location which the subject is looking at, subjects in the VE

condition were more able to survey a particular location without having to control rotation using the joystick. Nonetheless, the head mounted display is a relatively unnatural piece of equipment to use. The subjects tended to be very familiar with the use of a computer monitor and even navigation in similar non-immersive environments that are so prevalent in first person computer games.

Advantages of the NVE over the VE condition include higher resolution, refresh rate, and color. The NVE was presented on a monitor which ran at 640x480 resolution whereas the HMD had a 350x230 resolution. Color reproduction was also superior on the monitor used to implement the NVE condition. While it possible to limit the resolution, refresh rate, and color depth on the HMD, doing so was deemed to be counterproductive. To do so would also suggest that we, for example, limit the field of vision of the subjects in the RW training condition. A major component of the study is to determine the efficacy of various forms of training. One way to measure this is the cost effectiveness of training using the different computer conditions. It would seem to be counter to our goals to attempt to control for visual fidelity differences in HMD and computer monitor, as such differences are key in real world applications of the technology.

The primary purpose of this study was to determine if adequate knowledge of a venue can be obtained through training in a virtual environment as opposed to a real world training situation. The results of this study have shown that this is indeed the case - the computer training groups performed as well or better than the real world conditions by any measure.

The comparison of the various forms of computer training were less conclusive. The exocentric model training condition performed well, as did the non-immersive Virtual Environment condition. Analysis on the variance of the subjects' test results show a high variance in the RW and VE condition and a relatively lower variance and degree

of error in the NVE and Model conditions -- even though the same computer model was in use. This suggests that there are possibly independent advantageous factors in both the NVE and Model training conditions which may be combined to develop an even more effective computer based virtual environment training tool.

While the intersubject variance was large, it was shown in this study that a virtual environment training system can be as effective a real world experience of the same space with regards to the spatial localization of major landmarks. With this baseline established and the successful development of a runtime virtual walkthrough system, it is possible to forge ahead and focus more specifically on developing and testing theories and techniques and determining their effects on spatial knowledge acquisition ability.

4.0 Further Research

Several ideas have been postulated to account for the effectiveness of the NVE and Model training conditions over the Real World and VE, but these hypotheses must be tested in order to analyze the factors which are relevant in a comparison of the effectiveness of computer training methods.

4.1 Factors in Model Generation

A major topic of research is to determine the relationship between the efficacy of the immersive and non-immersive virtual environments. The superior performance of the experimental subjects in the NVE training conditions brings up interesting questions with regard to the physical and psychological stress factors which inherent in developing novel training methods. Pausch, Proffitt, and Williams (1997) utilized the same display for both NVE and VE training conditions eliminating many of the variables associated with the fidelity of the hardware system.

Research issues which indubitably arise in the generation of virtual environments for training purposes are of fidelity versus ease of development. While we note that the fidelity of the NVE system is greater than that of the VE system (even though the same model and textures were used) due to hardware limitations, it is unclear that it is the fidelity which is the root of the under-performance of the VE subjects. A series of experiments fixing the display apparatus and modifying the simulation for texture detail, viewport size, frame rate, perspective, color, and other graphical and hardware issues would serve to resolve the issues surrounding the factors which are predominant in the learning task and also aid in the determination of engineering trade-offs in virtual environment model generation.

4.2 Work and Distance Estimation

While virtual environments walkthroughs are generally deemed to be deficient because of the lack of *work* required to move from one location to another, it was interesting to note that in this study, the Real World control group fared worse than the two egocentric Virtual Environment walkthrough groups in both bearing and distance estimations. A possible issue is the rate of movement in the virtual walkthroughs. Because of the limited maximum rate of locomotion in the virtual walkthroughs, subjects were in effect constrained to exploring the environment at a constant rate.

Future experiments analyzing the variance in distance estimations in training situations which fixed and varying rates of locomotion within the venue would be interesting. Perhaps the *time* taken to traverse a certain distance (instead of the *work* expended to do so) was used by the subjects in the virtual environment training conditions to estimate the distance travelled. Another explanation could be that the relatively low rate of travel *forces* the subjects to more closely observe the simulation. An interesting study would compare several fixed rates of locomotion versus variable rates of locomotion to determine the relationship between rate of traversal and spatial knowledge acquisition.

4.3 Extensions of the Virtual Environment System

Experiments on training using virtual environments include utilizing different navigation metaphors, enhancements of the virtual environment experience such as the inclusion of sound, and modifications of the immersive walkthrough in order to augment the learning experience. The augmentations may include the ability to see through walls on demand, a dynamically updated overhead map of the venue, or the ability to change viewpoints or fly above the venue on demand.

Future experiments on using virtual environments as a training tool will center around searching for unique ways of utilizing a computer training system to reinforce the learning experience as opposed to attempting to emulate traditional training tools.

4.4 Venue Complexity

A more complicated venue could be introduced to further differentiate the efficacy of different training methods and to introduce 3-dimensional factors into the learning task. Four floors of building NW30 have been architecturally modeled to serve as a more topologically complex venue for subsequent experiments.

The NW30 building comprises of approximately 89,695 square feet of space. The accessible paths on the selected floors are highlighted on following figure. The number of loops and decision points are also listed. Loops are conduits which connect to themselves -- if a subject follows a loop, she will eventually find herself at her original point. Decision points are defined as branches in a conduit with two or more possible choices of direction. The complexity of a given venue is indicated by the number of decision points and loops in that venue.

Note that there are a number of stairwells in the venue, and each stairwell is also a decision point. The effective complexity of the building is thus greatly increased because of the many loops which exist once the stairwells and 3D paths are taken into consideration. Complexity can be arbitrarily increased by selecting paths with multi-story waypoints. Conversely, it is also possible to use the virtual model for a less complex navigational study simply by limiting the subject to one floor.

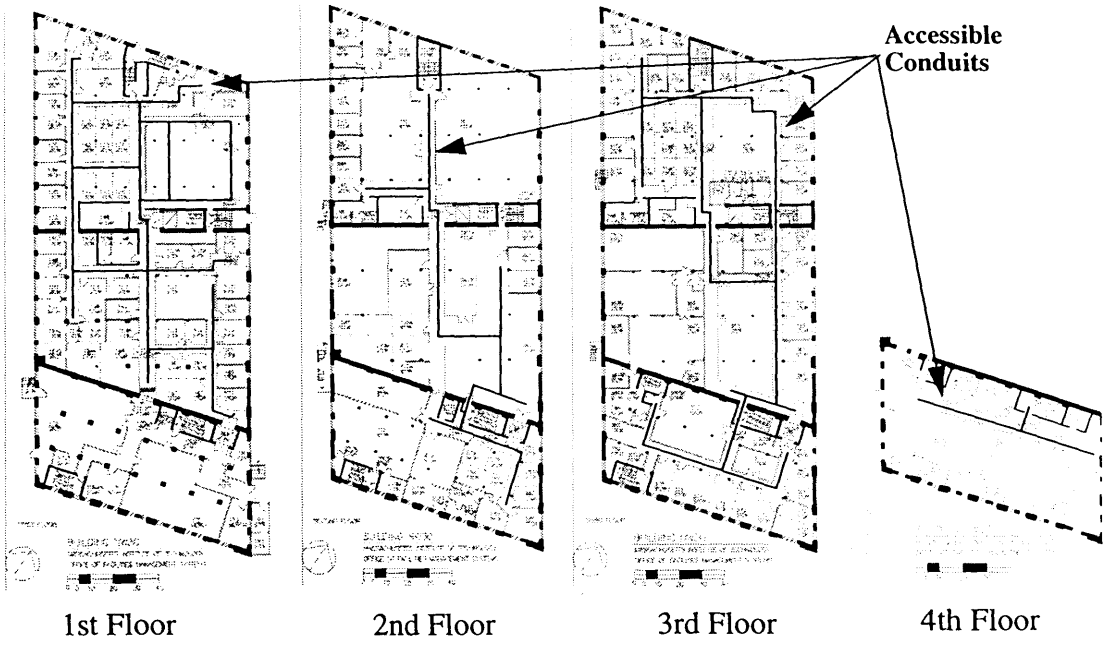


FIGURE 25. Building NW30 Model

4.5 Wayfaring (Route Finding)

Accuracy in the pointing task and in distance estimation was used in this study as the determinant of spatial knowledge acquisition. Wayfinding can also be used as a measure of a subject's ability to recognize a space. Subjects can be placed in the venue and asked to find the quickest or most efficient route to a given target. Obstacles may be placed in the optimal route to force the subject to find a new path to the target.

The development of the NW30 model will allow for more complex wayfaring tasks with several different checkpoints in between starting location and the target. The number of wrong turns, distance travelled, and time taken can be used as measures of efficiency in this task.

5.0 Appendix

5.1 Runtime Simulator Modules

The primary files for running experimental trials are expt1_hmd.bat, expt1_nohmd.bat, practice_hmd.bat, and practice_nohmd.bat. These script files automatically set the on-screen resolution and output region of the screen to the headmounted display and start up the correct runtime version of the walkthrough binary.

The *_hmd.bat files use a compiled EasyScene binary which activates the Polhemus headtracker and uses coordinates received from the tracker to determine the region of the virtual model to be out put to the headmounted display.

The *_nohmd.bat files use a slightly different version of the EasyScene binary which disables the use of the headmounted display for use in non-immersive conditions. Disabling the headtracker allows for more comfortable navigation in the non-immersive conditions because of the existence of noise in the HMD tracking which is amplified when coupled with the monitor in the non-immersive condition.

Directory containing .dwb files: ~/EXPT1/*

498284 Apr 29 00:01 Model.dwb **Building 36-7 Model for walkthrough.**

478636 May 3 21:42 Model_NoTop.dwb **Building 36-7 Model for VE Model.**

3816 Apr 28 18:46 car2.dwb **Model of collision detection bounding object.**

1960576 Apr 28 19:53 escene_hmd **Compiled ES code with HMD interface.**

1960576 Apr 28 17:53 escene_nohmd **Compiled ES code without HMD interface (for non-immersive VE walkthrough).**

2822 May 8 02:12 expt1.set **ES .set file for the experimental trial.**

35 Apr 28 20:05 expt1_hmd.bat **Batch file to run immersive VE.**

56 Apr 28 20:05 expt1_nohmd.bat **Batch file to run non-immersive VE.**

40 Apr 28 17:55 lowres **Sets terminal to 640x480 for non-immersive VE.**

4294 May 8 04:10 practice.set **ES .set file for practice sessions.**

38 Apr 28 20:06 practice_hmd.bat **Batch file to run immersive practice session.**

59 Apr 28 20:06 practice_nohmd.bat **Batch file to run non-immersive practice session.**

42 Apr 28 17:55 stereo **Sets headmounted display to stereo mode.**

1502 May 7 17:42 topview.set **ES .set file for virtual model mode.**

42 Apr 28 17:55 unstereo **Sets headmounted display to mono mode and sets full screen resolution on terminal.**

5.2 Sample EasyScene Script (.set) File

The .set files are used by EasyScene to configure the GUI to produce the correct simulation. These files are simple script files which are relatively easy to modify and can be used to load various models into the GUI, change the size of the viewport and the field of view, change the camera angles, and modify the input controls.

The .set files can also be used to pass parameters back to any C code which is compiled into the runtime escene binary. Examples of this are the qdev "UPARROWKEY" bindings which call the function "up" in collide.c. Finally, this .set file also sets the collision detection vectors. Further explanation of the .set file and programming EasyScene can be found in the Coryphaeus text book and sample source code of which the following file and the collide.c file is based.

```
# Search path to look for models
path "../MODELS:";
# Set up the process model
process default;
# Open a window and let the user position it on the screen
pipe "Window #1" add on screen 0,
viewport 0,640,0,480;
# key bindings
qdev "UPARROWKEY" function "up", autorepeat;
qdev "DOWNARROWKEY" function "down", autorepeat;
qdev "RIGHTARROWKEY" function "right", autorepeat;
qdev "LEFTARROWKEY" function "left", autorepeat;
# hold down left shift key to move faster
qdev "LEFTSHIFTKEY" function "shift", autorepeat;
# hold down left alt key to move sideways
qdev "LEFTALTKEY" function "alt", autorepeat;
configure;
```

```
# load the person model and place it at coordinates -14.6, 1.4, 11.7;
rotate it to angle 330, 0, 0.

object "person" load "car2.dwb", translate -14.6, 1.4, 11.7, rotate
330,0,0;

# create a channel, set the different parameters and then attach it to
the person object

channel "front view" add on pipe "Window #1",

viewport 0.0 1.0 0.0 1.0,

attach "person",

clip      0.1, 20000.0,

viewpos   0.0, 0.0, 0.0,

viewrot   0.0, 0.0, 0.0

fov       55.0, -1.0;

# Sample code to add a window with a channel "top view" add on pipe
"Window #1"; gives a top-down view of the model.  Useful for debugging
collision detection vectors.

#  viewport 0.0 1 0 1,

#  attach "person",

#  clip 0.1, 20000,

# Notice that the camera is rotate -90 degrees to look "downward".

# viewrot 0,-90,0

# Place the camera 10 units above the model so it looks like a bird's
eye view.

#  viewpos 0,10,0

# fov 100, -1;

# load the other objects of interest

object "Model" load "Model.dwb";

# initialize the person so that the car collides at the 'PRIMitive'
level against all objects with a mask of 0x000f and the function to
call when there's a collision is 'carCollideSimple'.

# Also set up line segments around # Also set up line segments around
the car object and then enable the collision so that the nearest
object for each line segment is returned.


object "person" mask 0xf000;
```

```
object "Model" mask 0x000f;
object "person"
  initIsect PRIM mask 0x000f function "personCollideComplex",
  addSeg pos 0.0 0.5 0.0 dir 0.0 -1.0 0.0 len .50 RAW,     # down
  addSeg pos 0.0 0.0 0.0 dir 0.0 0.0 -1.0 len .5 COOKED, # front
  addSeg pos 0.0 0.0 0.0 dir 0.0 0.0  1.0 len .5 COOKED, # back
  addSeg pos 0.0 0.0 0.0 dir  1.0 0.0 0.0 len .1 COOKED, # right
  addSeg pos 0.0 0.0 0.0 dir -1.0 0.0 0.0 len .1 COOKED, # left
enable NEAR 1;
userCallback "setCollideTolerance" argument "1";
userCallback "setHeightTolerance" argument ".1";
userCallback "calibrateTracker" argument "x";
userCallback "calibrateJoystick" argument "x";
userCallback "setJoystickMode" argument "2";
# add a few modules [in collide.c] that control the car's movement
module "movement module" add
  postinit "initPerson"
  preapp "updatePersonComplex";
```

5.3 Simulator Module Development

This is the directory for the EasyScene walkthrough development. The included Makefile compiles all listed modules into a runtime EasyScene binary.

Directory containing .c files: ~/project/new/*

842 Apr 16 18:15 fastrak.c **Polhemus Fastrack code.**

825 Apr 16 18:15 joy.c **Joystick code.**

386 Apr 16 18:15 joy.h **Joystick code.**

288 Apr 16 18:15 serial.h **Serial device code.**

752 Apr 16 18:15 serial.c **Serial device code.**

6744 Apr 28 17:11 Makefile **Makefile to compile into escene binaries.**

12072 Apr 28 19:50 collide.c **Main collision detection, movement, HMD code.**

5.4 Control and Collision Detection Sourcecode

The following file is a modification of the collide.c file provided as an example source code for implementing collision detection using Coryphaeus' EasyScene. Modifications were made to the source in order to support the Polhemus headtracker and a serial joystick. The new EasyScene binary can be compiled by typing "make" in the directory with the object files. The included Makefile includes and links all the necessary EasyScene libraries.

```
/*
 *        Copyright (c) Coryphaeus Software 1995
 *
 * Permission to use, copy, modify, distribute, and sell this software
 * and its documentation for any purpose is hereby granted without
 * fee, provided that (i) the above copyright notices and this
 * permission notice appear in all copies of the software and related
 * documentation, and (ii) the name of Coryphaeus Software may not be
 * used in any advertising or publicity relating to the software
 * without the specific, prior written permission of Coryphaeus
Software.
 *
 * THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND,
 * EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY
 * WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
 *
 * IN NO EVENT SHALL CORYPHAEUS SOFTWARE BE LIABLE FOR ANY SPECIAL,
 * INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY
 * DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,
 * WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY
 * THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE
 * OR PERFORMANCE OF THIS SOFTWARE.
 *
```

```
 */
/* Modified by Glenn Koh so it actually works */
/* Modified by Maciej Stachowiak for use with Polhemus Fastrak tracker
*/

/* $Id$ */

#include <bstring.h>
#include <stdlib.h>
#include <Performer/pf.h>
#include "_es.h"
#include <esIsect.h>
#include <math.h>
#include <unistd.h>
#include "joy.h"
#include "ft.h"

#define DOWN 0
#define FRONT 1
#define BACK 2
#define RIGHT 3
#define LEFT 4
#define SPEED 0.2f
#define HISPEED shiftKey*0.4f

  /*
   * prototypes
   */
int testCollision (pfVec3, pfVec3);
void slideWithWall (pfVec3 pos, pfVec3 normal, pfVec3 ipoint, int
negate);
void jdisp( struct joy_packet);
  /*
```

```
   * Handle to the person
   */

esObject *person = NULL;

  /*
   * Collision detection enable flag
   * The other values are flags set by the module that will be called
   * whenever there's a collision
   */

static int enableFlag = 1;
static int collidedFlag = 1;
static int js_mode=-1;
static pfVec3 v[10], n[10]; /* collision points and normals */
static int flag[10];        /* true if the segment 'i' collided */
static int half[10];
static float height;        /* 1st segment is a down segment */

  /*
   * The distance(s) at which we actually collide with an object or
the
   * ground
   */

static float TOLERANCE = 0.6f;
static float HTOLERANCE = 0.1f;

  /*
   * Key bindings that are used to actually drive the person.  Similar
   * bindings may be used to bind runtime functions to the ES .set
files.
   */
```

```
int upKey=0, downKey=0, rightKey=0, leftKey=0, shiftKey=0, altKey =
0;
void up (void) { upKey = 1; }
void down (void) { downKey = 1; }
void right (void) { rightKey = 1; }
void left (void) { leftKey = 1; }
void alt (void) { altKey = 1; }
void shift (void) { shiftKey = 1; }

void enableCollision (void)
{
  enableFlag = ! enableFlag;
  fprintf (stderr, "Collision %s\n",
(enableFlag)?"Enabled":"Disabled");
}

void setJoystickMode (char *arg)
{
  js_mode = atoi (arg);
  fprintf (stderr,%

/*
 * These are user Callbacks called from the script. Easy way to change
 * the collision parameters
 */

void setCollideTolerance (char *arg)
{
  TOLERANCE = atof (arg);
  fprintf (stderr, "Setting tolerance to %.4f\n", TOLERANCE);
}
```

```
void setHeightTolerance (char *arg)
{
  HTOLERANCE = atof (arg);
  fprintf (stderr, "Setting Height tolerance to %.4f\n", HTOLERANCE);
}

static struct ft_unit *u;
static struct ft_packet pkt;
struct ft_packet base_pkt;
static int js_toggle=0;
struct joy_packet jpkt;
struct joy_unit *ju;

/* Gets the initial packet from the Polhemus tracker */
void calibrateTracker (char *arg)
{
  u = ft_open("/dev/ttym3");
  ft_getpkt(u,&base_pkt);
}

void calibrateJoystick (char *arg)
{
  ju=joy_open("/dev/ttym1");
}

/*
 * Find out if the person's been loaded.
 * If it has, then make sure we can see the collision segments if any.
 * We can turn off collision segments by toggling when we're done with
 * debugging.
 * /
```

```
void initPerson (void)
{
  person = esFindObject ("person", global->scenes->firstObject);
  if (! person)
  {
    fprintf (stderr, "Couldn't find person\n");
    global->running = 0;
    return;
  }
  esToggleDrawIsectObjectSegs (person);
}

/*
 * Convenience macros to go from EasyScene to Performer coordinates
 * NOTE:
 *  These change the values in place
 */

#define PFTOES(_x) \
  do { \
  pfVec3 tmp; \
  \
    PFCOPY_VEC3 (tmp, _x); \
    _x[0] = tmp[0]; \
    _x[1] = tmp[2]; \
    _x[2] = -tmp[1]; \
  } while (0);

#define ESTOPF(_x) \
  do { \
  pfVec3 tmp; \
  \
```

```
    PFCOPY_VEC3 (tmp, _x); \
    _x[0] = tmp[0]; \
    _x[1] = -tmp[2]; \
    _x[2] = tmp[1]; \
  } while (0);

/*
 * get a fastrak packet and set headings.  Note that the Polhemus
 * packets are subtracted from the intializing packet in order to
 * calibrate the tracker.
 */

  ft_getpkt(u,&pkt);
  person->h=pkt.az-base_pkt.az;
  person->r=-pkt.rl-base_pkt.rl+10;
  person->p=-pkt.el-base_pkt.el;

/* get a joystick packet if the joystick mode is non-negative */
  if (js_mode >0) {
    if (!js_toggle) {
      joy_getpkt(ju,&jpkt);
      jdisp(jpkt);
      js_toggle = 1;
    } else {
      js_toggle = 0;
    }
  }

/* Checks for keyboard input */
  if (rightKey)
  {
    if (altKey)
```

```
    {
    float s,c;

      pfSinCos (person->h-90.0f, &s, &c);
      person->x -= (SPEED+HISPEED)*s;
      person->z -= (SPEED+HISPEED)*c;
      altKey = 0;
    } else {
        base_pkt.az += (2.5f + shiftKey*1.5f);
person->h -= (2.5f + shiftKey*1.5f);
    }
    rightKey = 0;
  }

  if (leftKey) {
      if (altKey) {
          float s,c;
  pfSinCos (person->h+90.0f, &s, &c);
  person->x -= (SPEED+HISPEED)*s;
  person->z -= (SPEED+HISPEED)*c;
  altKey = 0;
      } else {
          base_pkt.az -= (2.5f + shiftKey*1.5f);
          person->h += (2.5f + shiftKey*1.5f);
      }
      leftKey = 0;
  }
  if (shiftKey) shiftKey = 0;
}

/*
 * Complex collision detection module. Sets up a collidedFlag so that
```

```
 * the complementary module "updatePersonComplex" knows that there's
 * been a collision and then reacts accordingly.
*/

void personCollideComplex (esObjectIsectInfo *oi)
{
register int i;
pfVec3 oldPos, newPos;
float d;
collidedFlag = 1;
PFSET_VEC3 (oldPos, person->x, person->y, person->z);
ESTOPF (oldPos);

/*
 * Initialize all the collision flags to zero
 */
for (i=0; i<10; ++i) { flag[i] = 0; half[i] = 0; }
flag[FRONT]=0;
flag[LEFT]=0;
flag[BACK]=0;
flag[RIGHT]=0;

  if (oi && enableFlag)
  {
    for (i=0; i<oi->numSegs; ++i)
    {
    pfVec3 ipoint, normal, o;
    pfSeg seg;

      if (! oi->ii[i].numHitObjects) continue;

      /*
```

```
         * Get the collision point and the normal at that point
         */
        PFCOPY_VEC3 (ipoint, oi->ii[i].si[0].ipoint);
        PFCOPY_VEC3 (normal, oi->ii[i].si[0].normal);
        if (i)
        {
        pfSeg seg;
        pfVec3 nl, vl;

#ifdef SIMPLE_DISTANCE_CALCULATION
/*
 * Simplest case calculate the distance from the current person's
 * position to the intersection point
 */
PFSET_VEC3 (o, person->x, person->y, person->z);
d = PFDISTANCE_PT3 (o, ipoint);
#else
/*
 * Use the dot product of the segment that connects the current
 * position & the collision point and the normal at the collision
 * point to calculate the actual distance from the object
 */
PFCOPY_VEC3 (vl, ipoint); ESTOPF (vl);
        PFCOPY_VEC3 (nl, normal); ESTOPF (nl);
pfNormalizeVec3 (nl);
pfMakePtsSeg (&seg, oldPos, vl);
PFSCALE_VEC3 (seg.dir, seg.length, seg.dir);
d = PFDOT_VEC3 (nl, seg.dir);
d = PF_ABS (d);
#endif
if (d <= TOLERANCE)
{
```

```
  PFCOPY_VEC3 (v[i], ipoint);
  PFCOPY_VEC3 (n[i], normal);
  flag[i] = 1;
  if (d <= TOLERANCE/2.0f) half[i] = 1;
}
      }
      else
      {
flag[0] = 1;
height = ipoint[PF_Y];
      }
    }
  }
}
/*
 * Complex way to use the collision results to bump off objects
 */

void updatePersonComplex (void)
{
  pfVec3 oldPos, newPos;
  PFSET_VEC3 (oldPos, person->x, person->y, person->z);
  ESTOPF (oldPos);
/*  We don't need to check for flag[DOWN] in this model, but this
 *  allows us to have the person walk up stairs.  Person's y coord.
 *  increases by the HTOLERANCE whenever a collision is detected with
 *  the downward pointing vector.
 */
/*   if (flag[DOWN]) person->y = height + HTOLERANCE; */

/* Keyboard movement routines.  Note the extra flag reset commands
 * flag [xxxx] = 0, which seem to be necessary as there are cases
```

```
* where the flags do not reset until another flag is set.
*/
 if (upKey)
 {
   if (flag[FRONT] == 0)
   {
   float s,c;
     pfSinCos (person->h, &s, &c);
     person->x -= (SPEED+HISPEED)*s;
     person->z -= (SPEED+HISPEED)*c;
   }
    flag [FRONT] = 0;
   upKey = 0;
 }

 if (downKey)
 {
   if (flag[BACK] == 0)
   {
   float s,c;
     pfSinCos (person->h, &s, &c);
     person->x += (SPEED+HISPEED)*s;
     person->z += (SPEED+HISPEED)*c;
   }
   flag [BACK]=0;

  downKey = 0;
 }
 if (rightKey)
 {
   if (altKey)
   {
```

```
    float s,c;
      if (flag[RIGHT] == 0)
      {
pfSinCos (person->h-90.0f, &s, &c);
person->x -= (SPEED+HISPEED)*s;
person->z -= (SPEED+HISPEED)*c;
      }
    }
    else
    {
      base_pkt.az += (2.5f + shiftKey*1.5f);
      person->h -= (2.5f + shiftKey*1.5f);

      flag[RIGHT] = 0;
    }
    rightKey = 0;
  }

  if (leftKey)
  {
   if (altKey)
    {
    float s,c;
      if (flag[LEFT] == 0)
      {
pfSinCos (person->h+90.0f, &s, &c);
person->x -= (SPEED+HISPEED)*s;
person->z -= (SPEED+HISPEED)*c;
      }
    }
    else
    {
```

```
      base_pkt.az -= (2.5f + shiftKey*1.5f);
      person->h += (2.5f + shiftKey*1.5f);
      flag[LEFT]=0;
     }
     leftKey = 0;
   }

/* Get a fastrak packet and set headings
 * Basically turns the orientation of the person to the heading
 * received from the HMD.  Headings are subtracted from the base_pkt
 * headings obtained during the calibration of the HMD.
 */

   ft_getpkt(u,&pkt);
   person->h=pkt.az-base_pkt.az;
   person->r=-pkt.rl-base_pkt.rl;
   person->p=-pkt.el-base_pkt.el;

  /* get a joystick packet if the joystick mode is non-negative */
   if (js_mode >0) {
     if (!js_toggle) {
       joy_getpkt(ju,&jpkt);
       jdisp(jpkt);
       js_toggle = 1;
     } else {
       js_toggle = 0;
   }
}

   if (altKey) altKey = 0;
   if (shiftKey) shiftKey = 0;
}
```

```
echo (char *arg)
{
  printf("%s\n",arg);

}


void jdisp(struct joy_packet pkt) {
  float s,c;
  int x,y,z;
  x = (int)(pkt.x)-151;
  y = (int)(pkt.y)-150;
  z = (int)(pkt.z)-7;
  x = abs(x)<4 ? 0: x - copysign(3,x);
  y = abs(y)<4 ? 0 : y - copysign(3,y);
  z = abs(z)<3 ? 0 : z - copysign(2,z);
  printf("x:%d,y:%d\n",x,y);
  switch (js_mode) {

default:

pfSinCos (person->h, &s, &c);

  if (x < -10)   /* if joystick is pointing left */
  {
float s,c;
if (flag[LEFT]==0)
    {
    base_pkt.az -= (2.5f + shiftKey*1.5f);

                person->h += (2.5f + shiftKey*1.5f);
```

```
    }

flag[LEFT]==0;

}  /*end if pointing left */

   if (x > 10)
{ float s,c; if (flag[RIGHT]==0)
{
      base_pkt.az += (2.5f + shiftKey*1.5f);
      person->h -= (2.5f + shiftKey*1.5f);
 }

flag[RIGHT]=0;

    }  /*end if pointing right */

/* The person's translation speed varies in proportion to the value
 * from the joystick.  This can be easily changed to a digital rate
 * setting by following the analogous code in the keyboard control
 * section.
 */
if (y > 4)
{if (flag[FRONT] == 0)
{float s,c;
       pfSinCos (person->h, &s, &c);
       person->x -= (y*.03)*s;
       person->z -= (y*.03)*c;
}
flag[FRONT]=0;
}
```

```
if (y<-4)
  {
    if (flag[BACK] == 0)
    {float s,c;
      pfSinCos (person->h, &s, &c);
      person->x -= (y*.03)*s;
      person->z -= (y*.03)*c;
    }
    flag [BACK]=0;
  }
 flag[FRONT]=0;
 flag[BACK]=0;
 flag[LEFT]=0;
 flag[RIGHT]=0;
 break;
  }
}
/* End of Collide.c */
```

5.5 Polhemus Headtracker Interface Sourcecode

```
#include <malloc.h>
#include <termios.h>
#include <unistd.h>
#include "serial.h"
#include "ft.h"

/*
 * fastrak.c - Routines that handle communication with the Polhemus
 * the fastrak device.
 *
 * $Id: fastrak.c,v 1.3 1997/01/22 21:10:10 mstachow Exp mstachow $
 *
 */

struct ft_unit *ft_open(const char *dev) {
  struct ft_unit *unit;
  unit = malloc(sizeof(struct ft_unit));
  unit->dev = dev;
  unit->fd = serial_init(dev,B38400);
  return unit;
}

void ft_close(struct ft_unit *unit) {
  serial_close(unit->fd);
  free(unit);
}

 void ft_getpkt(struct ft_unit *unit, struct ft_packet *pkt) {
  char pktbuf[48];
  write(unit->fd,"P",1);
  read(unit->fd,pktbuf,47);
```

```
  sscanf(pktbuf,"01 %f %f %f %f %f",&(pkt->x),&(pkt->y),&(pkt->z),
         &(pkt->az),&(pkt->el),&(pkt->rl));
  puts(pktbuf);
}
```

5.6 Joystick Interface Sourcecode

The joystick has a serial interface to the Onyx running at 2400 baud. X,Y,and Z values corresponding to the joystick control (X and Y) and the knob (Z) are each converted into an 8 bit value. joy.c polls the joystick for the combined 24 bit value when called by the collide.c main routine.

Note that the maximum deflection range of the joystick is less than 2^8. The secondary control knob is also not currently in use, so it is possible to increase the joystick data feedback rate by reducing the number of bits assigned to the X and Y data packets as well as elminating the Z data packet if it is not required.

BEGIN joy.h:

```
/*
 * joy.h - Interface to the Joystick.
 *
 * $Id: joy.h,v 1.2 1997/02/04 14:02:33 mstachow Exp $
 */

struct joy_unit {
  const char *dev;
  int fd;
};

struct joy_packet {
  unsigned char x,y,z;
};
```

```
struct joy_unit *joy_open(const char *dev);
void joy_close(struct joy_unit *unit);
void joy_getpkt(struct joy_unit *unit, struct joy_packet *pkt);

#endif
```

END joy.h

BEGIN joy.c

```
 /*
 * joy.c - Interface to the joystick.
 *
 * $Id: joy.c,v 1.3 1997/02/04 14:02:08 mstachow Exp $
 */


struct joy_unit *joy_open(const char *dev) {
  struct joy_unit *unit;
  unit = malloc(sizeof(struct joy_unit));
  unit->dev = dev;
  unit->fd = serial_init(dev,B2400);
  return unit;
}

void joy_close(struct joy_unit* unit) {
  serial_close(unit->fd);
  free(unit);
}

void joy_getpkt(struct joy_unit *unit, struct joy_packet * pkt) {
  unsigned char pktbuf[6];
  puts("In joy_getpkt");
```

```
  write(unit->fd," ",1);
  puts("Wrote");
  read(unit->fd,pktbuf,5);
  puts("Read");
  sscanf(pktbuf,"%c%c%c", &pkt->x, &pkt->y, &pkt->z);

printf("(%d)(%d)(%d)\n",(int)pktbuf[0],(int)pktbuf[1],(int)pktbuf[2]
);
}
```

END joy.c

5.7 Serial Interface Sourcecode

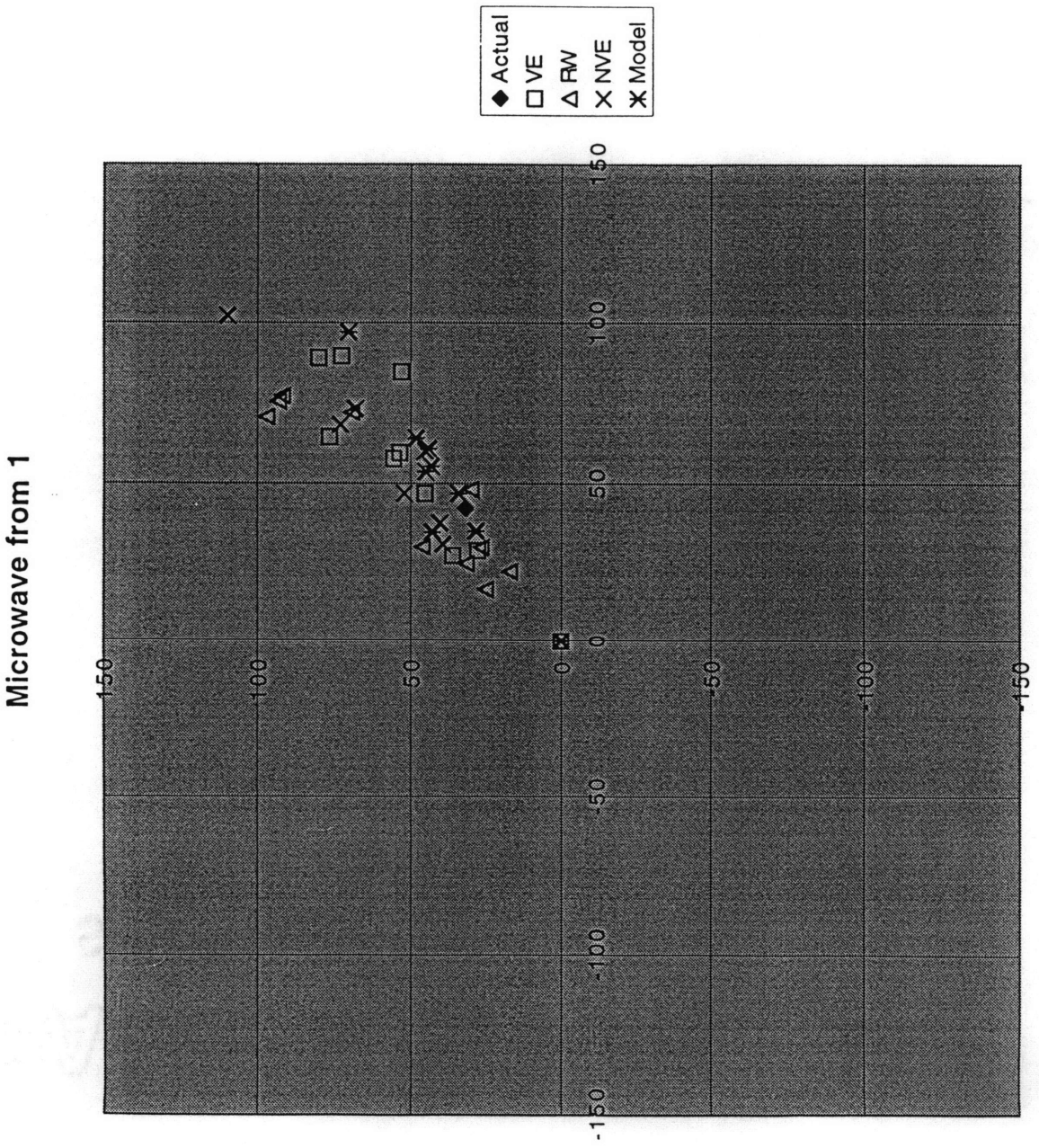
Serial communication routines used to open and monitor the serial port. Allows the interfacing with the Polhemus 3Space Fastrak and the joystick.

```
BEGIN serial.h:
#ifndef SERIAL_H
#define SERIAL_H

/*
 * serial.h - Basic serial communication routines
 * for opening and closing the serial port.
 *
 * $Id: serial.h,v 1.3 1997/01/21 19:08:59 mstachow Exp $
 *
 */


int serial_init(const char *line, int speed);
void serial_close(int ttyfd);

#endif
END serial.h

BEGIN serial.c:
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <termios.h>
#include "serial.h"
```

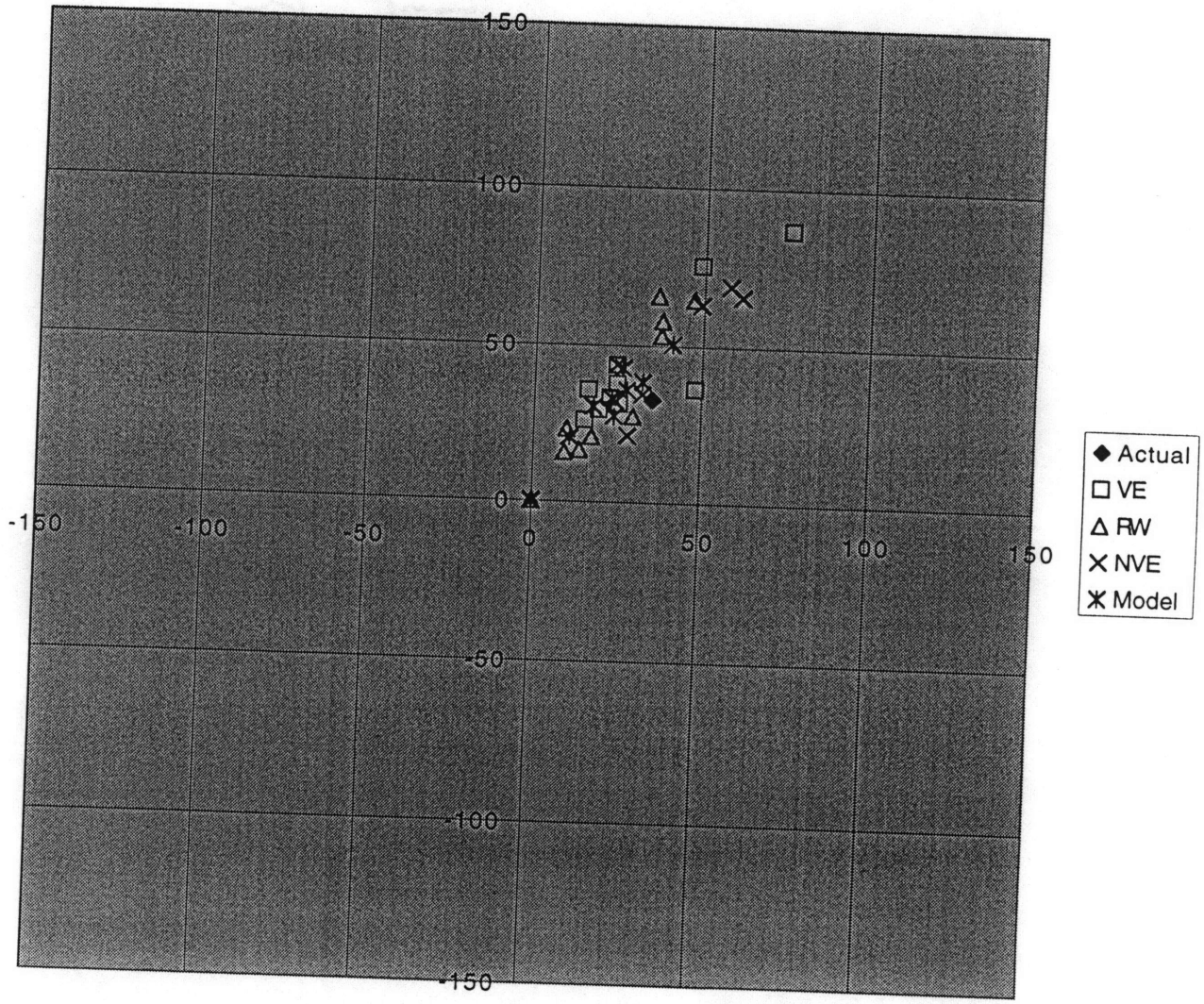
```
/*
 * serial.c - Basic serial communication routines
 * for opening and closing the serial port.
 *
 * $Id: serial.c,v 1.3 1997/01/22 21:08:33 mstachow Exp $
 *
 */


int serial_init(const char *line, int speed) {
  int ttyfd;
  struct termios tty;
  ttyfd=open(line,O_RDWR|O_NDELAY);
  fcntl(ttyfd,F_SETFL,fcntl(ttyfd,F_GETFL) & (~FNDELAY));
  tcgetattr(ttyfd,&tty);
  tty.c_iflag = 0;
  tty.c_oflag = 0;
  tty.c_cflag = speed | CS8 | CREAD | CLOCAL;
  tty.c_lflag &= ICANON;
  tty.c_cc[VTIME]=0;
  tty.c_cc[VMIN]=0;
  tcsetattr(ttyfd,TCSANOW,&tty);
  write(ttyfd,"c",1);
  return(ttyfd);
}

void serial_close(int ttyfd){
  close(ttyfd);
}
End serial.h
```

5.8 XY Localization Data Sets

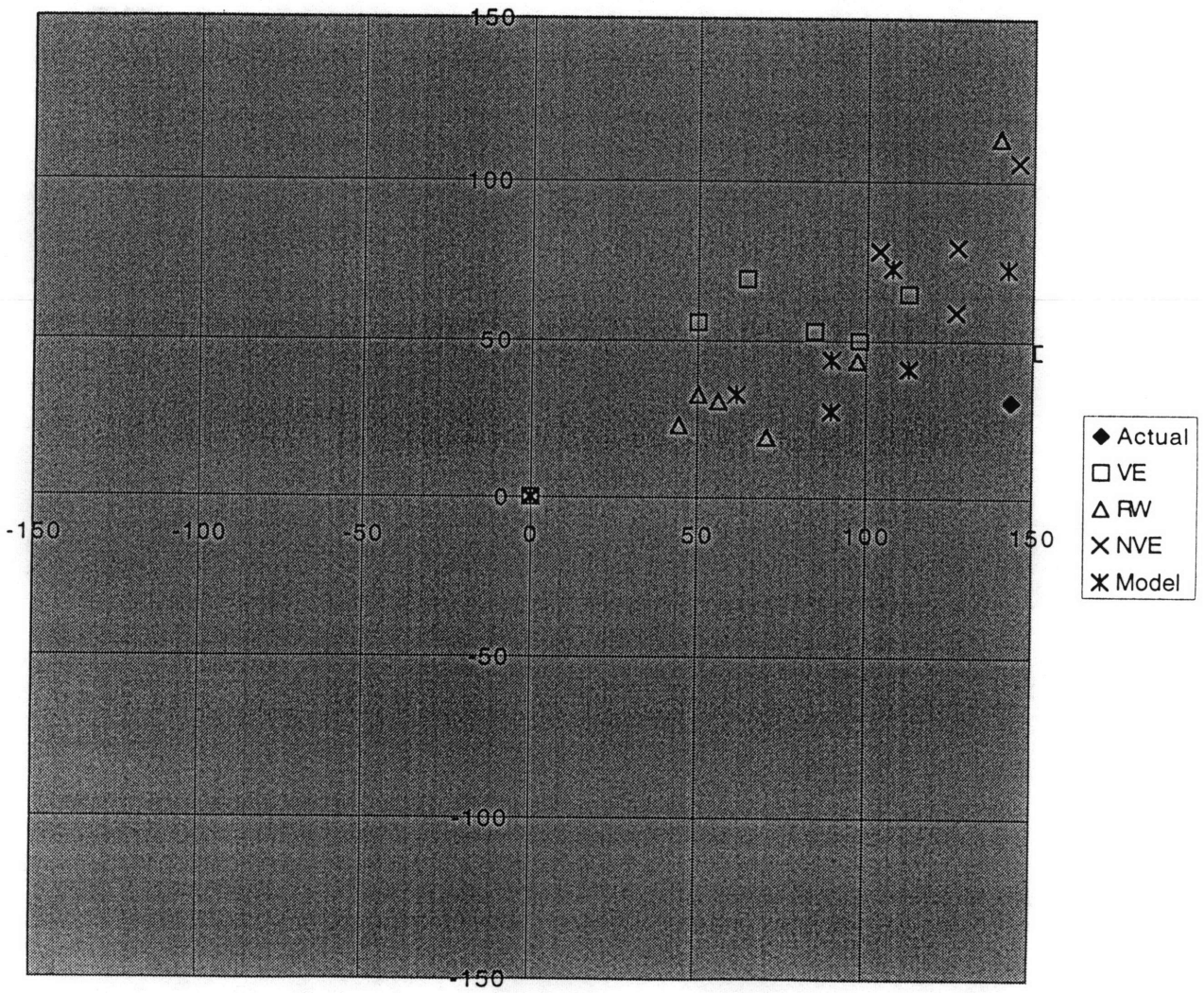
The following plots are calculated based upon subjects' bearing and distance estimations from each of the four pointing locations. Subjects are located at point (0,0).



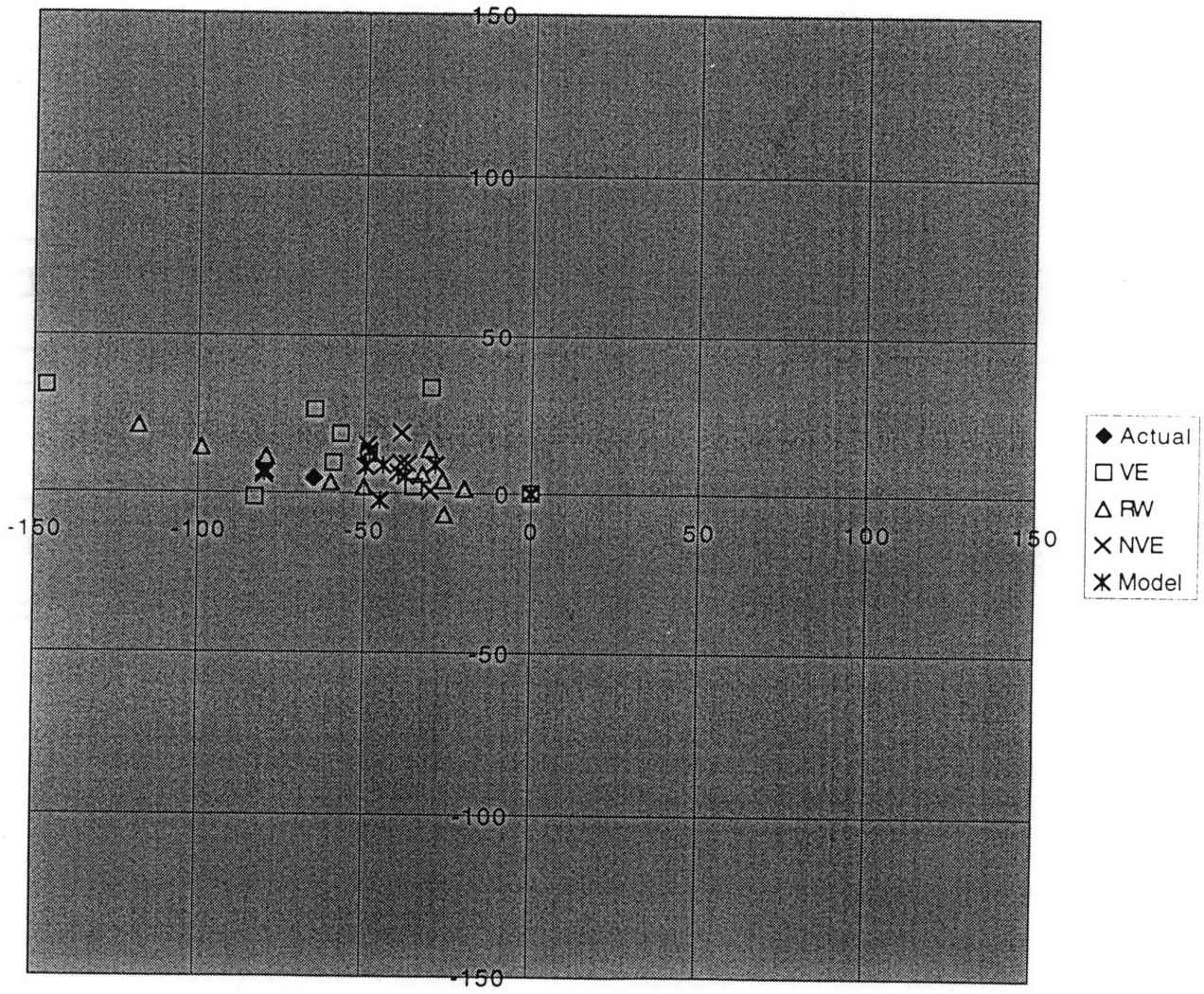
Conf. Table from 1



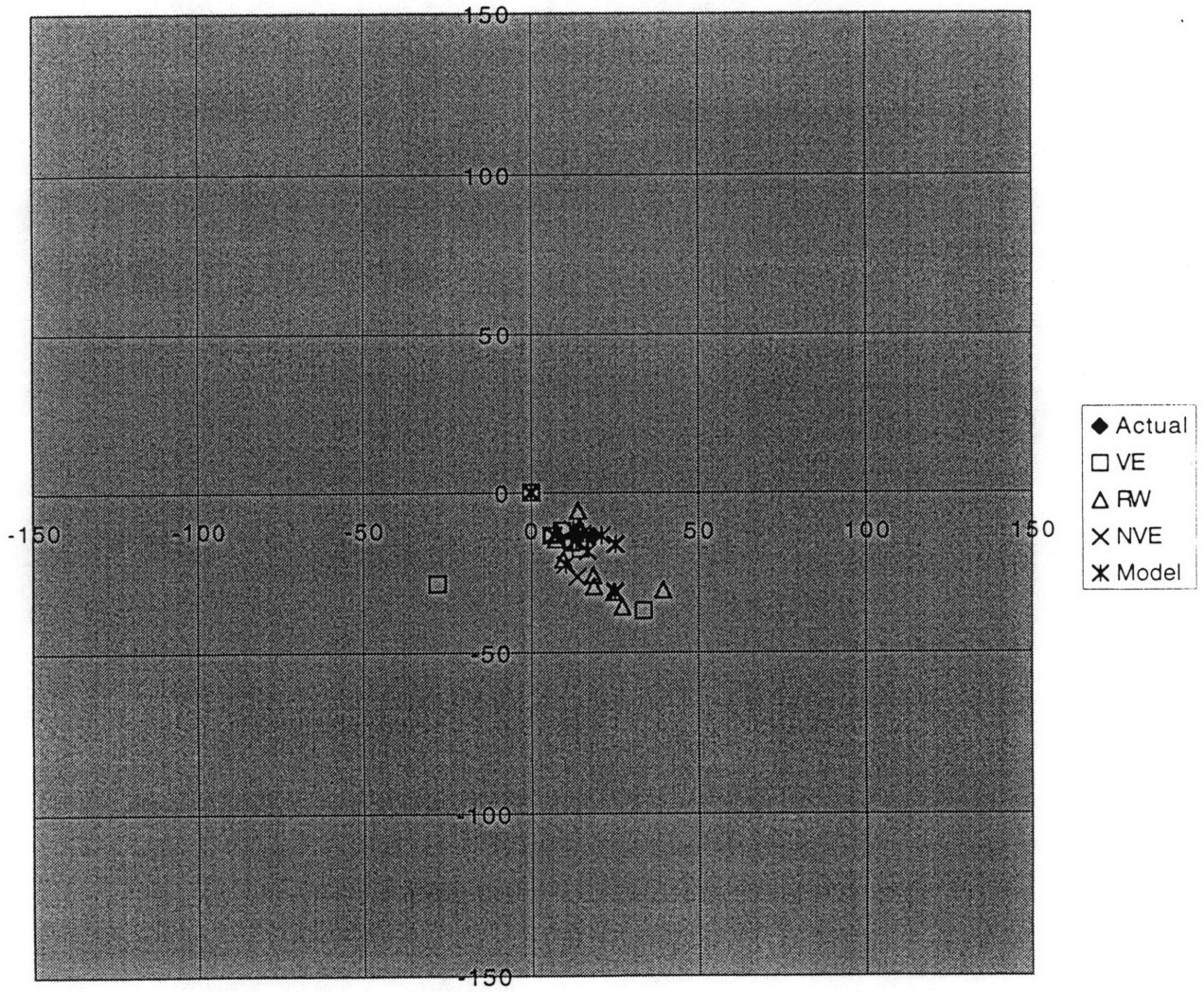
End of Hallway from 1



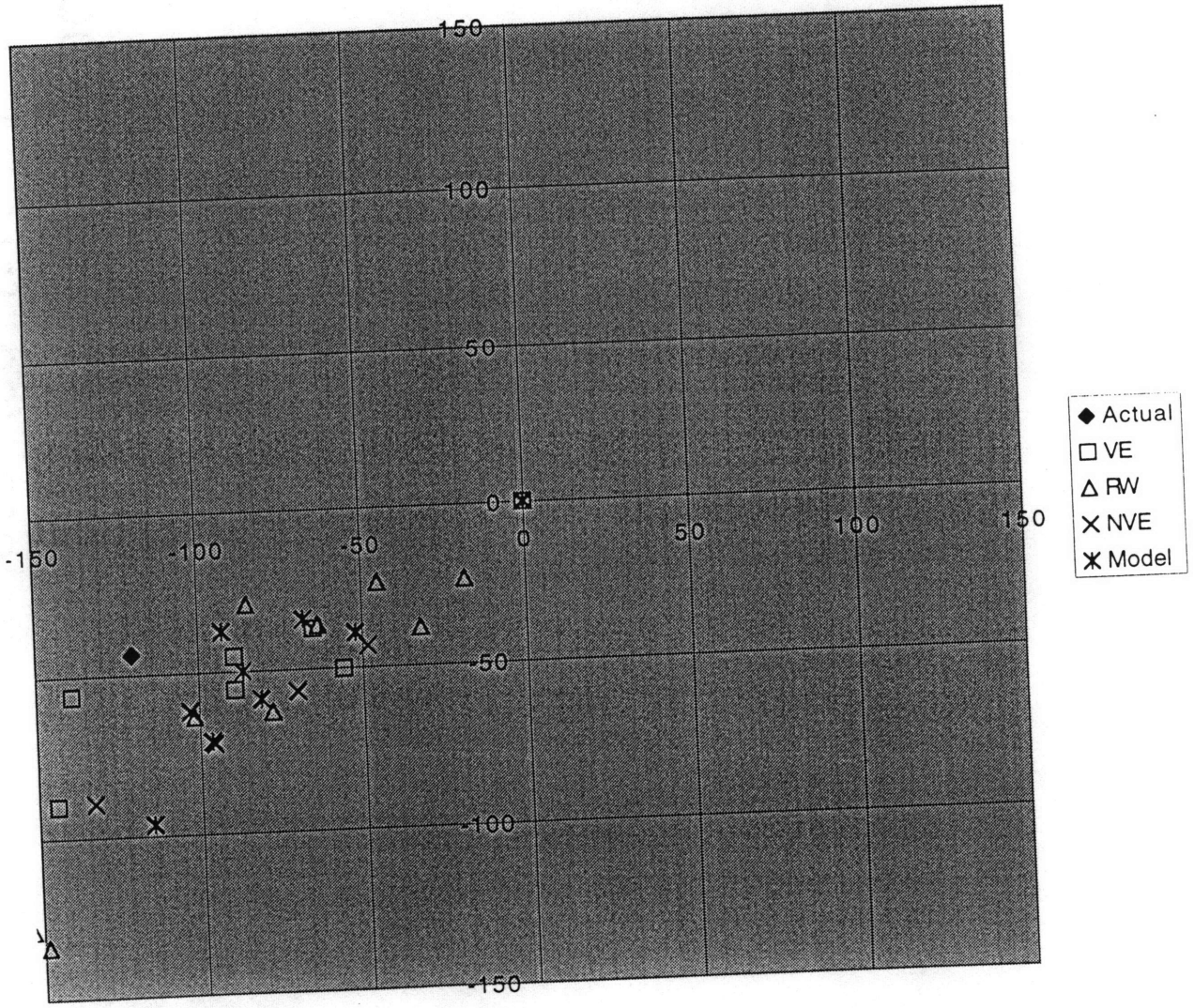
Microwave from 2



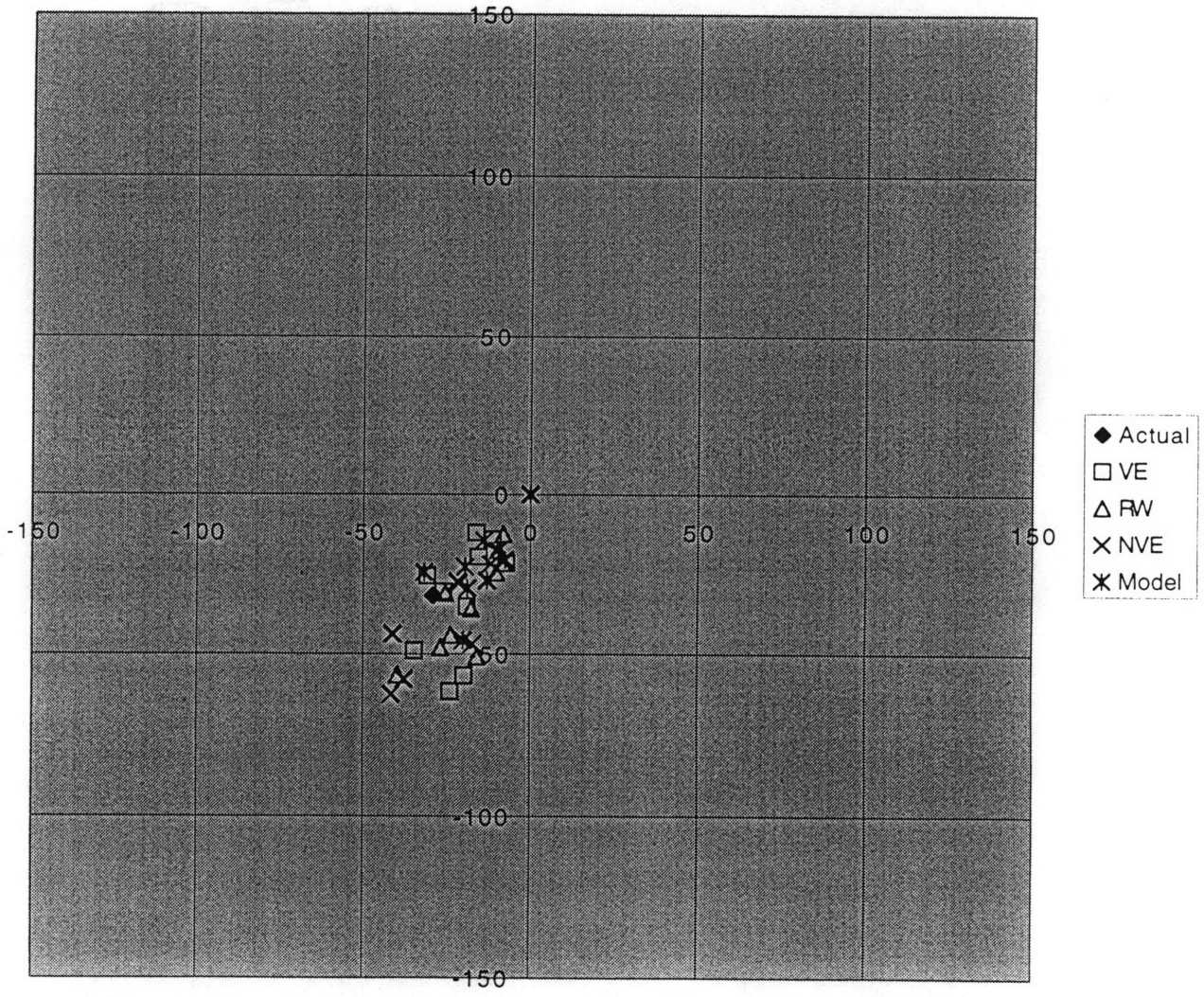
End of Hallway from 2



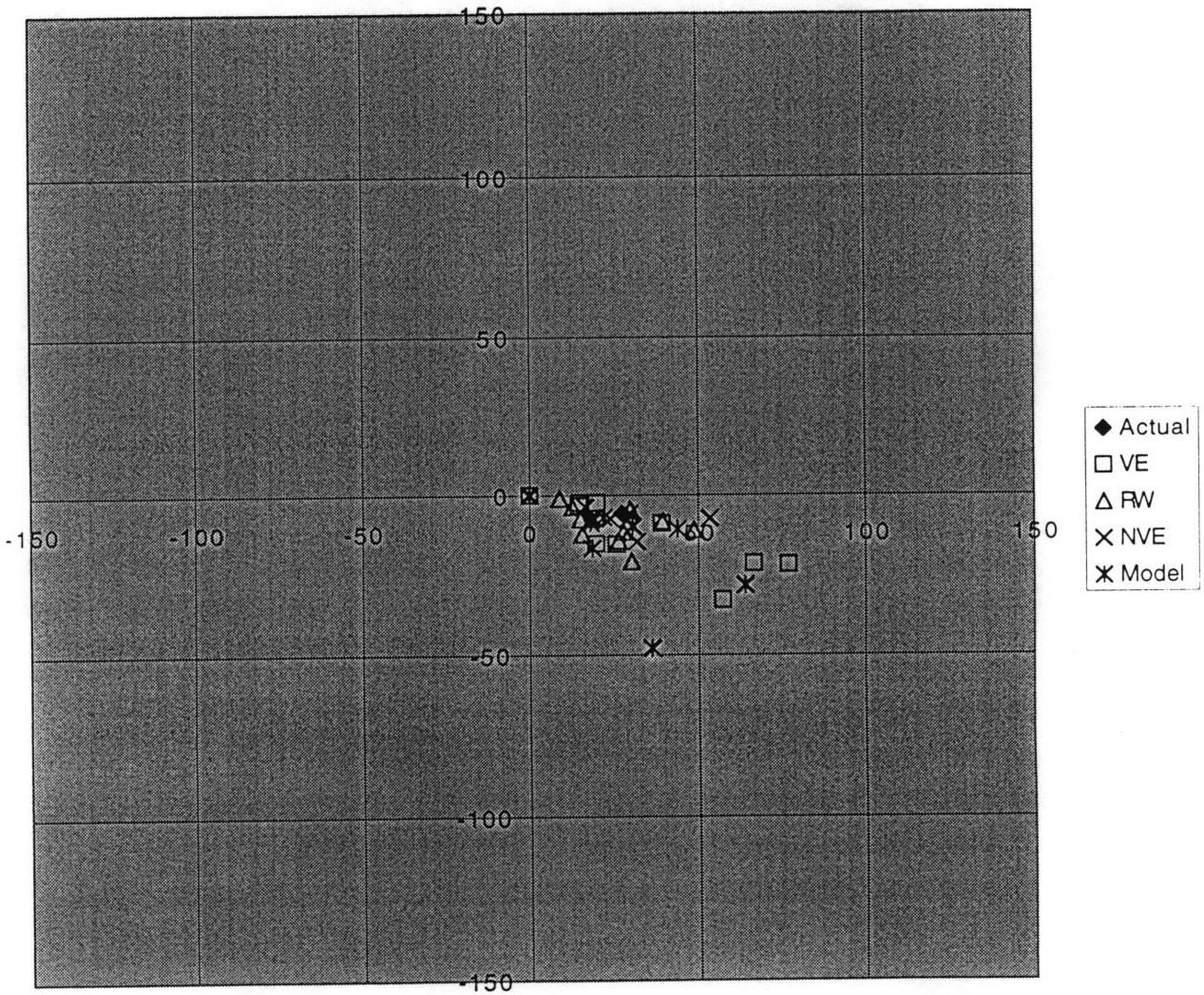
M. Elevator from 2



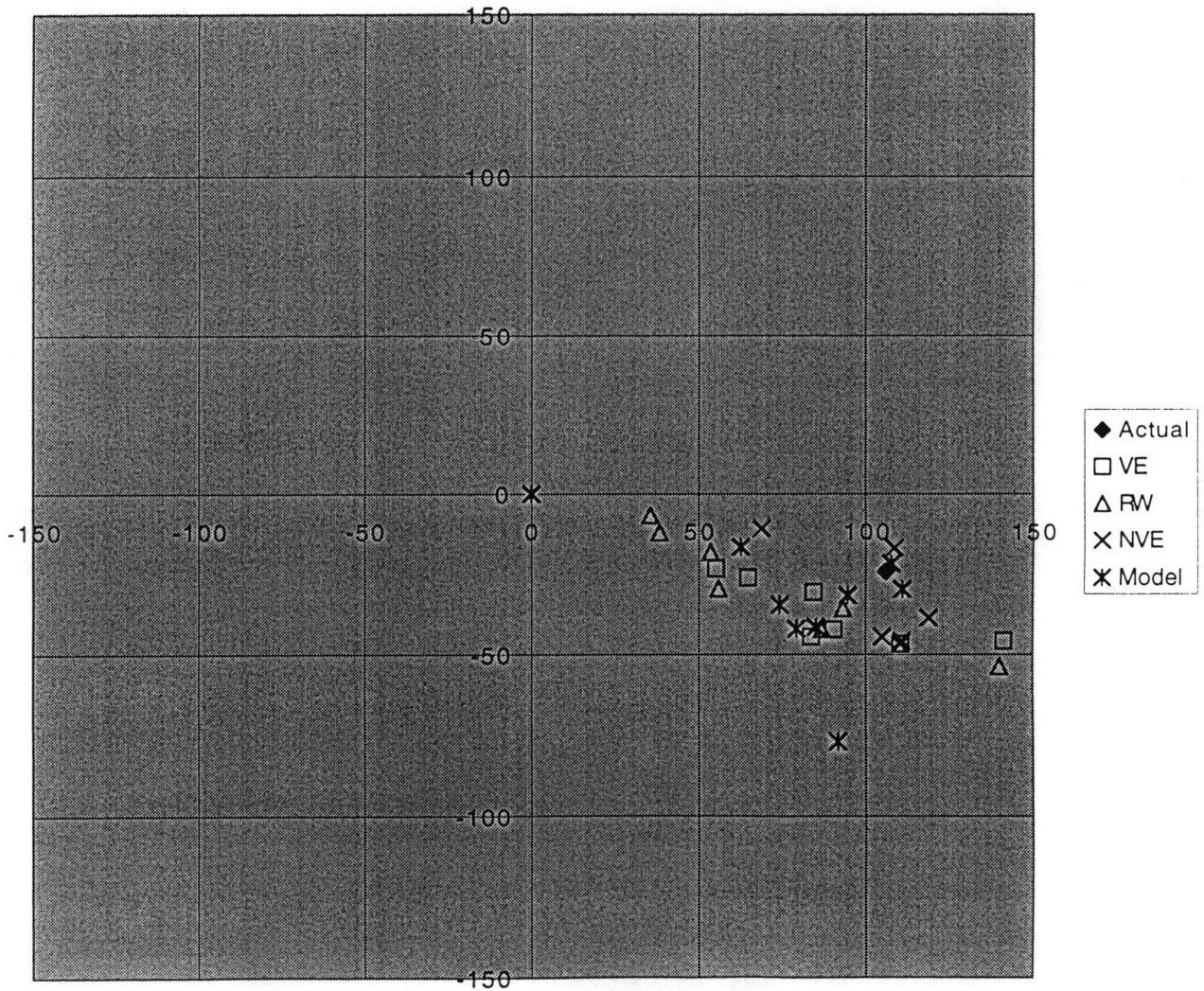
W. Fountain from 3



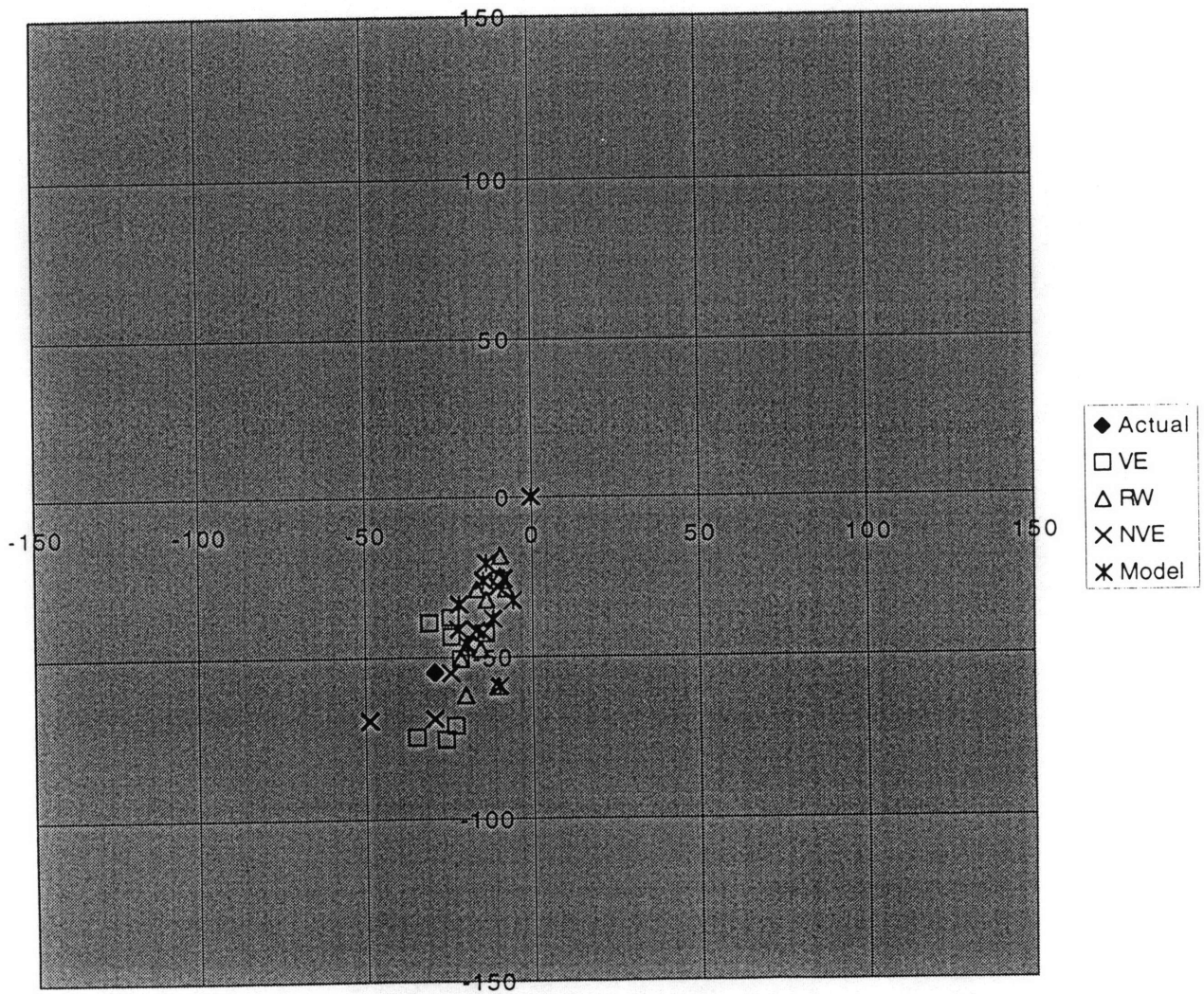
Microwave from 3



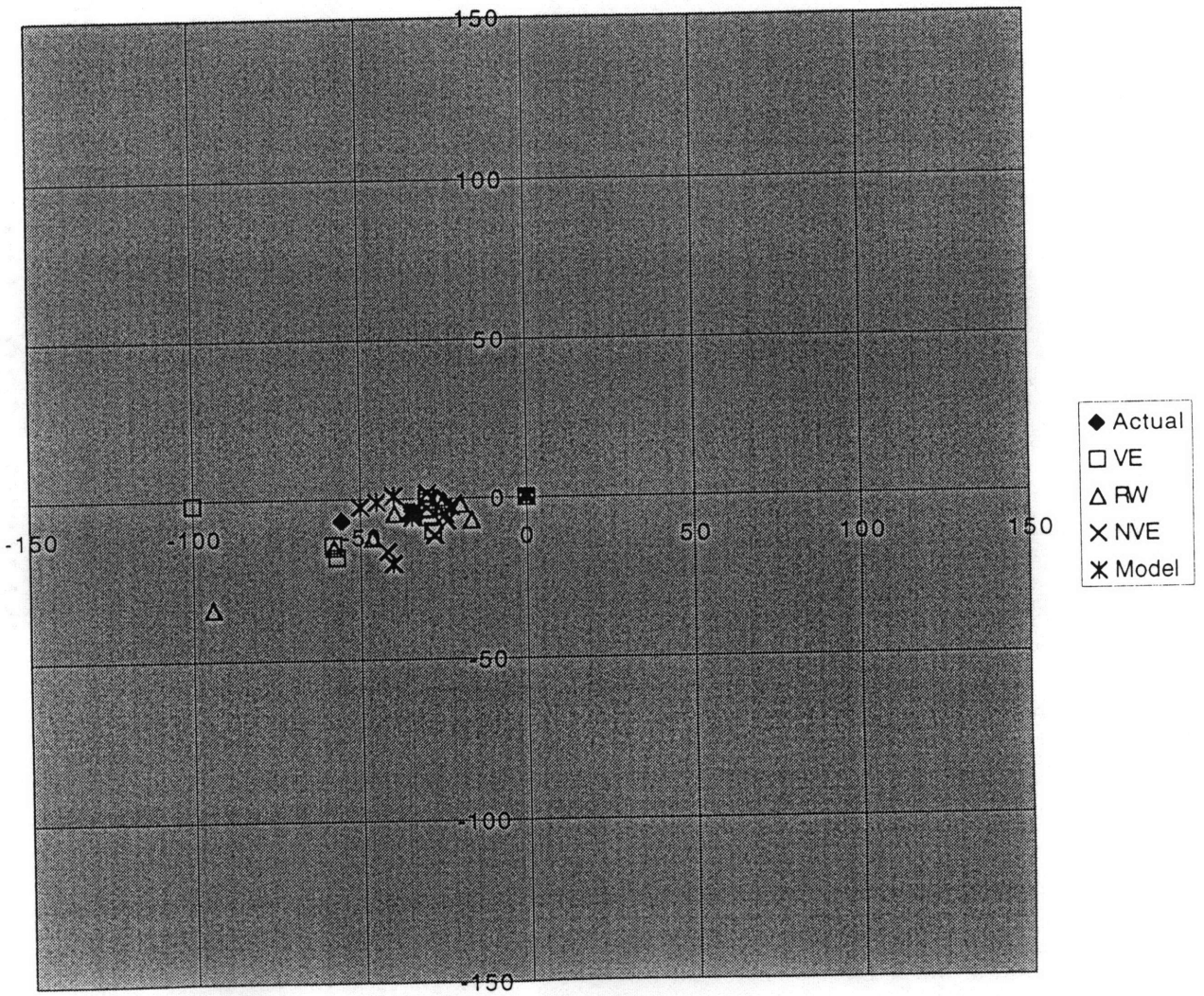
End of Hallway from 3



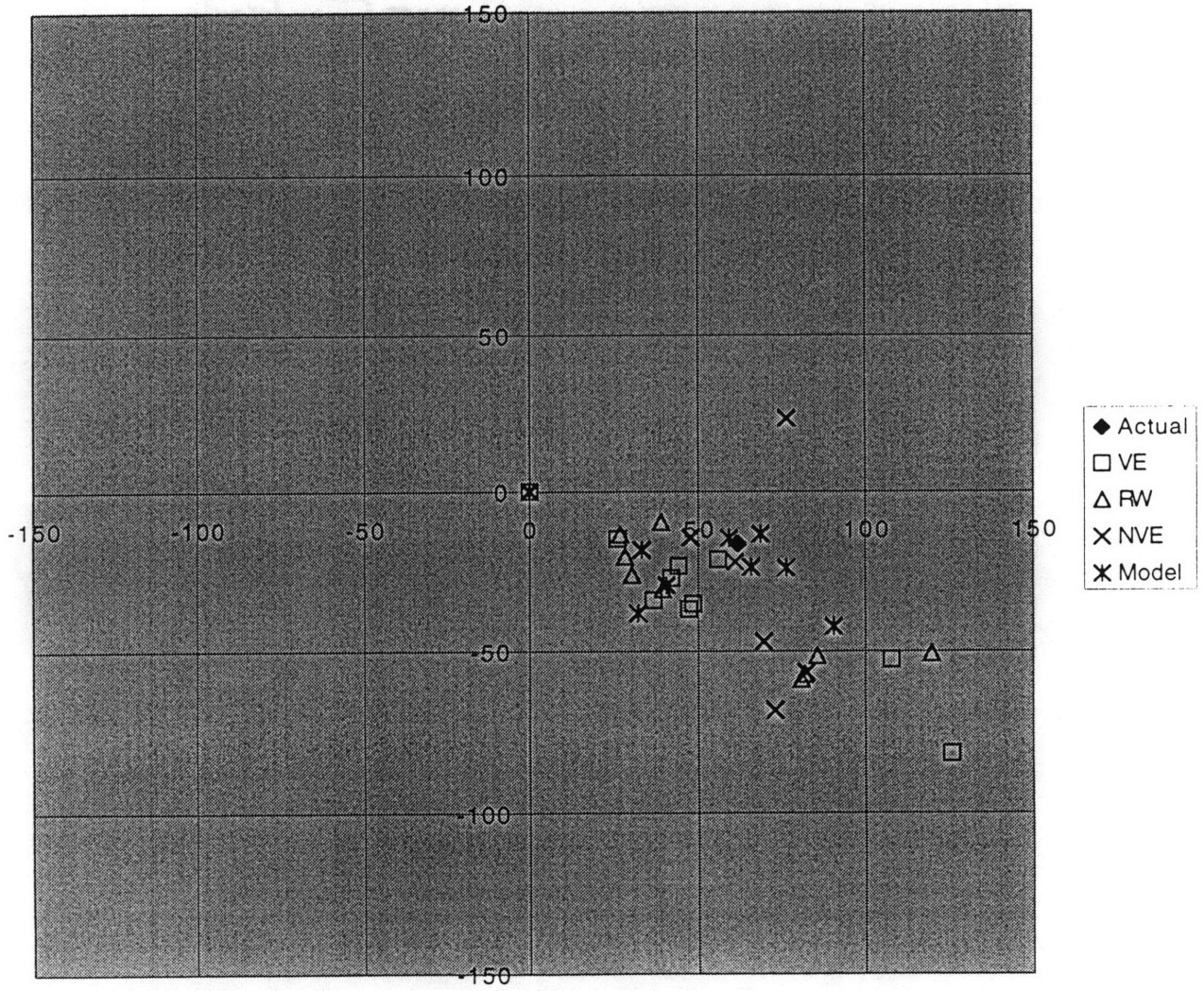
M. Elevator from 3



Conf. Table from 4



End of Hallway from 4



6.0 References

Satalich, G.A. (1995). Navigation and Wayfinding in Virtual Reality: Finding the Proper Tools and Cues to Enhance Navigational Awareness. *Masters Thesis.* University of Washington.

Mandler, J.M. (date). The Development of Spatial Cognition: On Topological and Euclidean Representation

Siegel, A.W., and White, S.H., (1975). The Development of Spatial Representation of large-Scale Environments. In H.W. Reese (Ed.), *Advances in Child Development and Behavior.* New York: Academic Press.

Witmer, B.G., Bailey, J.H., and Knerr, B.W., (1995). Training Dismounted Soldiers in Virtual Environments: Route Learning and Transfer. U.S. Army Research Institute for the Behavioral and Social Sciences.

Siegel, A.W., (1981). The Externalization of Cognitive Maps by Children and Adults: In Search of Ways to Ask Better Questions. In L.S. Liben, A.H. Patterson, & N. Newcombe (Eds.), *Spatial Representation and Behavior Across the Life Span: Theory and Application.* New York: Academic Press.

Tate, D.L., Sibert, L., and King, T., (1997). Virtual Environments for Shipboard Firefighting Training. In Proceedings of the Virtual Reality Annual International Symposium (VRAIS '97), Albuquerque, NM, pp. 61-68.

Pausch, R., Proffitt, D., and Williams, G., (1997). Quantifying Immersion in Virtual Reality. Currently submitted to ACM SIGGRAPH 1997. University of Virginia.

Stoakley, R., Conway, M.J., and Pausch, R., (1995). Virtual Reality on a WIM: Interactive Worlds in Miniature. University of Virginia.

Brooks, F.P., Jr., (1992). Six Generations of Building Walkthroughs. Final Technical Report, Walkthrough Project. National Science Foundation TR92-026

Darken, R.P., and Sibert, J.L., (1993). A Toolset for Navigation in Virtual Environments. *Proceedings of the ACM Symposium on User Interface Software and Technology.* November, 1993, Atlanta, GA 157-165.