Officer of the Deck: Validation and Verification of a Virtual Environment for Training

Nicholas J. Pioch

RLE Technical Report No. 596

June 1995

The Research Laboratory of Electronics
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139-4307

This work was supported in part by the Naval Air Warfare Center Training Systems Division under Contract N61339-94-C-0087 and in part by by the U.S. Department of the Navy Office of Naval Research under Grant N00014-93-1-1399.

Acknowledgments

The author would like to acknowledge the sponsors of this research, NAWC/TSD in Orlando, and ONR in Washington, D.C. He would also like to thank Bill Levison and the other members of the BBN training team for their guidance on relevant training issues and their supervision of the pilot experiments and help and advice with the perceptual cue tuning. The author is also indebted to Officer Allen Andrew and the other officers from the school in Groton, CT, for their valuable critiques of the OOD simulation and information about the details of the task domain. Finally, a heartfelt thanks to David Zeltzer, my advisor, for always guiding the thesis in the right direction, and to VETT colleagues Walt Aviles, Jim Bandy, Rakesh Gupta, Dorrie Hall, J.F. Lee, Jonathan Pfautz, and Brett Reid, at MIT; and David Fowlkes and Scott Davidson in Orlando, without whom this work would not have been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this era of decreased defense spending, the military is increasingly turning to computer simulators to train personnel in all branches of the service. Simulators can systematically train for a wide range of possible scenarios without the high cost and high risk of actual flight time in an Air Force jet or cruise time in a Navy submarine. Yet, conventional simulators are not without problems of their own, such as lack of reconfigurability and the need for large graphics displays and physical mockups of the simulated vehicle. To address these issues a growing body of research is being directed toward using virtual environments for training. In a virtual environment (VE) a trainee interacts with a 3D computer-generated world by seeing, hearing, and even feeling simulated objects. Since VE simulators rely more heavily on computer displays and sensors than conventional simulators, they may eventually be more cost-effective; the same set of devices and displays could host a range of simulators for different vehicles. Moreover, they are likely to be portable and reconfigurable, so that VE systems could even be taken onboard different Navy vessels, with only a "turnkey" effort needed to convert the simulation to a different vessel [SZ93].

1.1 Virtual Environment Technology for Training

Part of this research is taking place here in the Sensory Communication Group of the MIT Research Laboratory of Electronics. The Virtual Environment Technology for Training (VETT) project, sponsored by the Naval Air Warfare Center Training Services Division

(NAWC/TSD) and the Office of Naval Research (ONR), is aimed at exploring the value of using VEs for training. The VETT project team at MIT has implemented a high-performance distributed VE testbed to support the chosen training applications of the VETT advisory committee and its satellite groups. The VETT lab currently contains several Silicon Graphics workstations (including an Onyx with Reality Engine 2), field-sequential stereoscopic glasses, a large-screen projector, a VPL Dataglove2, a Virtual Research VR4 head-mounted display (HMD), several Polhemus 6-DOF position sensors, and a force-feedback device called the PhanToM, developed by Thomas Massie of the MIT AI lab. The initial VEs implemented on the testbed involved only simple graphics databases and interactions, mainly to test the fully integrated configuration of the hardware and software components. For example, one demonstration used the HMD and Polhemus sensor to allow a user to view a graphical representation of the lab called the "virtual room." Users could also wear a VPL dataglove with a Polhemus sensor attached, to see a virtual representation of their right hand; interactions were chiefly gestural, including grabbing and throwing of certain objects as well as pointing to "fly" around the room (Figure 1.1).

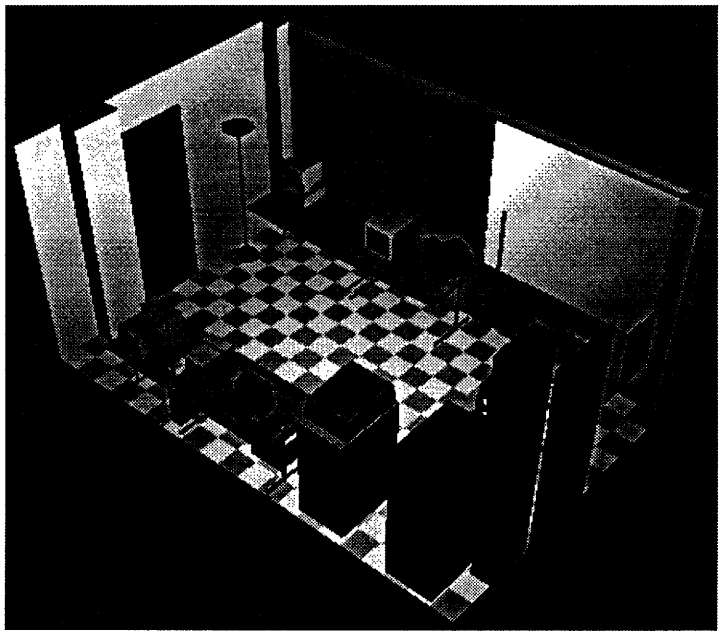


Figure 1.1 Virtual room demonstation.

Another demonstration combined visual, auditory, and haptic sensory feedback to render a virtual "air hockey" game. Users could wear stereo glasses to see a four-sided playing

surface with puck and hockey stick in 3D. They would use the thimble on the end of the PhanToM's robot-arm (Figure 1.2a) to control the hockey stick and feel collisions with the puck or the hockey table (Figure 1.2b). Each time the stick or the puck struck a wall, a sound would be heard based on the material of the wall and the force of the collision. From these simple virtual worlds which demonstrated the testbed's capabilities, the MIT group began work on its first real training task for VETT.

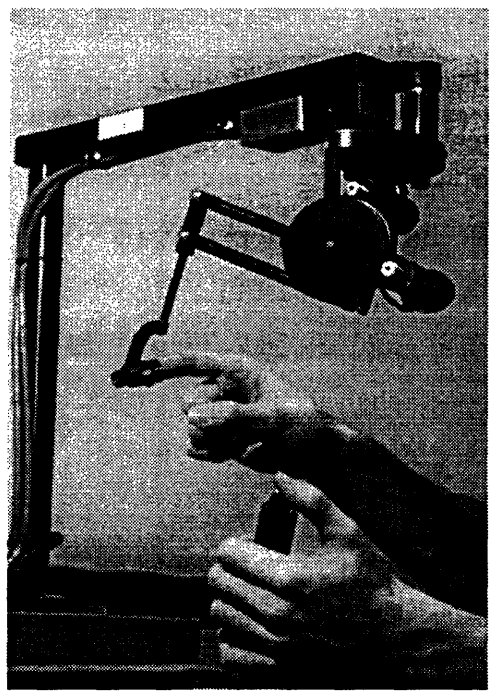


Figure 1.2a) PhanToM force-feedback device.

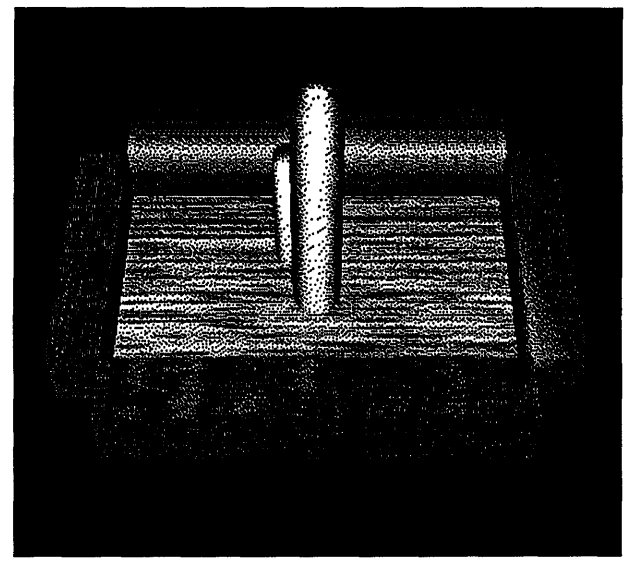


Figure 1.2b) Virtual puck demonstration.

1.2 The Officer of the Deck

Current efforts of the VETT testbed have been directed toward a VE simulator for the "Officer of the Deck" (OOD), a position that rotates among several junior submarine officers. One of the chief responsibilities of the OOD is to command the vessel safely into and out of a harbor from a vantage point above deck called the conning tower. The OOD receives advice from a chief navigator leading a piloting team below deck and gives commands verbally to a helmsman, also below deck. Of these three participants, the OOD is the only one with a full unimpeded view of landmarks, navigation aids such as buoys, and incidental water

traffic in the vicinity of the submarine. Although the OOD's duties include many different tasks, the simulator is aimed at training the harbor navigation task, which will henceforth be referred to as the "OOD task." Such a VE simulator would fill a large gap in submarine training procedures, as there is currently not even a conventional simulator for the OOD task, even though simulators exist for the navigator and the helmsman. Trainees learn solely from watching experienced personnel perform the task. They must be constantly supervised until they have improved enough to do it alone, receiving less and less supervision as they improve. Although there is little or no haptic component to this task, the necessity of an all-directional view, the verbal nature of the communication, and the cognitive task of harbor channel navigation make this an interesting candidate for VE training on the testbed.

With the help of VETT training experts from Bolt, Beranak and Newman, Inc. (BBN), the MIT testbed team has implemented the initial version (Version 1.0) of the OOD simulator. This version uses SGI's Performer for detailed graphics rendering of a model of King's Bay, Georgia (chosen for its relatively flat topography), including key navigational objects such as buoys, range markers, and turning aids. A Polhemus sensor is mounted on the VR4 HMD, allowing the system to render the appropriate view for any head position and orientation. BBN's HARK speech recognition software is used to recognize submarine commands as well as some system-specific commands for alternative views of nautical maps and textual tables. Feedback from the helmsman is given through prerecorded audio files, chosen according to the recognized speech command. The major software components include the main graphics loop, written in C using Performer; the submarine dynamics in C++; the speech interface, written in C and a specialized grammar input file; a GUI-based Experimenter's Interface, written in C++ using the widget-based Forms library; and a global "blackboard" process for communication of data across machines.

1.3 Related Work

Very little work has been done specifically on validation and verification of virtual environments, especially as applied to training. Part of the reason for this is the subjective nature of "presence" in a VE system. The strength of presence depends highly on the individual,

as well as the types of devices used and the interactions they afford. The same VE may seem quite compelling to one user while another user may feel more immersed watching a passive 2D television show. Also, perceptual cues in the VE are difficult to formally or quantitatively verify, other than by written surveys or oral questions asking whether a user saw objects in stereo, for example, or perceived different colors of objects. To some extent, validation and verification may soon become more actively emphasized in the development of VEs, especially with the event of recent CASE tools such as Silicon Graphics' CASE-Vision, enabling construction of more readable and easily verifiable code for real-time VE simulations.

A good description of validation and verification for general software projects can be found in [Boe84]. Harwood defines validation and verification techniques for Air Traffic Control (ATC) through a categorization of "human-centered systems" issues into technical usability, domain suitability, and user acceptability [Har92]. Somewhat more relevant here is the work on simulation validation by Knepell and Arangno [KA93]. They comprehensively describe formal methods for validation of simulations, beginning with five major assessment processes: conceptual model validation, software verification, operational validation, data validation, and internal security verification. The application of these assessment processes to the OOD simulation will be discussed in Chapter 4.

Perhaps the most strikingly similar existing VE system to the OOD simulation is the MARS Virtual Reality Simulator, currently used in the Canadian naval forces. Built by the Defence and Civil Institute of Environmental Medicine (DCIEM), this VE system is used to train "Officers of the Watch" whose jobs involve keeping their vessel in strict formation relative to other nearby surface vessels [Mag93]. The commands used by Officers of the Watch are similar to those of American OODs, since both tasks occur solely on the surface. However, the MARS trainer teaches only open-sea maneuvers where the visual scene is relatively simple; the only objects in the ocean environment are the trainee's vessel and its companion vessels in the formation. The OOD task is considerably more complex, since it involves navigational aids such as buoys and supplemental information from nautical charts and tables. Furthermore, subjects in the MARS trainer do not need to worry about running aground in shallow waters or keeping the submarine in the center of a narrow channel. One

major advantage of the DCIEM effort was the prior existence of a routine formal training program for Officers of the Watch, allowing the researchers to conduct simple transfer-of-training experiments comparing groups trained in the VE to those trained only at sea. The results showed small but significant differences in the performance of the VE-trained officers over those who received the routine training. Designing similar experiments for the OOD simulator will be more difficult since there is no formal training or evaluation program for OODs; meaningful performance measures will have to be created from scratch [PL94a].

1.4 Autonomy, Interaction, and Presence

As of yet, no concensus on how to characterize or describe a given VE has emerged in computer-related fields. What is needed is a uniform taxonomy for describing VEs so that apples will no longer be compared to oranges. To ensure that the descriptions used in this thesis are worded and organized consistently, I will use a three-dimensional taxonomy based on autonomy, interaction and presence [Zel92]. Autonomy refers to the degree to which different agents in the VE (aside from the user) act and respond independently and/or intelligently to changes. Autonomy is determined by the computational models and processes used in the implementation of the VE. On the low end of this scale is a static dataset such as the "virtual room" above, and on the opposite end is a 3D "Holodeck" situation in which virtual people walk, talk, and touch like the real mccoy. Interaction is defined as the means by which the user can have an effect on objects or conditions in the virtual world and use the sensory feedback to determine the next course of action. Interaction occurs through a logical interface, which may be formally described through activity charts or state charts [Har87, Wel89]. Presence, used in many ways in various literature, here will refer to the number of different input and output channels in the sensory trackers and displays, and their ability to realistically display the VE, resulting in a sensation of being immersed in the virtual world. Presence is enabled by a physical interface that may include devices for several sensory communication modes, such as HMDs, speech recognizers, or force feedback joysticks. This three-part taxonomy will serve as the basis for the descriptions of the requirements, implementation, and validation of the OOD simulator in

the ensuing chapters.

1.5 Validation

Simulation using VEs is still in the infant stage as a technology and does not have a well established design, implementation, and validation methodology. In contrast, there is a considerable literature on conventional simulators, especially flight simulators, which must pass stringent tests in order to be certified by the Federal Aviation Administration for use in actual training [RS86]. In fact, any simulator which meets these requirements and is part of an approved training program may be used as a substitute for the use of a real aircraft [ERJD85]. With VE simulators, the specific approach used for validation may be somewhat different, but the need for such a validation is just as great. This thesis describes the methods used to validate and verify the implementation of the OOD simulator. The problem addressed is not whether the VE successfully transfers training performance to performance in the real world task; that is an open research question in itself. The thesis will tackle the more immediate necessity of validating the implementation decisions made so far and verifying that the current system meets the requirements specified in the design phase.

Chapter 2 provides a task analysis of the OOD's responsibilities and gives requirements for Version 1.0 of the simulation. Chapter 3 discusses the implementation of Version 1.0, referring to the corresponding requirements where necessary. Chapter 4 describes the validation of the implementation, with verification of some of the important modules and algorithms. Future work on remaining problems with the implementation and possible extensions of validation techniques to other VE issues are discussed in Chapter 5. Finally, Chapter 6 concludes the thesis by giving general guidelines for validation and verification of VEs for training.

Chapter 2

Requirements

After the OOD task was chosen for implementation on the VETT testbed, a series of design meetings between the MIT group and the BBN training group ensued. The purpose of the meetings was to determine the salient features needed in a VE to properly train the OOD task. For the most part, this was the responsibility of BBN, but close communication with the MIT group was necessary to keep BBN advised as to the capabilities and limitations of the current technology. Much of their initial analysis was based on interviews with a Chief Petty Officer from Charleston Harbor who had personal experience in the training of ship-handling techniques. Later, as the prototype system was built, the MIT group had the opportunity to demonstrate the simulator to naval officers who had recently performed the OOD task. Their feedback was highly valuable in identifying important features that were either missing or visually impoverished. For brevity's sake, this section will only describe the final set of requirements for the task, rather than the initial set and all the incremental revisions made during the design phase. Furthermore, both BBN and MIT realized from the start that simulating the full range of environmental conditions and communication aspects of the task would constitute severe real-time rendering problems and push experimental testing of the simulation well behind the desired schedule. Therefore, an iterative software development process was adopted, using a series of evolutionary prototypes (see [Gom90] for a discussion of throwaway prototyping vs. evolutionary prototyping strategies). The first OOD prototype, for which the requirements in this chapter are specified, will be referred to as Version 1.0. This first version simulates only the most essential elements of

the task, leaving additional task details for later prototypes. Breaking up the implementation in this way would allow a set of pilot experiments to be performed on Version 1.0 to determine whether novice subjects can use the system to learn the virtual harbor. If so, these initial experiments would also reveal the length of a typical learning curve. This and other experimental results could be used to choose which task elements to include in the next prototype. This chapter and the three succeeding chapters will discuss only Version 1.0, although some discussion of future versions is given in Chapter 5.

2.1 Task Analysis

In order to derive a reasonable set of requirements for a simulation such as this, an in-depth task analysis must be performed first. In the words of Drury et al., task analysis is "a formal methodology which describes and analyzes the performance demands made on the human elements of a system" [DPC+87]. The task analysis identifies the various kinds of actions required of the OOD and from this the components of the simulator's human-machine interface can be derived. Task analyses may use a wide variety of formats, but for this thesis a flowchart-based format is used, as demonstrated in [Car90] for logistical aircraft missions. Before describing the details of the flowcharts, however, an illustrated overview of the OOD's surroundings is in order.

2.1.1 The Harbor Environment

The OOD's prescribed harbor path is a narrow channel marked on either side by floating buoys, colored green on the left and red on the right, going in the inbound direction. Each buoy is uniquely numbered, to enable identification of the buoy on the navigational charts used by the OOD and the navigation team (Figure 2.1). The channel typically begins several nautical miles away from the bay. At the bay entrance, four yellow warning buoys mark a widened turning basin which is the last point at which the sub may turn around in the event of bad weather (the task is usually only performed under favorable conditions). After that, the OOD is committed to continue forward navigation of the channel till reaching the docking area. (the docking task is not included in this simulation). The channel itself

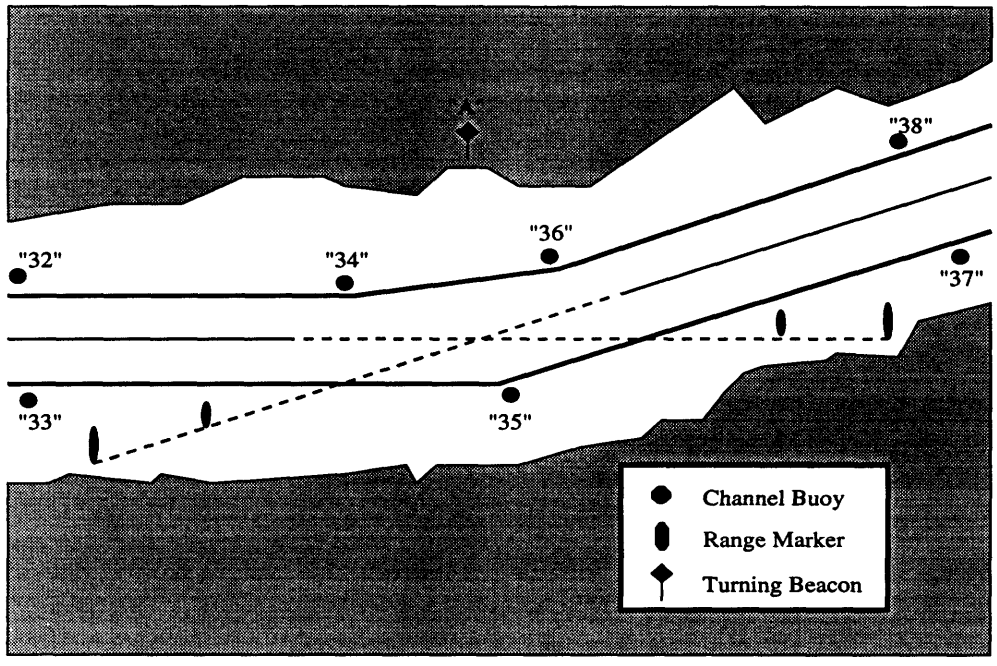


Figure 2.1 A sample harbor channel with numbered buoys.

is divided into a series of straight, connecting segments. Except on turning areas at the transitions between segments, the OOD's main goal is to keep the submarine as close as possible to the segment's centerline while compensating for water currents and avoiding conflicts with occasional water traffic. To aid with centerline alignment, a pair of range markers is placed along an extension of the centerline; if the smaller marker is in line with the larger one, the submarine is on the centerline (Figure 2.2a). If the larger one appears to the right of the taller one, then the vessel is to the right of track and must adjust left to regain the centerline (Figure 2.2b).

Making an accurate turn from one straight segment to the next is one of the most challenging aspects of the task. To help the OOD with the timing of a turn, special navigation objects called turning aids are used. Each turning aid is precisely placed so that the next turn must be started when the line of sight to it reaches a certain prescribed bearing. This bearing is measured with a hand-operated compass.

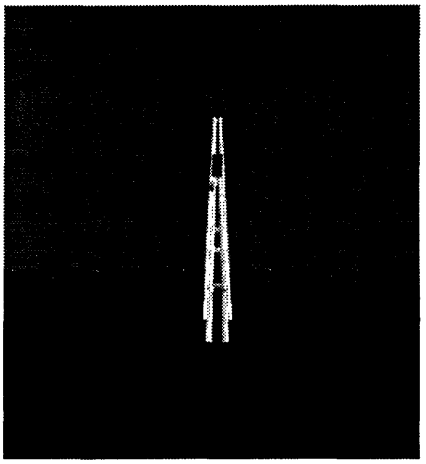


Figure 2.2a Range markers aligned.

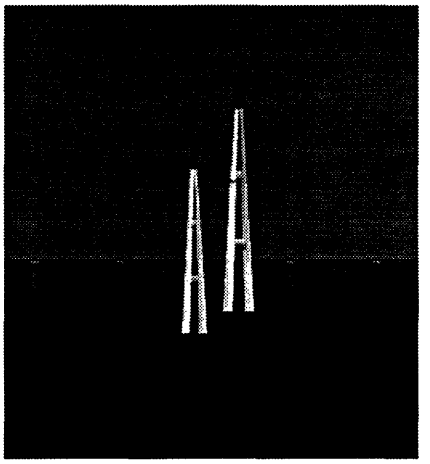


Figure 2.2b Ranges showing right of track.

2.1.2 Navigator and Helmsman

Although the OOD is effectively in charge of commanding the movements of the boat while it is on the harbor surface, he does not perform the navigation task unaided. The helmsman below deck actually operates the steering controls of the sub, carrying out and acknowledging each verbal OOD command as it is received. Furthermore, members of the piloting team, also below deck, operate the periscope and employ triangulation methods to obtain accurate estimates of the boat's current location on a detailed nautical chart. When it is deemed necessary or useful, the chief navigator occasionally gives information or advice to the OOD, including distance to the next turn, approximate distance from the centerline, recommendation to use a new course heading or to maintain current heading, and "yellow warnings" given when the sub approaches dangerous water depths close to or outside the edge of the channel. Alternatively, the OOD may at any time give a verbal request to the navigator for any of this information. However, if visibility is sufficient, the OOD should be able to make reasonable judgments on his own about his location, centerline track error, and turning times, based solely on his surroundings. In addition, the OOD is the only one in this group who has a full unimpeded view in all directions, allowing him to identify any approaching water traffic with whom a mutually safe passage must be negotiated.

2.1.3 OOD Task Flowcharts

The figures in Appendix A show the logical flow between the different elements of the OOD task. To a lesser degree of detail they describe the activities of the navigator and helmsman as well, and how they interact with the OOD. The flow charts are hierarchically organized using a top-down approach, with Figure A.1 giving the highest level breakdown of the task. Dashed action boxes indicate optional actions and dashed, directional vertical lines represent communication from one team member to another. Shadowed action boxes are "opened up" to a lower level of detail in other figures. The initial location of the submarine at the beginning of the task is assumed to be at the start of the first channel segment, between the first pair of red and green buoys. The submarine is headed along the inbound route beginning at standard speed (12 knots). Figure 2.3 below shows a simplified diagram of the submarine and the locations of the OOD and other crew members involved in the task.

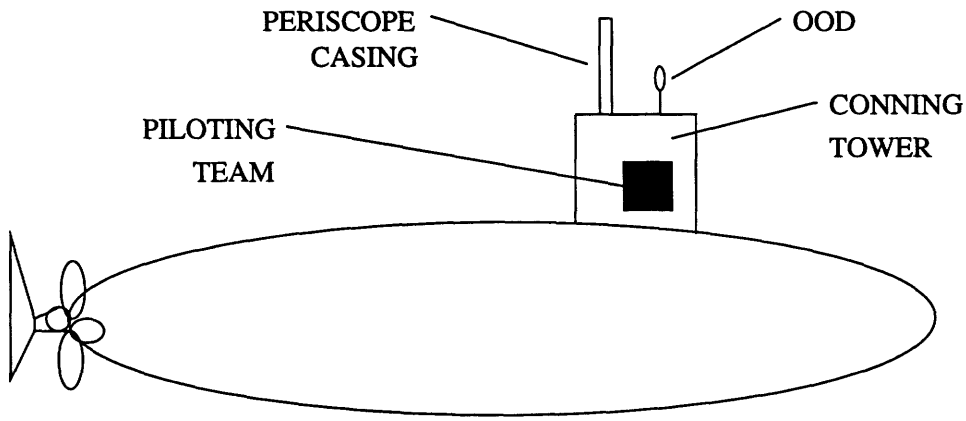


Figure 2.3 The OOD commands a piloting team below deck from his vantage point on the conning tower.

The first task element is centerline alignment (Figure A.2). In the first action box, the OOD consults a "course card" containing numerical information about each channel segment. He focuses on the top row which corresponds to the first channel segment, using the third column to determine whether the range markers are located off the bow side or the stern side. Depending on the answer, he looks either straight ahead or directly aft (careful

to look around the periscope casing) to find the pair of range markers for this segment, using binoculars if necessary. The next decision box asks whether the range markers are perfectly aligned. If so, the OOD allows the helmsman to maintain the current heading. If not, the OOD must give an appropriate correction to the left or right of the current heading, varying the size of the correction in degrees according to the amount of separation between the misaligned pair of range markers (Figure A.3). As described in Figure A.4 this occurs typically through a steer command, "Steer two-six-eight" for example, in which the OOD specifies a desired three-digit compass heading in degrees. Alternatively, if the heading change is great enough, the OOD may give a rudder command such as "Left standard rudder" to set the rudder to a specific angle, and *then* give a heading command as above to let the helmsman know the heading on which the turn should stop and the vessel should be straightened. If the OOD is unable to get the necessary information from the range markers, he may instead ask for advice from the navigator and give the recommended steering commands.

The second task element is turning aid identification (Figure A.5). Again, the OOD consults the course card, this time using the fourth column to find the name of the turning aid provided for this channel segment. The OOD then looks at the chart and identifies that navaid. By using his current estimate of the sub's location on the chart, he determines the approximate direction in which to look for this navaid, and then looks in that direction until he sees it, using binoculars if necessary.

The third task element is making the turn (Figure A.6). Now the OOD checks the fifth column of the course card for the turn bearing indicating the timing of the next turn. He makes a mental note of the value (call it angle α). He then looks for the turning aid found in the last step and makes use of his compass to get an accurate bearing on it. This step is repeated until the compass reading nears angle α. Then the OOD checks the course card again to find the heading of the next segment (given in column 1 of the subsequent row), mentally noting this number as well, call it angle β. Then the OOD again uses the compass to take bearings on the turning aid and when the compass reading finally reaches angle α, he gives a command to steer the boat to angle β. This can again be done by a single heading command or as a rudder command followed by a heading command, as in Figure

A.4. If the OOD is not sure when the turn is coming up, he may ask the navigator for an estimate of the distance to the turn, following any additional recommendations given as well. On more difficult turns, the speed of the submarine might be lowered to 8 knots with the command "All ahead two-thirds" or even to 4 knots with the command "All ahead one-third." A standard speed of 12 knots may be resumed at the end of the turn with the command "All ahead standard."

After the turn is completed, the OOD draws a line in pencil through the row of the course card corresponding to the finished segment. Then, the task loops back to task element one to correct for any misalignment with the next channel segment. This process repeats until the last segment is completed.

Note that there are several ongoing task elements which run "in parallel" with the three sequential task elements above. The first is avoiding collision with navaids (Figure A.7). This consists of frequent checks in the vicinity of the submarine and some distance along its current heading to make sure the boat is not headed for a buoy or other navaid. If it is headed for a buoy, a correction must be made to the left if the buoy is red and to the right if it is green (the reverse holds for the outbound case). The second parallel element is monitoring water traffic (Figure A.8). The OOD must also check for other watercraft which may occasionally venture close to the path of the sub. Should this occur, the OOD must attempt to make radio contact and negotiate a mutually safe passage. In an emergency situation in either of the above elements, the OOD may have to stop the sub or even reverse direction to avoid a collision with an ocean object or watercraft. The third parallel task element is compensating for water currents (Figure A.9). Strong water currents and backcurrents often skew the sub from its intended heading. Thus, the OOD must frequently recheck the alignment of the range markers, this time giving a slightly above normal correction in the opposite direction of the drift. When the markers finally align again, the OOD may wish to give a heading command with an extra offset proportional to the strength of the current, in the direction opposite to the drift.

2.2 Autonomy: Computational Models and Processes

This section describes the computational models and processes required of the VE for adequate training of the harbor navigation task described above. This includes specifications for the visual representation of objects as well as their behavioral attributes. Most of the section deals with the environmental features external to the submarine, except for the concluding paragraphs which discuss the sub's visual representation and dynamics. The design decisions in this section and subsequent sections have been strongly influenced by guidance from the BBN training team, as well as by the engineering tradeoffs regarding different VE devices and displays.

2.2.1 Channel Segments and Centerlines

Since the task at the highest level is to navigate a harbor channel, the simulation must be given the exact configuration of the channel and how it bends from segment to segment. A simple polygonal representation should be sufficient since the channel is a series of straight segments, with no curved edges,

For the longer segments, of length 500 yds. or more, a centerline must be specified, since it is the optimum location for staying inside the channel. Centerlines are also important for performance characterization of an OOD's path in initial experiments, although in real situations close centerline tracking is given a wide range of emphasis depending on the preference of the captain. Each centerline extends from a point at least 200 yds. from the beginning of the segment to a corresponding point over 200 yds. from the end of the segment. This allows the OOD to perform a turn without fear of suffering a penalty for deviation from a centerline.

Even though the channel boundaries and centerlines will not be explicitly displayed as physical objects in the VE (except in the map view), the encoded knowledge of their locations relative to the submarine is important for performance measurements of simulation experiments. Later versions of the simulation may actually use visual display of the channel and centerline as artificial cues to aid performance (see Section 5.2).

2.2.2 Buoys

Buoys have three main identifying features: shape, color, and number. All of the buoys marking the boundaries of the channel are the same basic shape and have heights of either 16 ft. or 11 ft. above the water surface [Gua94]. As one enters the channel going inbound, the buoys on the left are green and those on the right are red. The number on these buoys is a block-style number painted in white on the upper right corner of the buoy, on the side facing the channel (Figure 2.4a). Identifying the buoy number is an important part of the navigation task. Therefore, in the simulation, rather than recreating the number size faithfully, the numbers should be enlarged to compensate for the resolution limitations of the HMD. Subjects must be able to read the number on the nearest buoy at all times so that they can identify that buoy on the simulated chart to help determine their position. Thus, the maximum distance to the closest buoy over all points in the channel must be measured from a valid nautical chart of King's Bay and subjects must be able to read buoy numbers, using binoculars if need be, from all distances less than this threshold. Figure 2.4b shows that this maximum distance is 540 yds., between the red buoys numbered 30 and 32. Therefore, the buoy numbers in the VE must be large enough to be legible from distances of 540 yds.. Since we are not training for visual target acquisition, any necessary size or color changes made to the VE buoy numbers to accomplish this goal is justified, as long as the main buoy color does not change. Furthermore, based on discussions with naval officers, buoys must be visible from distances up to 2000 yds., and buoy colors should also be distinguishable up to that distance. If this is not achievable using faithful buoy sizes, then the buoys must be scaled up to meet this threshold of visibility.

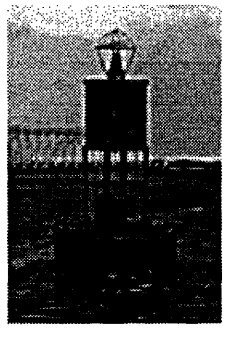


Figure 2.4a) Channel buoy.

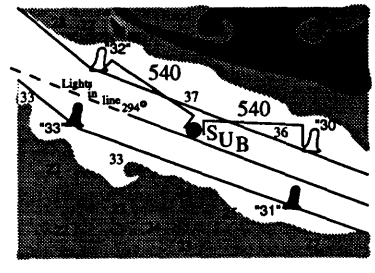


Figure 2.4b) Maximum distance to nearest buoy.

2.2.3 Range Markers

Range markers are tall structures of varying form used as visual aids to align the submarine with the centerline of the channel segment. They always appear in pairs outside the channel proper, on the extensions of the centerlines. Though range markers may be found on land or in water as lightships, towers, or even simple wooden posts, the task-relevant feature common to all is a striped rectangular "dayboard" on the top, 4 ft. by 8 ft. in dimension, which faces the channel centerline [Mal77]. The goal is to align the dayboards of the two range markers on the current segment. Perfect alignment means that the vessel is currently on the segment's centerline (Figure 2.5a). If the rear range marker appears to the left (right) of the front marker, then the sub must be left (right) of the centerline (Figure 2.5b and 2.5c). Occasionally, a segment uses a rear-oriented pair of range markers, in which case the OOD must look astern to check his alignment (Figure 2.5d), making navigation slightly more difficult, due to a possible obstruction by the periscope and a distraction from the forward direction. Range markers in King's Bay vary in height from 16 ft. to 98 ft., thus the OOD must often use binoculars to discern dayboard separation from distances of more than a nautical mile. Some range markers also function as turning aids which are described below.

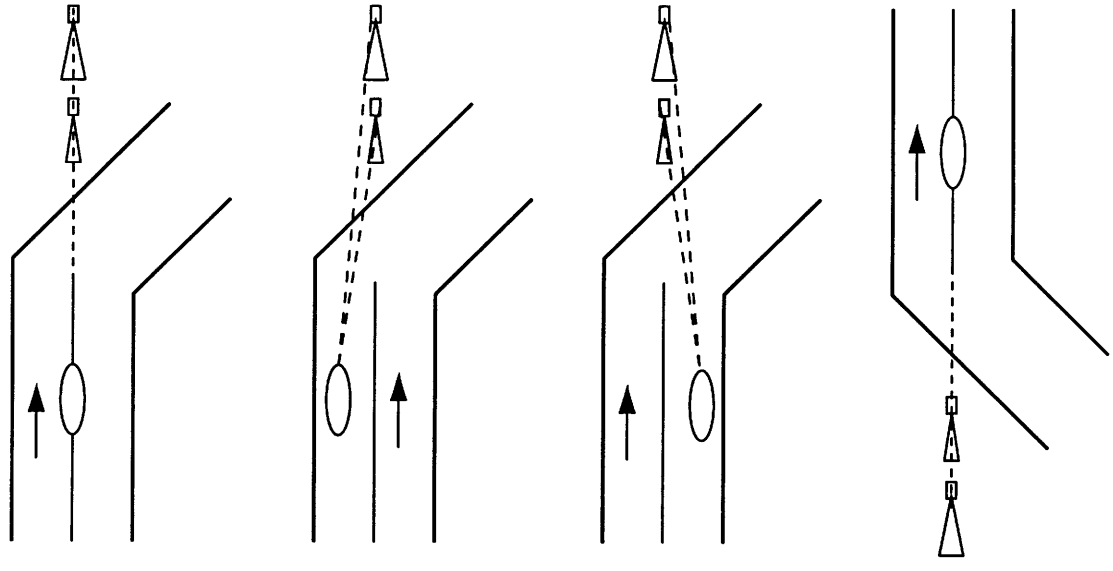


Figure 2.5 The arrow represents the submarine's direction of motion.
a) Perfect alignment. **b)** Off-track left. **c)** Off-track right. **d)** Perfect rear alignment.

Unlike buoys, identification of a particular range marker (unless it is also a turning aid) is of secondary importance, since they are already of a distinctive shape and color. Accuracy in location is the most vital attribute for these objects, since the slightest offset in either direction could provide a negative cue that will force the OOD off course. First, positions of each tower must be read in from a valid nautical chart as accurately as possible. Then, their positions must be checked geometrically to be on the centerline of the corresponding channel segment. If a marker is off track, it must be moved to the closest point on the centerline extension (Figure 2.6). Some markers at the intersection of two centerline extensions may be used for more than one segment, so these special markers should be placed at the trigonometrically calculated intersection.

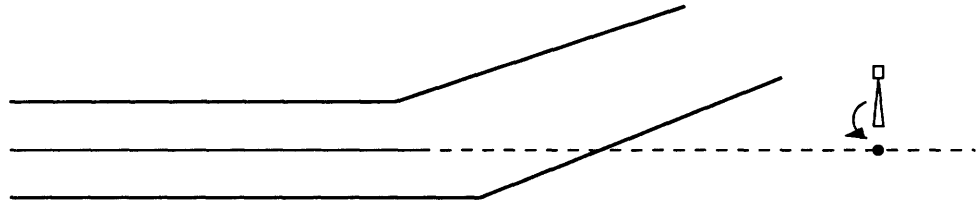


Figure 2.6 Adjustment of an off-track range marker.

Another issue is height scaling. The specified heights from a valid nautical chart must be used as faithfully as possible, but if the range markers are too small to be recognized in the simulation from a distance, then an upscaling of the heights may be necessary. Moreover, not only must a subject be able to discern the range markers themselves, but they must be to discern a horizontal separation between the dayboards when the sub is appreciably off-track. In particular, for deviations of 20 yds. or more, subjects should be able to tell whether they are right or left of the centerline from the apparent range marker dayboard separation (with use of binoculars allowed). Any upscaling of the range marker heights to meet this requirement is justified.

2.2.4 Turning Aids

Turning aids are used to help the OOD decide when to begin a turn from one segment to the next. When the vessel approaches the end of a channel segment, the OOD must

locate the turning aid mentioned in the course card (see section 2.3.2.5) and estimate its bearing with the aid of a compass. When the bearing reaches the value specified in the course card, the OOD should command a turn, assuming he is currently on the centerline of the original segment (appropriate corrections must be made to this value if the sub starts from an off-track position). For King's Bay, the course card bearings also assume this is done at the "transit speed" of 12 knots; faster speeds would produce a larger turn radius and thus require a slightly earlier turn, while slower speeds allow a later turn time. Turning aids come in several forms, including lights, special beacons, or even range markers doubling as turning aids. In King's Bay there are eight turning aids, four of which are special diamond-shaped or triangular daybeacons and four of which are range markers, also used for centerline alignment on other segments of the channel (figure 2.7).

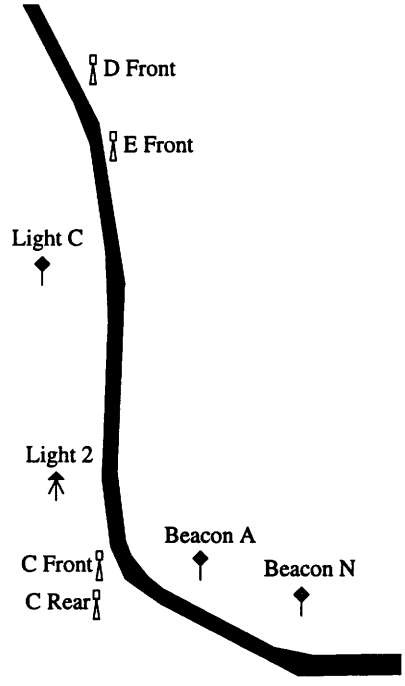


Figure 2.7 Turning aids for the King's Bay channel. (Figure not drawn to scale).

The most important attribute of turning aids for VE training is again accurate placement. Their positions must be derived carefully from the chart and then checked with the bearing prescribed in the course card to validate that the submarine will go from one centerline to the next if a turn is begun at 12 knots with the beacon at that bearing. Chapter 4 describes several approaches to this validation including geometric modeling, autopilot testing, and expert human-piloted sessions using the fully integrated simulation.

2.2.5 Land and Water Appearance

King's Bay was chosen in part because the topology of the mainland and islands is very flat, and cultural features are few in number. This should make the visual simulation of the land relatively simple to implement (recreating the bridge-studded San Francisco Bay could take months just to recreate the skyline). However, the shape of the shoreline must be accurately drawn, matching closely that of a valid nautical chart. If actual trees cannot be drawn, at least texture simulating green grass and trees must be used on the land to provide perspective depth cues that help the user estimate the submarine's location relative to the land.

The ocean must be colored light blue, with a static texture simulating the rippling effect of calm seas. No wave action or bowspray is required in Version 1.0. Also, the wake behind the submarine or around buoys (due to currents) need not be represented in Version 1.0.

2.2.6 Water Depths

To enable the simulation to detect whether the submarine has hit bottom in shallow waters, ocean depths must be provided in the vicinity of the channel. While it is surfaced, the submarine typically extends to a depth of 28 ft. below the surface. Thus, in locations where the water depth is less than 28 ft., the system must produce some sensation of running aground. A short but violent shaking of the visual view followed by a verbal message from below deck should be sufficient. Depths inside the channel vary depending on tides and weather conditions, but are uniformly safe for the submarine (except for two points, miles from the bay entrance), hence it is sufficient to model the entire channel at a uniform depth. A local pilot often accompanies the OOD to help advise on water currents, tidal and

weather conditions, and the locations of shallow areas and deeper areas (this participant is not to be included in Version 1.0). Near the edges of the channel, the water depth database should have a resolution of at least one datapoint every three square arc-seconds (approx. 100 yds.), since some areas of the seabed just outside the channel rise up to unsafe depths in just 100 yds.. Areas more than 400 yds. outside the channel need only a resolution of one datapoint every ten square arc-seconds, since the submarine is less likely to even make it this far out where the the depths are almost always unsafe.

2.2.7 Submarine

The submarine requires the most complex computational model in the VE since it is the focus of the training task and is the only object in Version 1.0 that moves and reacts to the trainee's commands. Both detailed visual representation and accurate equations for the dynamics of the submarine are necessary.

2.2.7.1 Visual Representation

Since the viewpoint throughout most of the task will be centered on the conning tower of the submarine, a detailed polygonal model for the submarine is necessary. In particular, a model of a Los Angeles class fast-attack submarine must be used, since this is the class of vehicle being simulated for the OOD task. If such a detailed model is not commercially available, commercial CAD or modeling packages should be employed to draft a less-detailed model on which a series of color texture maps derived from photographs should be applied. The rudder in the model must be kinematically articulated to rotate about a Z-axis through the stern of the sub (pitch rotations are not required since the task does not involve diving). Based on conversations with naval submarine officers, the submarine should be drawn low enough underwater to submerge the entire propeller and to see several yards of water between the rudder and the upper hull.

2.2.7.2 Dynamics

The dynamics for the submarine should be obtained from a valid source and encoded in a module separate from the graphics process so that unit testing and simple simulations may

be performed without having to "jack into" the full VE system (printing the submarine state parameters to the terminal or to an experimenter's interface file is often sufficient during development and testing). This will also allow the dynamics to be run on a separate machine from the graphics process in case the additional processor load from the dynamics computations would unacceptably slow down the graphics frame rate. The complexity of the dynamics model should be kept within reasonable limits so that the frame rate of the dynamics process at least exceeds that of the graphics process (see Section 2.4.1).

2.2.8 Other Watercraft

In the real world, other watercraft can constitute serious obstacles to the safe navigation of a harbor channel. Dealing with other military craft is usually a straightforward task since there are well-defined protocols for ship-to-ship communication and negotiation of safe passage. But civilian craft may be very unpredictable, and the OOD may need to give high deference to the other craft, sometimes stopping or even reversing direction to ensure that a collision will be avoided. To simplify the challenge of navigation in Version 1.0, only static watercraft are required, mainly to enhance the realism of the ocean scene. The craft should be placed well away from the channel areas so that they will not constitute even a minor worry to the trainee. Later versions may include pre-scripted paths for other watercraft, with a small range of possible encoded verbal interactions for simulating negotiation of safe passages.

2.3 Interaction: Logical Interface

As described above, the task of navigating the harbor does not fall solely on the OOD's shoulders. The OOD's commands are actually carried out by a helmsman operating below deck and he receives advice from a navigation team, also below deck, who have access to a more extensive array of charts on which they use triangulation methods to plot the submarine's position and relay it to the OOD. Other information provided by the navigator includes approximate distance from the centerline, distance from the next turn, and a "yellow warning" given when the sub approaches the channel boundary. In some instances

a navigator may even recommend a specific command, possibly to correct a misalignment with the centerline. However, the OOD has the authority to give steering commands because he has a full unimpeded view of the harbor and the surrounding water traffic.

An ideal human-machine interface would model both the helmsman and the navigator, providing occasional advice from the navigator when appropriate as well as the helmsman's acknowledgments of commands received from the OOD. It would also simulate radio communication with other watercraft desiring to cross the channel, to negotiate a mutually safe passage. However, it was decided that for Version 1.0, only the communication between the OOD and the helmsman was essential, leaving the navigator and passing watercraft for later versions. This would force the trainee to learn more quickly how to use the available navaids and physical aids in the task. In addition, the model of the helmsman required in Version 1.0 is highly simplified, in that he is assumed never to make mistakes. If the OOD command is correctly recognized, the command should be acknowledged through a verbal repetition of the command followed by the words "Bridge, Helm, Aye," a shipboard etiquette indicating successful communication from the bridge (OOD) to the helm. Later versions may include more complex human modeling in which the helmsman makes occasional mistakes at random, increasing the responsibility of the OOD trainee to check that each command given is carried out as directed.

2.3.1 Speech Input and Output

The simulator must be able to recognize and respond to verbal commands from the trainee. Only a subset of the full range of possible OOD commands need be implemented for initial training studies. This subset focuses on the task of steering the submarine using three main types of commands: rudder commands, heading commands, and speed commands. A rudder command includes a direction (left or right) and an angular extent of turn. This angle varies from "standard" (15°) to "full" (30°) to "hard" (35°). Heading commands give a specific compass heading, and the helmsman's goal is to make whatever rudder adjustments are necessary to achieve that new heading as quickly as possible. Heading commands may be given individually, but are often issued immediately after a rudder command, in which case the helmsman starts with the specified sharpness from the rudder command and eases

Table 2.1: List of OOD Commands with corresponding responses from the helm.

OOD Command	Helm Response
[Helm bridge] [all] ahead one third.	All ahead one third, bridge, helm, aye. Bridge helm maneuvering answers all ahead one third.
[Helm bridge] [all] ahead two thirds.	All ahead two thirds, bridge, helm, aye. Bridge helm maneuvering answers all ahead two thirds.
[Helm bridge] [all] ahead standard.	All ahead standard, bridge helm, aye. Bridge helm maneuvering answers all ahead standard.
[Helm bridge] right standard rudder.	Right fifteen degrees rudder, bridge, helm, aye.
[Helm bridge] left standard rudder.	Left fifteen degrees rudder, bridge, helm, aye.
[Helm bridge] right full rudder.	Right full rudder, bridge, helm, aye.
[Helm bridge] left full rudder.	Left full rudder, bridge, helm, aye.
[Helm bridge] right hard rudder.	Right hard rudder, bridge, helm, aye.
[Helm bridge] left hard rudder.	Left hard rudder, bridge, helm, aye.
[Helm bridge] rudder amidships.	Rudder amidships, bridge, helm, aye.
[Helm bridge] steady on $digit $digit $digit.	Steady on $digit $digit $digit, bridge, helm, aye.
[Helm bridge] steer $digit $digit $digit.	Steer $digit $digit $digit, bridge, helm, aye.
[Helm bridge] steady.	Steady, bridge, helm, aye.

the rudder back to center position as the sub approaches the given heading. Finally, speed commands give the helmsman a desired "bell sounding," or discrete water speed to achieve. Bell soundings range from "one-third" (4 knots) to "two-thirds" (8 knots) to "standard" (12 knots). Faster speeds are possible, such as "full" (16 knots) or "flank" (20 knots), but to simplify initial training studies, Version 1.0 should only accept one-third, two-thirds, or standard speeds [WHLG94]. The left column of Table 2.1 gives the full range of OOD commands required for Version 1.0. Further voice commands may be required as part of the logical interface for switching between different views and using physical aids; these commands will be described later in Section 2.3.2.

After the system recognizes a spoken OOD command, the verbal response of the helmsman should be simulated using pre-recorded audio output. The proper response for each

of the commands is a repetition of the OOD command, followed by the words "Bridge, Helm, Aye," to indicate that the command was successfully conveyed from the bridge to the helm. This verbal repetition is necessary not only for enhanced realism but as a check that the speech recognition system recognized the command correctly. As with the real task, immediate feedback on incorrect recognitions is important so that the subject may try the command again with more careful enunciation, losing only a minimum amount of time due to the missed command. If a command is not recognized by the second or third try, an instructor's interface should offer a button-based GUI (Graphical User Interface) for entering any command, as an alternative to the voice recognition (validation of this instructor's interface is not within the scope of this thesis). Since such misrecognitions could impair the performance of a trainee through unwanted delays in turns or speed changes, stringent requirements on the recognition rates are imposed (see Section 2.4.4).

For speed commands, there is a slight variation on the verbal response. After repeating the OOD command as described above, an additional phrase, "Helm, Bridge, Maneuvering answers all ahead <SPEED>" where SPEED is the bell sounding given in the original command (either one-third, two-thirds, or full). This is to indicate that the engine order was relayed to and acknowledged by the maneuvering crew as well [WHLG94]. The right column in Table 2.1 gives the proper response from the helmsman for all allowed OOD commands in Version 1.0.

2.3.2 Physical Aids

The OOD is not expected to memorize the locations of every single buoy, range marker, and turning beacon to perform the task without any physical aids. On the contrary, navigation charts of the harbor area are available, even on the conning tower, so that the vessel's position in the overall channel can be estimated by matching up features on the chart with those in the vicinity of the sub. In addition, the OOD has a course card, a handheld paper which contains vital information about each channel segment, such as the heading of the segment, its length, the name of the turning beacon to be used, and the prescribed bearing of the beacon for beginning the next turn. Furthermore, the OOD is equipped with 10x binoculars, which are helpful in identifying numbers on buoys and are necessary to see the

range markers unless they are unusually close. A digital compass may be used to mark the bearings of range markers and turning beacons. Finally, the OOD can monitor the heading and speed of the submarine through an electronic "suitcase" that is carried above deck and plugged into an electrical junction on the conning tower.

2.3.2.1 Binoculars

In the real harbor navigation task, the OOD has a pair of 10x binoculars available to help identify distant objects that are difficult to distinguish with the naked eye. The binoculars are used sparingly, mainly to get a better look at the separation between the range markers when they are especially distant. However, they represent an important physical aid in the OOD task, and therefore the capability to bring up a similarly magnified view in the simulation is required. Due to impoverished resolution in the VE, this capability may be all the more valuable; if users cannot see objects in the VE as clearly as they would in a real life situation, providing a magnified view using virtual binoculars may be necessary just to distinguish objects with the same clarity one would expect with the naked eye in the real situation. According to the BBN training team, this extra reliance on binoculars should not have adverse effects on training. Moreover, the simulation is not required to be reproduce the feeling of grabbing a pair of real binoculars and looking through them. Any reasonable alternate method of invoking the binoculars, for example by giving a voice command, may be substituted, as long as it does not require more than a few seconds of time. However, as additional feedback that the system is in binocular viewing mode, a dark frame should be drawn all around the outer perimeter of the field of view.

2.3.2.2 Compass

Another important physical aid used by the OOD is a compass. This handheld device can be used to get a bearing on a particular object, usually a turning aid. The compass is particularly important in determining the best time to turn based on the recommended turn bearing given in the course card (see Section 2.3.2.5 below), and therefore must be included in the simulation. Again, the physical feel of the compass need not be reproduced; indeed, depicting numerical readings on a device as small as the compass may prove difficult

if not impossible with the limited resolution of the displays. Using a "heads-up display" of a fixed numerical compass reading overlaid on top of the normal OOD's view is an acceptable substitute for using a handheld compass. This fixed compass display may be invoked by a special voice command if desired. The numbers should be large enough to be readable, yet not so large as to obscure important objects in the scene.

2.3.2.3 Heading and Speed Display

Officers of the Deck also have available information about the submarine's current heading and water speed. The information is conveyed through an electronic "suitcase" that plugs into an electrical junction on the conning tower. The information in the suitcase reproduces the current readings of heading and speed from the actual instruments below decks. The heading is displayed numerically, to the nearest degree. The speed information tells only what bell sounding the submarine is currently using. In Version 1.0, this will be either one-third (4 knots), two-thirds (8 knots), or standard (12 knots) since those are the only allowed speeds. The simulation is not required use a virtual plug-in suitcase to depict the numerical readings for the same reasons given above in the section about the compass. Including the information in large, fixed numerical format is acceptable, perhaps using a voice command to invoke the information. Using a heads-up display approach, similar to that of the suggested compass display above, is allowed only if the numbers do not obscure major portions of the view.

2.3.2.4 Last Command Display

After giving a command, the OOD often writes it down on a small piece of paper, since several minutes may pass before another command is needed. This helps improve situational awareness by offering a written record of exactly what the boat is currently supposed to be doing. Thus, the simulation is required to provide access to a display of the last command given. The command need not be handwritten by the trainee, since additional devices such as electronic writing pads or 3D wands would be necessary. Instead, the last recognized command should be automatically available for viewing in a special display mode. The command should be spelled out in words and numerals, not abbreviated using only a few

characters. It may be combined with another special display, such as the heading and speed display or the course card, as long as all elements of the combined display are readable. The types of commands to be displayed should include only the rudder, heading, and speed commands from Table 2.1. Any extra voice commands needed for changing viewing modes or using physical aids should not be displayed. The exact wording of the command need not be represented, as long as the command type (rudder, heading, or speed) is given along with the command value (rudder angle, compass direction, or bell sounding).

2.3.2.5 Course Card

One of the most important physical aids used by the OOD is the course card. This is a small, handheld piece of paper containing a table of information about the channel. Each row of the course card contains vital information about a particular channel segment. The first column, labeled "Course," gives the heading of the segment. If the sub is on the centerline of that segment, the OOD should maintain this heading to stay on track. The second column, labeled "Dist," gives the length of the segment in either yards, for shorter segments, or nautical miles, for longer ones (the units are labelled). The third column, labeled "Navaids," gives the name of the turning aid used for this segment. This allows the OOD to identify the turning aid on the charts. The fourth column, labeled "Range," lists either "AHD" or "AST" depending on whether the range markers for this segment are located ahead (directly in front of the sub) or astern (directly behind the sub). If the segment is short and has no associated pair of range markers, the fourth column is simply left blank. The fifth column, labeled "Turn Bg," gives the prescribed compass bearing for the turning aid that marks the time the OOD should begin the turn toward the next segment. The sixth column, "Rel Bg," serves the same purpose except that the value is in degrees relative to the heading of the segment (eg Turn Bg - Course = Rel Bg). Finally, the seventh column, labeled "New Course," lists the heading of the next segment which will be commanded in the turn of the current segment (this should be equal to Column 1 of the next row). Every channel has two separate course cards, one for inbound navigation and one for outbound navigation.

OOD Version 1.0 must provide a special view simulating the King's Bay course card.

As many rows and columns from the original King's Bay course card as possible should be included, within the limits of readability in the display. At least five rows should be included from the tables below to give subjects adequate practice with the task. Instead of being modeled as a handheld piece of paper, the course card view should cover a large, fixed rectangular area within the display. Otherwise, an additional position sensor for the hand would be required, and the resulting image jitter would make the small text difficult to read. Invoking the course card through a voice command or some alternate method is acceptable. The course card should cover most or all of the field of view to improve the legibility of the entries. Both an inbound and an outbound version must be available, depending on the starting parameters of the simulation (inbound task vs. outbound task). The course card data must be taken directly from the actual King's Bay course cards used by real OODs, unless it is shown that this data proves inappropriate for the simplifications employed in Version 1.0 such as lack of water currents. In this case, the turn bearings may be modified to help place the sub closer to the next centerline when each turn is completed. Slight adjustments to the Course column are also allowed for those segments that do not have explicitly annotated headings on the nautical charts from which the channel vertices were read. The remaining columns should be left unchanged. Tables 2.2 and 2.3 give the required entries for the inbound and outbound King's Bay course cards, respectively.

Table 2.2: Inbound course card for King's Bay.

Course	Dist	Navaids	Range	Turn Bg	Rel Bg	New Course
268	7.7 NM	BEACON "N"	AHD	323	55	268.0
294	2210 YD	LIGHT "A"	AHD/AST	16	82	302
302	350 YD	C REAR	AHD	258	311	331
331	330 YD	C FRONT	AHD	230	264	350
350	650 YD	LIGHT "2"	AHD	228	238	4
4	2050 YD	LIGHT "C"	AHD/AST	266	262	351
351	2490 YD	E FRONT	AHD	130	139	332
332	1500 YD		AST			

Table 2.3: Outbound course card for King's Bay.

Course	Dist	Navaids	Range	Turn Bg	Rel Bg	New Course
152	1540 YD	D FRONT	AHD/AST	17	225	171
171	2460 YD	LIGHT "C"	AST	243	72	184
184	2050 YD	LIGHT "2"	AHD/AST	210	26	170
170	610 YD	LIGHT "2"	AST	272	102	151
151	380 YD	C FRONT	AST	255	104	122
122	450 YD	LIGHT "A"	AST	58	296	114
114	2090 YD	BEACON "N"	AHD/AST	7	253	88
88	7.7 MI YD		AST			

As mentioned in the task analysis, it is standard procedure for an OOD to cross out a row of the course card after the turn at the end of the corresponding channel segment has been completed. The simulation must provide this capability to help the OOD keep track of where to look next on the course card. Since introducing haptic devices or a tracked stylus simulating a pencil would unjustifiably add to the complexity of the logical interface and the physical device configuration, any reasonable alternative, such as a special voice command identifying the finished row, is acceptable.

2.3.2.6 Charts

Although the most detailed charts and methods for determining position and heading are used by the navigation team below deck, the OOD has access to a summarized navigational chart that includes buoys, range markers, and turning aids. The chart also contains markings for the channel boundaries, centerlines, and water depths outside the channel. A portion of King's Bay is shown in Figure 2.8 on the next page, taken from the Defense Mapping Agency (DMA) nautical chart.

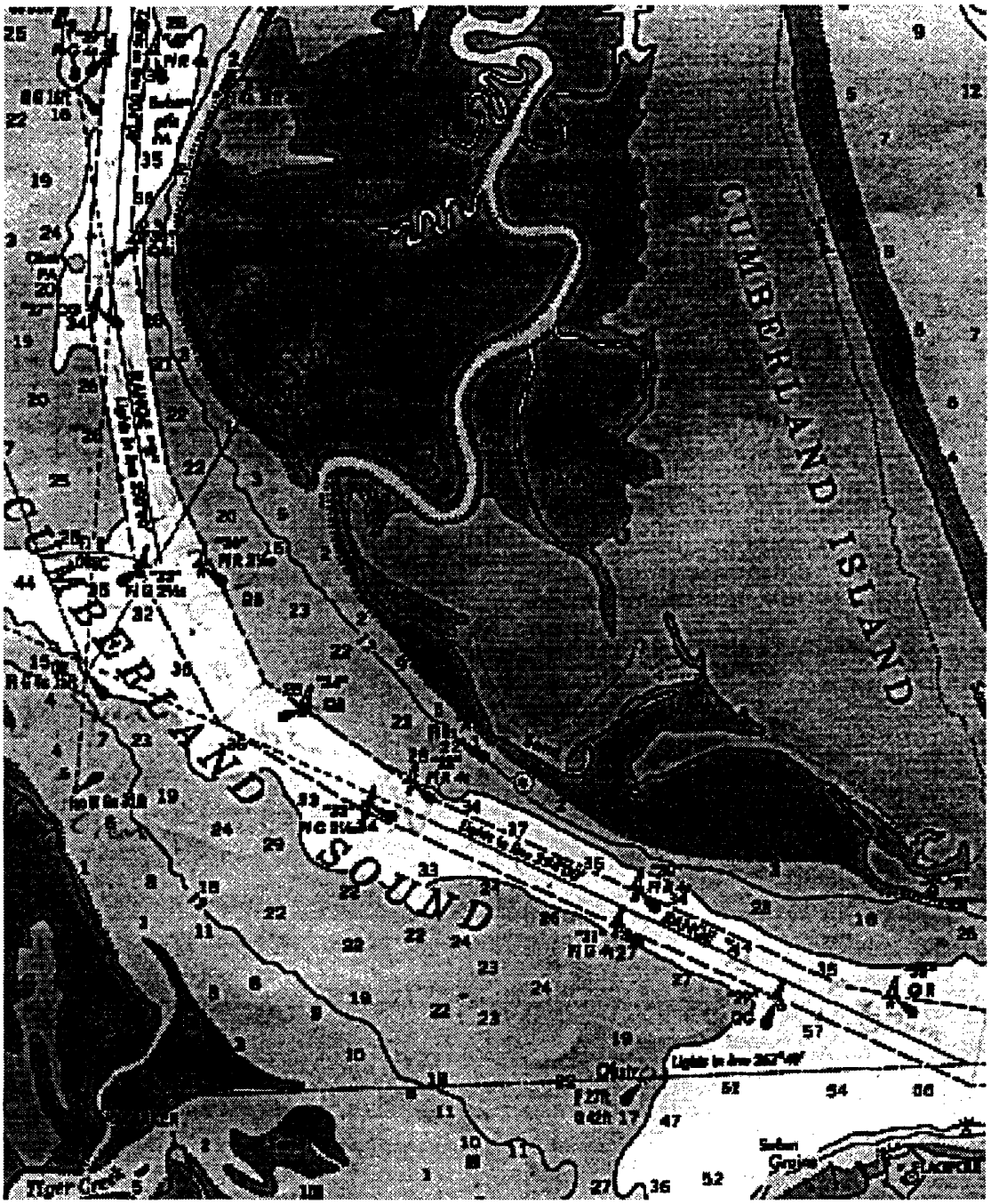


Figure 2.8 Portion of DMA nautical chart for King's Bay. Slanted triangles denote channel buoys, with buoy numbers in quotes. Dots and teardrops on the centerline extensions denote range markers, with heights appearing alongside in feet. Turning Beacon N is shown at the southeast corner of Cumberland Island.

Including all of these oceanographic features in a "virtual chart" for the simulation may prove difficult because of the limited resolution and field of view of the computer displays. However, since the navigator's advice is not included in Version 1.0, access to at least a simplified chart view is necessary for helping the trainee to estimate the sub's position and identify particular objects and their relationship to the overall course. This chart view may be invoked by a special voice command and should fill the entire field of view. The following basic chart features are required in Version 1.0:

- Demarcation of land and water, using different colors.
- All buoys along the channel boundaries.
- Locations and names of turning aids (names may be abbreviated to one or two characters, but must correspond to Column 4 of the course card).
- Either legible buoy numbers OR a dedicated voice command that causes the view to center on and highlight a particular buoy.
- Either legible heading numbers for each segment (matching Column 1 of the course card) OR a dedicated voice command that causes the view to center on and highlight a particular segment.

The following features are recommended for inclusion in the chart view but are not necessary in Version 1.0:

- Channel boundary demarcation (either by color or boundary lines).
- Centerline demarcation (for those segments with centerlines).
- Locations of all range markers.
- Water depths in feet for areas outside the channel.

All locations of features should be obtained from a valid nautical chart, preferably the same chart used to obtain locations for the environmental database. Beyond the features mentioned above, the chart view must not reveal any additional cues that could help the trainee determine the submarine's location. For example, if the chart view is implemented

as a plan view of the same graphics scene used for the submarine view, then the submarine must be deleted from the chart view. Otherwise, the extra information would make the simulated task much easier than the real world task, impairing the training usefulness of the simulator.

2.4 Presence: Physical Interface

As mentioned in Chapter 1, presence is a "qualitative measure of the number and fidelity of available sensory input and output channels," to the extent that they contribute to a feeling of immersion in the virtual world [Zel92]. This section gives the requirements of the visual and auditory sensors and displays (no haptic interactions are required for the simulated OOD task). All or most of these sensors and displays will be commercial off-the-shelf (COTS) products, since building devices from scratch would require far more manpower and time than allotted. The requirements are not rigid, since meeting every single one of them may prove impossible given the current state of the technology. Rather, they should serve as guidelines to govern a search for the best commercially available compromise, emphasizing the features most important for this particular task.

2.4.1 Graphics Rendering Platform

Perhaps the most important aspect of the simulation is the ability to render realistic, high-resolution graphics images in real-time. The extent to which this can be achieved is, of course, dependent on both the types of displays supported by the hardware platform, as well as the computational performance capabilities of the hardware and software rendering facilities. A set of suggested specifications for the graphics renderer are summarized below. They should be treated as guidelines based on currently available technology, rather than as hard requirements.

- Resolution: At least 500 x 400 full-color pixels.
- Color: RGB component color.
- Frame Rate: At least 10 Hz.

- End-to-End Latency: Less than 50 ms.
- Shading: Phong or Gouraud shading.
- Light Sources: Ambient or directional, with color.
- Object Material Properties: Ambient/Diffuse/Specular Reflection, Color Texture Maps.

The resolution of the frame buffer should be on the order of 500 x 400 full-color pixels. Anything lower may lead to difficulties in displaying readable text and symbols in the virtual chart and course card. For example, a typical row in the course card has thirty non-space characters. If we assume a minimum of ten for the pixel width of a readable character and a minimum of five pixels of space between each character, then at least 30 X 15, or 450 horizontal pixels would be required (there should also be space left on each side of the text to prevent any chance of the HMD blocking out text near the border). Unfortunately, HMDs in the desired price range do not yet match this resolution, thus the resolution requirement is stronger than the required HMD resolution below. This will allow for later inclusion of new HMDs with improved resolution.

Since color identification of buoys both in the submarine view and the virtual chart is of vital importance, The workstation's video output must support 3-component RGB color signals for best quality images in the HMD. To simulate ambient illumination of varying times of day as well as the point illumination of lights on various navaids (in future versions) the chosen graphics software platform must support ambient or directional colored light sources. If polygonal representations are used for objects in the VE, Phong or Gouraud shading should be available, in which the shading of pixels between edges or vertices is smoothly interpolated. This provides a good approximation to the shading of smooth curved surfaces (the approximation improves with the denseness of the surface polygons), which should contribute considerably to the subject's sense of presence, especially from atop the sleekly rounded attack sub. Since objects in the VE will have varying degrees of shine, dullness, and reflectivity, the graphics should support ambient, diffuse, and specular reflective components on polygonal objects. Perhaps most importantly of all, color texture mapping onto individual polygons or groups of polygons from color image files should be supported. This will allow portions of video images to be "pasted" onto objects, often lead-

ing to much more realistic appearances. These color textures also help provide perspective depth cues since the apparent size of a repeated texture will decrease with distance from the viewpoint.

These capabilities contribute greatly to the apparent realism of the rendered scene. However, care should be taken to keep the complexity of the scene (in terms of number of polygons, light sources, texture maps, etc.) low enough to achieve a minimum frame rate of 10 Hz., which is a common rule-of-thumb in many computer graphics circles. Moreover, due to the cognitive nature of the OOD task and the low bandwidth of the verbal communication involved, this relatively low frame rate should be adequate. On the other hand, the end-to-end latency of the overall graphics pipeline (time between the onset of a position sensor's movement and the corresponding change in the display) should not exceed 50 ms. Lags greater than this could cause a psychological dissociation of head movement with view control, a "sensory rearrangement" that could hamper training effectiveness and may even cause simulator sickness [Oma91]. However, the graphics pipeline latency for an update of the submarine's position in the dynamics need only be less than 250 ms. This less stringent requirement is used here since this type of lag is not noticeable by the user and since the boat never goes fast enough for that amount of lag to noticeably affect the timing of commands. However, the update rate of the sub's position should at least match the graphics frame rate so that a new sub position is obtained each frame.

2.4.2 Head-Mounted Display

The visual display used by the trainee to view the virtual harbor environment should be a head-mounted display (HMD) with a special tracker that monitors the head position and orientation. This will allow the trainee to look in any direction and see the appropriate view in that direction. Since a real OOD is able to look all around him at any time, this type of display should provide a much greater sense of presence than a fixed large-screen display or a "cave" that tries to project a 360° spherical range of view onto three or four sharply angled walls. Some more expensive displays go a step further than low-end HMDs and actually monitor the direction of gaze by tracking the user's eye movements so that lower resolution images can be used in regions outside the eye's center of focus. This level

of tracking precision is not required for the OOD application, and may be added only when such a display has become widely and inexpensively available and its psychological effects have been more thoroughly researched. Other types of displays besides HMDs may be used during simulations to allow the instructor and others to monitor the trainee's view, but only an HMD should be used on the trainee. The suggested specifications for the HMD, based on subjective evaluations of the existing technology, are as follows:

- Display Type: Head-mounted.
- Resolution: At least 300 x 200 full-color pixels.
- Field of View: At least 50° (diagonally).
- Color: RGB component color.
- Weight: Less than 3 lbs.
- Cost: Less than $10000.

The resolution specification is close to the low end of the range available for the given cost. However, for the OOD task, a wide field of view is the most important feature, thus sacrifices in resolution and even color may be tolerated if they will improve the size of the field of view. As mentioned above, since identifying colors of navaids is vital to the OOD task, high quality RGB component color input must be available, matching the requirement on the output of the graphics rendering platform. The total weight of the headgear should be limited to 3 lbs. or less to ensure that the helmet is not too uncomfortable to wear for long periods of time, since prolonged discomfort could have adverse effects on performance and useful training. The cost requirement is also not engraved in stone, but serves as an indicator of the highest price range in which the simulator would still be sufficiently cost effective for the Navy.

2.4.3 Head Tracker

In order to determine the appropriate view of the environment at each moment, an accurate tracking device must be used on the HMD to monitor the trainee's head position and

orientation. The sensor must be small enough to be easily mountable on the HMD, and light enough so that the combined weight of the HMD and sensor does not exceed the 3 lb limit. The suggested specifications for the tracker are as follows:

- Tracking capabilities: Position and Orientation.
- Position Accuracy: 0.1 in. or less.
- Orientation Accuracy: 0.5° or less.
- Singularities: None within the normal range of viewing angles.
- Range: At least a 3 ft. radius.
- Latency: Less than 25 ms.
- Update Rate: At least 20 Hz.

Again, the above requirements are not absolute, and tradeoffs that compromise one feature to enhance another may be allowed. However, the tracker must capture the position *and* orientation of the trainee, so that the trainee can look not only in any direction, but can also move his or her viewpoint in order to see around obstacles like the periscope housing. The resolution should be in the range of 0.1 inches for position, since positional jitter outside this range would make reading text or estimating range marker separation difficult and possibly strenuous to the eyes. Resolution for orientation angles should within 0.5°, so that the whole-number readings in the compass display are accurate. Furthermore, there should be no singularities in orientation tracking, unless they are well outside of the normal range of viewing angles for the task (for example, looking straight up is not useful in this task, so a singularity in that direction is allowed). The useful range of the tracker should be a sphere at least 3 ft. in radius (or cube of comparable size) and the jitter range both in distance and solid angle must be well under the resolution requirements within this space. This range will allow trainees to comfortably move their heads and upper bodies and, for cases when the range markers lie astern, they will be able to move their viewpoint far enough from center to see around the thick periscope housing protruding from the conning tower behind them. Latency should be kept as low as possible, but in the worst case should

not exceed 25 ms. (including transmission latency). This will allow up to an additional 25 ms. of latency due to the ensuing graphics update. The actual share of latency due to graphics versus head tracker is not important as long as the overall latency is kept below 50 ms. The 20 Hz. update rate is suggested to ensure that the tracker may be polled at a rate at least as high as the graphics frame rate.

2.4.4 Speech Recognition

As mentioned in Section 2.3.1, the simulator must have the capability to recognize the English phrases given in Table 2.1. Due to the heavy reliance of the OOD on verbal commands for communication, the speech recognition system must satisfy fairly robust requirements. First, the system should be speaker-independent. This means that the system must be able to recognize native English speakers (without major speech impediments) of any sex or pitch range without the need for any pre-trial training sessions, as some speech recognition systems require in order to learn the characteristics of the new speaker. Avoiding these lengthy training periods will save a great deal of time during experiments and will make the final system much easier to use in the field. Second, the system must exhibit 90% recognition rates or better. This means that out of all acceptable commands given, the percentage of commands that are either rejected as invalid or misinterpreted as a different phrase is less than 10%. Even a three digit heading command in which the heading is misrecognized by only a single degree should count as an error here. Good recognition rates will help minimize performance problems due to lost time on commands that had to be repeated or entered manually by the instructor. Third, the maximum time needed to recognize a command must be less than two seconds, since the real helmsman simultaneously acknowledges a command and begins to carry it out almost immediately after it is received. Fourth, the system must be insensitive to any background noise in order to prevent false recognitions while the trainee was actually silent. Peripheral filtering devices such as a push-to-talk button or a directional noise-canceling microphone should be considered as add-ons should this be a problem. Finally, any additional hardware devices such as microphones that must be attached to the HMD should not exceed the 3 lb. threshold for the entire helmet apparatus and must keep the center of gravity of the apparatus near the center of

the user's head to help prevent simulator sickness.

2.4.5 Speech Output

A major aspect of the training is learning the shipboard protocol, which includes not just learning proper phrasing of OOD commands, but also learning the proper response to expect from the helmsman. Therefore, the simulation must be able to play pre-recorded speech audio files to simulate the verbal response of the helmsman. A computer speech synthesizer would suffice functionally, but is not as appropriate as pre-recorded human speech in this case because of its choppy inflection and monotonic pitch. For each major command a different audio file should be used, saying exactly what a real helmsman's response would say for that command, as given by Table 2.1. If desired, key words that are repeated in several commands, such as "Rudder" or "Ahead," may be recorded separately to save memory, as long as the sequence of audio files making up the entire response sound like a continuous, human-uttered sentence. The simulator should begin executing the command simultaneously while playing the speech output, since this is what is done in the real situation.

2.4.6 Environmental Audio

According to the naval officers we interviewed, the role of auditory cues other than spoken communication in the OOD task is minimal. Even though many buoys emit various sounds, such as ringing or whistles, the OOD typically does not use the sounds to navigate, since seeing a buoy is a much stronger positional cue than hearing it. We have chosen to include continuous ocean sounds and spatialized buoy sounds under the assumption that such sounds would contribute to the trainee's sense of presence. However, we have no empirical data or related work to support this assumption. The implementation of these sounds should be given a low priority relative to the above visual cues, comparable to the low priority of modeling cultural features on the land.

There are two main types of auditory display needed to convey the environmental sounds in the OOD task:

1. Ambient Sound - This includes mainly a continuous ocean sound simulating small

waves breaking against the submarine. No motor sound need be included, since the motor is relatively quiet at these surface speeds.

2. Positional Sounds - This includes any buoys in the vicinity of the sub that emit sounds. If possible, the direction of the sound relative to the OOD must be simulated by playing the sound at varying levels in the proper headphone. Also, the volume of the sound must vary inversely with the distance of the buoy emitting it. For Version 1.0, only the two nearest sound-emitting buoys need be simulated. The most common type of buoy sound is a ringing bell. Other buoy sounds, such as whistles or horns, occur only on special purpose buoys other than those marking the channel edges. Therefore only bell-type sounds are required in Version 1.0.

For both types of environmental audio, either computer-synthesized sounds or repeated pre-recorded audio files of the actual sound may be used, as long as they are recognizable as ocean waves and buoy sounds.

Chapter 3

Implementation

This chapter describes the implementation of the initial version (Version 1.0) of the OOD simulation. Version 1.0 serves as a "baseline" simulation in which the only the most essential elements of the task are included, with no special instructional cues beyond what would be seen in the real world. Future versions will support an "enhanced simulation" with refinements on the visual presentation of the baseline objects such as multiple levels of detail, and with artificial cues, such as highlighted channel boundaries and centerlines or intelligent tutoring [ZDA+94]. General validation techniques for these enhancements will be discussed in Chapter 5. For consistency, the chapter will be organized into sections dealing with the three dimensions of the VE taxonomy used so far, autonomy, interaction, and presence. However, a brief overview of the implementation will be given first so that the reader is aware of the modular structure of the different software components and their interfaces to the VE hardware.

3.1 Overview

The OOD simulator has been implemented using a large portion of the testbed's COTS workstations, devices, and software, including a head-mounted display, a position tracker, a Reality Engine for high-performance graphics rendering, and a speech recognition package. The testbed team has authored software modules for the submarine dynamics, graphics rendering of objects in the environmental database, inter-process communication, and drivers for the COTS hardware and software components. Additional software has been created for

an instructor's interface for entering submarine commands and an experimenter's interface for controlling conditions and recording data during experiments, but these two components will not be discussed in this thesis since they are not directly involved in the presentation of the VE to the trainee.

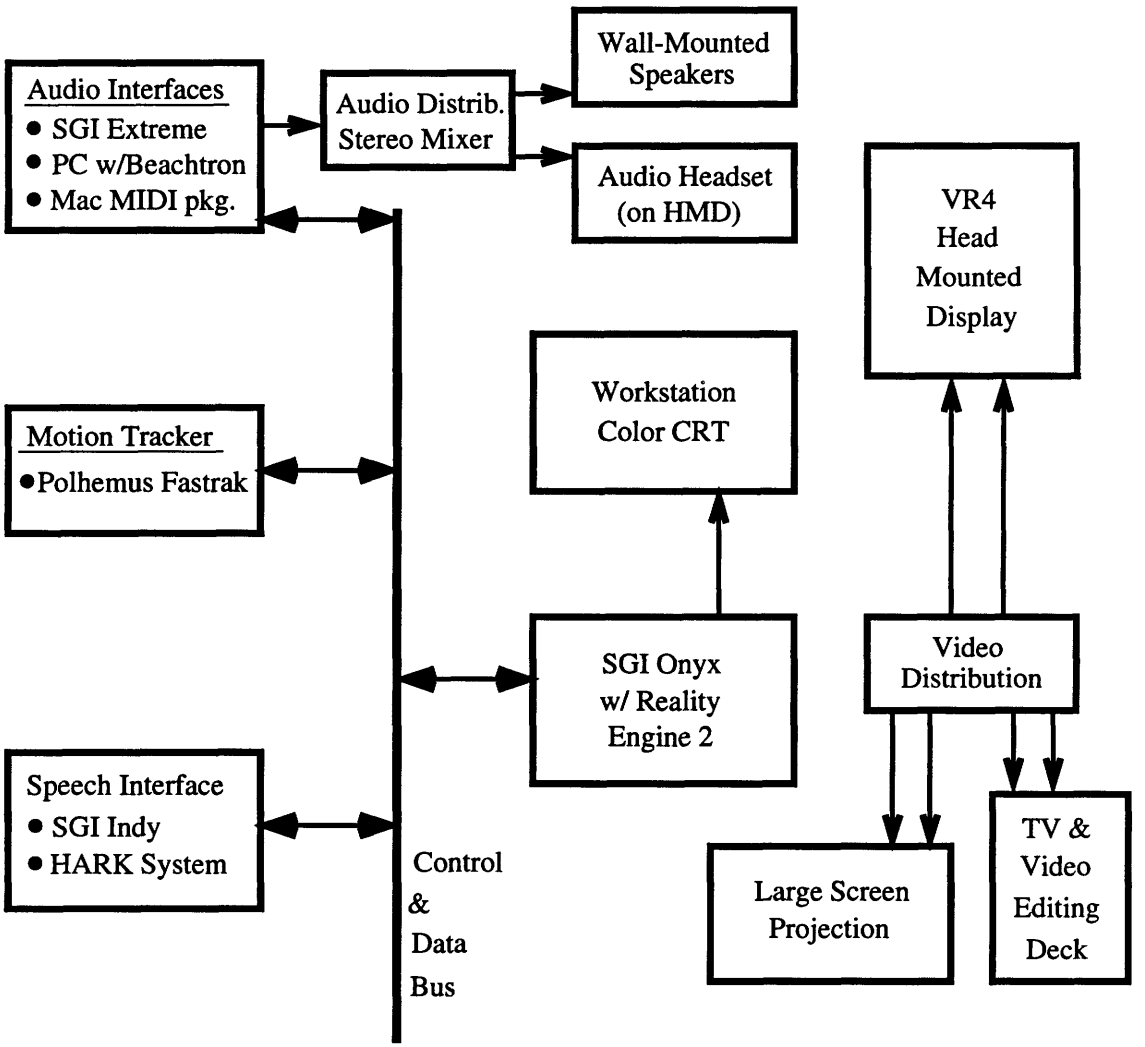


Figure 3.1 Hardware configuration for the OOD simulation. Figure adapted from VETT architecture slide by Walter Aviles.

3.1.1 Hardware

Figure 3.1 shows a block diagram of the hardware components used in the OOD simulation. Workstations and devices are distributed across two main labs: the VETT lab, which is

used as the experimental room; and the Animal lab (so named because it was once used for animal behavior experimentation) directly across the hall. The graphics is rendered by a 150-MegaHz. 2-processor Silicon Graphics Onyx workstation using a Reality Engine 2 graphics pipeline for shading, Z-buffering, anti-aliasing and texture mapping at real-time speeds. The graphics window, which constitutes one-fourth of the workstation CRT screen area, is piped to the Virtual Research VR4 helmet-mounted display (HMD), which uses LCD color pixel elements, driven by an RGB color signal. The visuals may be viewed simultaneously by the experimenter on a large screen projection display or a TV monitor connected to a video editing deck for recording raw footage of training sessions. One of the two main input devices is the Polhemus Fastrak sensor, a small device mounted on the HMD that measures head position and orientation. The Fastrak device connects to the Onyx via a serial port. The other main input device is the Sennheiser 16 KHz. directional microphone attached to the HMD, which, along with its push-to-talk button, is used for giving voice commands to the speech recognition system hosted on the SGI Indy. The Indy also handles speech output through a library of pre-recorded audio files. Audio cards on the Indy, Indigo Extreme, and MacIntosh allow any number of other pre-recorded sounds to be played back at any time. The Extreme controls the playing of these sounds through either a MacIntosh with MIDI card or sound files stored on the Extreme itself. Some of these auditory events may be spatialized by a device called the Beachtron, which is hosted on a PC compatible. Audio signals from these sources are input to an 8-channel stereo mixer whose output goes to the headphones on the HMD and optionally to a wall-mounted speaker system so the sounds can be heard by the experimenter as well. Finally, the Silicon Graphics Indy is used to run the speech recognition system and speech output.

3.1.2 Software

The two major software components of the OOD simulation are the dynamics process, implemented in C++, which effects the appropriate vehicular response based on the given command, and the graphics process, implemented using SGI's Performer and C, which renders the terrain, navaid, and watercraft databases and maintains the appropriate view based on the direction in which the trainee is looking. These two main components are supported by several secondary processes that are mostly lower level interfaces to the commercial

hardware devices or software packages. One such process is the speech recognition process, which is a waiting loop that takes in a string of recognized text from the speech recognition software and plays the appropriate pre-recorded audio output file to simulate the helmsman's response. In addition, this process encodes the command numerically and sends it across the network to either the dynamics process (for submarine steering commands) or the graphics process (for commands directly affecting the viewing mode). Another task-specific component is the process that spatializes the sounds of the two nearest buoys, given their locations and the location and orientation of the trainee's head. The other auxiliary processes are not specific to the OOD task but are very useful for multi-modal distributed VE systems in general. One such process is the "blackboard," which maintains on one workstation a global read/write database that is available to all processes on any other machine on the subnet [Nyg94]. This blackboard paradigm greatly simplifies the communication of data among the different processes. The other task-general component is a "sound server" process that sends sound information to a commercial MIDI package and determines whether the sound should be ambient or spatialized. Figure 3.2 provides a schematic diagram of the above software components and the type of data that is communicated among them.

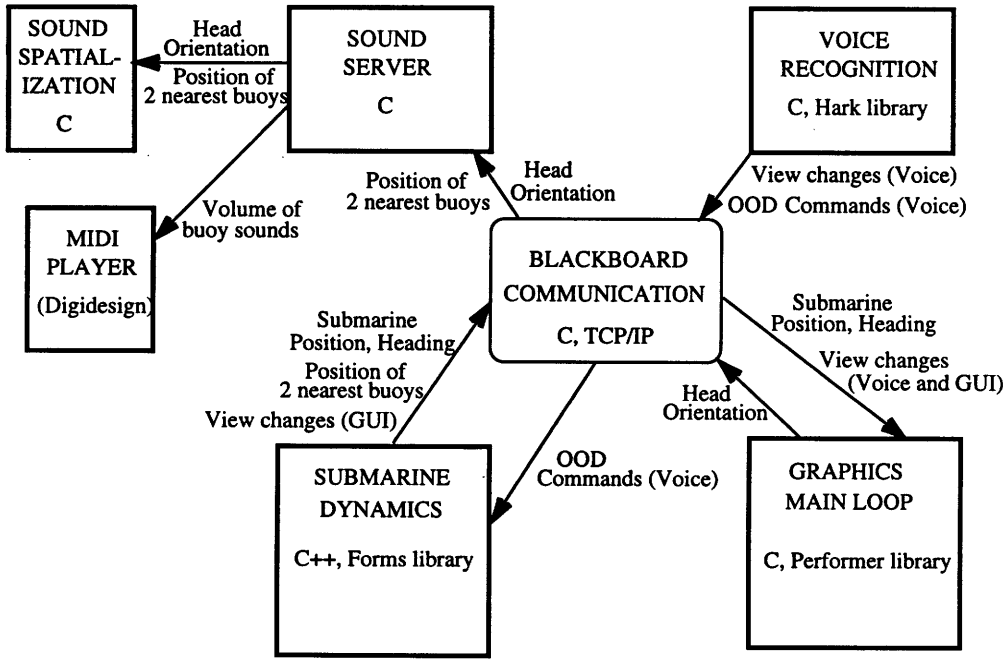


Figure 3.2 Software architecture for the OOD simulation.

3.2 Computational Models

The degree of autonomy in a VE is determined by the ability of its computational models to "act and react to simulated events and stimuli" [Zel92]. In the OOD task, most of the computational models lie on the extreme low end of the autonomy scale as they are simply static objects above the surface of the land or water, used to help the OOD navigate smoothly through the channel. The only truly autonomous entity in the VE is the submarine itself, which must accelerate or decelerate in response to speed commands and turn left or right with varying sharpness in response to steering commands. Nevertheless, even for passive objects, attention must be given to developing faithful visual representations and positional information. This section describes in detail the computational models used for Version 1.0 of the OOD simulation, beginning with the environmental database describing the locations of objects, and ending with a description of the model of the dynamics of the submarine. Illustrations of the visual representations of objects in the environmental database will also be included.

3.2.1 Environmental Database

The environmental database is actually a collection of several smaller databases, each initialized from a separate input file. Each database is used for a particular type of object and contains the X-Y locations of the objects, along with other relevant attributes, such as scaling or orientation. The data source used here was a set of nautical charts from the U.S. Defense Mapping Agency (DMA) showing annotated information next to buoys, range markers, turning aids, land masses, channel segments and centerlines.

The origin of all databases is placed at 30° 40′ North latitude, 81° 30′ West longitude (see Figure B.1 in Appendix B). This ensures that the X-Y coordinates of most objects are positive. Location descriptions using X-Y coordinates in yards were chosen because yards are commonly used in the U.S. Navy. However, this led to the need for a conversion formula between lat/long and X-Y in yards, since object locations read directly from the DMA charts are in lat/long only. Such a formula was easily derived from the scale bars for yards and lat/long units on the side of the the nautical charts.

3.2.1.1 Channel Segments and Centerlines

Chapter 2 mentioned the need for a polygonal representation of the segmented harbor channel. One conventional computer graphics polygonal representation is a numbered list of vertices followed by a list of polygons, each specified by a set of vertex numbers in counterclockwise order. Although appropriate for the general case, this representation fails to take advantage of the thin, parallel nature of the two sides of the channel. Thus, a more convenient format was chosen involving pairs of left-right vertices marking endpoints of a segment on the left and right sides of the channel. If there is no corresponding left (right) vertex for a given right (left) vertex in the actual channel, then the previous left (right) vertex may be reused if it is not too far away (Figure 3.3). This pairwise segmentation method also matches more closely the red and green opposing pairs of buoys used in the real channel. For certain turns, a small extra segment is used to separate the bordering straight segments from the turning area. Figure B.1 in the Appendix B illustrates the segmentation of the King's Bay channel into eleven numbered regions. In the input file, after an initial line containing the key word "CHANNEL BOUNDARIES" and the total number of segments, each line contains the X-Y position of the left vertex in the pair followed by that of the right vertex (in the inbound direction). Wherever possible, the vertices in each pair are derived from the lat/long coordinates of a bend in the channel on the DMA chart, with some extra vertices to fill in spots where a bend occurs only on one side of the channel. The channel segment data may be found in Appendix B, along with the above diagram of the channel with locations of the vertices and the database origin highlighted.

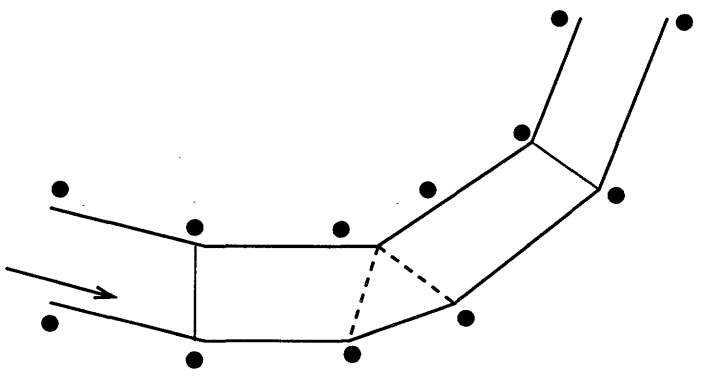


Figure 3.3 Reuse of a channel vertex for more than one vertex pair. Dots denote buoys and the arrow denotes inbound direction.

Centerlines are line segments running along the major axis of the channel segment, stopping on either end, more than 200 yds. before it would cross the boundaries of the segment. Because of this, centerlines are only useful for channel segments at least 500 yds. in length. In particular, segments 1, 7, and 9 above lack centerlines because they are all well shorter than 400 yds. (in fact, they are all turn segments, not course segments). In the input file, after specifying the channel segments, a line containing the key word "CENTERLINES" followed by the number of segments using centerlines is expected. Each subsequent line contains initially an integer index referring to the segment in the previous list containing this centerline (starting at 0), followed by the X-Y positions of the centerline startpoint and endpoint (in the inbound direction). Figure B.1 also gives centerlines for those segments that have them, while the first page of Appendix B contains the actual centerline data.

Note that in Version 1.0, no visual representation of the channel and centerlines is needed, except possibly in the view of the virtual chart. In fact, it is imperative that such visual cues be hidden, or else the subject may learn incorrect techniques that rely more on these artificial cues than on objects actually used in the real task (instructional cues of this type will be explored carefully in later versions). However, an internal representation of the channel and centerlines is necessary for purposes of quantifying performance of a path through the channel, and for determining when to resort to checking of water depths outside the channel (depths inside the channel are uniformly safe).

3.2.1.2 Navaids

The next database holds the descriptions of the three main types of navigational aids in the VE: buoys, range markers, and turning aids. Each type of navaid has a single visual representation; the database reads in locations from an input file, and the graphics process creates at these locations multiple instances of this visual representation, drawn at varying scales or orientations. For each type of navaid a brief description and illustration of the visual representation is given first, followed by a description of the input file format for specifying the multiple instances.

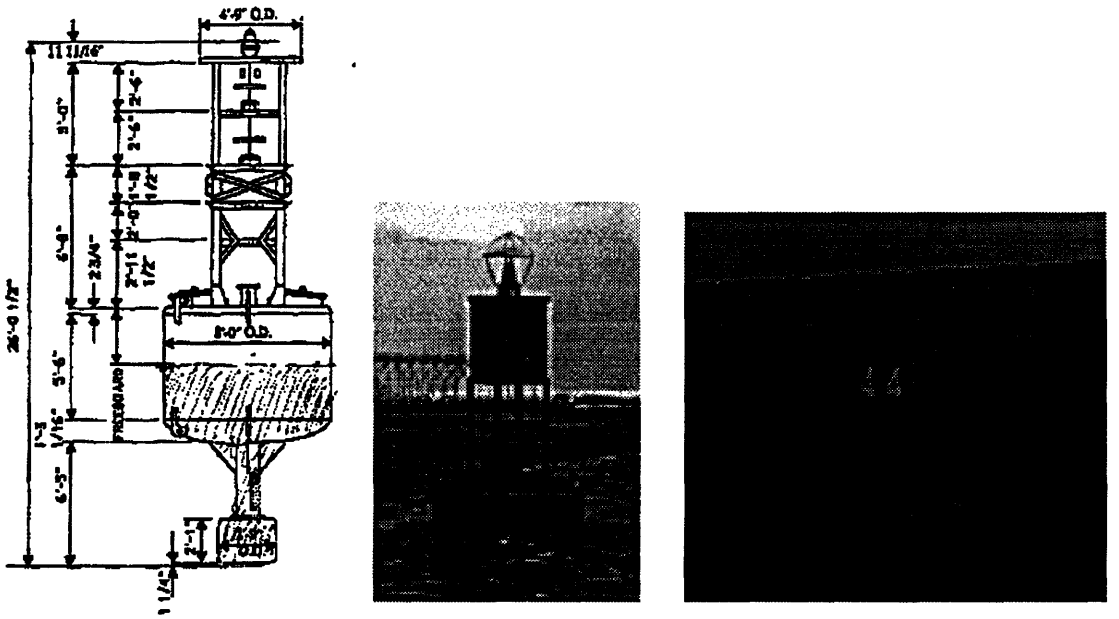


Figure 3.4
a) 8x26LR pillar buoy. **b)** Channel buoy in King's Bay. **c)** Channel buoy in VE.

3.2.1.2.1 Buoys

Figure 3.4a gives a schematic of one of the two main types of channel buoys in King's Bay. Figure 3.4b shows a photograph of this buoy as it appears in the real bay, while Figure 3.4c shows the visual representation of a buoy in the VE. Even though the heights of buoys in King's Bay come in two different sizes, either 11 ft. or 16 ft. above the surface, the buoys in the VE are all the same base size. This is mainly a prevention measure against size differences being misinterpreted as distance differences, since the biocular HMD display lacks the normal stereoscopic depth cues afforded by the real environment. Judging locations and relative distances of buoys is more important than recognizing their size differences, further justifying this simplification. Also, rather than displaying the buoy number on only one side, with 1 ft. numbers, the numbers are drawn enlarged and on all four sides, filling up most of the upper faces of the buoy. This will help meet the requirement that the nearest buoy number be legible at any point in the channel. As in the real world, the numbers are always white, but the buoy may be colored red, green, or yellow.

Each buoy appearing on the DMA chart needs to be drawn in the VE at the correct X-Y location, with the correct color and number. The navaid input file, therefore, uses a format for buoys that begins with the word "BUOY" followed by a color (either "red," "green,"

or "yellow"), then a number, and then the X and Y coordinates in yards as read originally from the DMA chart. Appendix C contains the navaid input file, which begins with a list of the buoys in the VE.

3.2.1.2.2 Range Markers

The visual representation of range markers is somewhat simplified in comparison to their actual appearance. The tower is drawn as a four-pronged support with a tall, slender pentahedral shape (Figure 3.5b). Atop this support is a flat, rectangular dayboard colored brightly orange. In reality, supports vary greatly in shape, and sometimes even elaborate tethered lightships are used [Mal77]. The real dayboards, 4 ft. wide by 8 ft. high, actually have three stripes – two outer red-orange stripes surrounding a white stripe (Figure 3.5a). Since range markers are usually located further away than any of the other types of navaids, these details would be lost in displays of low to medium resolution. The important feature for the task is the alignment of the two dayboards on a pair of range markers and this feature is best served by the simplified representation above.

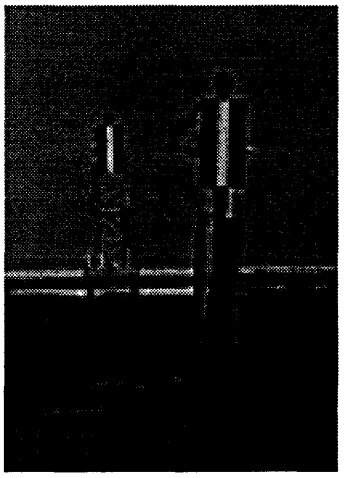


Figure 3.5a) Range markers in King's Bay.

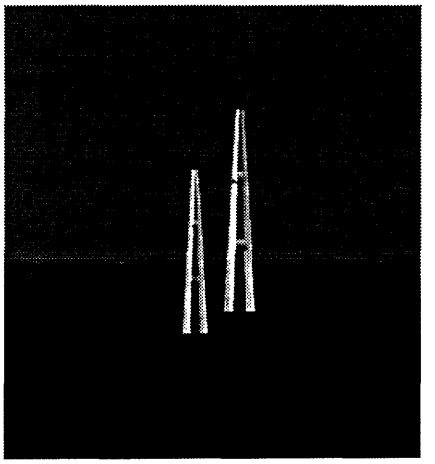


Figure 3.5b) Range markers in the VE.

The input file format for range markers differs slightly from buoys in that range markers require extra parameters to specify their heights and their orientations (since dayboards are very thin and must be oriented perpendicular to the centerline). No specification of color is given, since all range markers in the VE have the same color supports and dayboards. Therefore, the format for a range marker consists of the word "RANGE" followed by the

height in yards, followed by X-Y coordinates in yards, and finally a Z-rotation in degrees. This Z-rotation should equal the heading of the segment with which the range marker is aligned, forcing the flat dayboard to face the segment's centerline perpendicularly. See the navaid input file in Appendix C for the full list of range markers in the VE.

Figure 3.6
a) Beacon N in real bay. **b)** Beacon A in real bay. **c)** Light 2 in real bay. **d)** Beacon N in VE.

3.2.1.2.3 Turning Aids

Turning aids are the least abundant navaid in the database but are probably the most important functionally. Four of the eight turning aids are range markers, which have already been described above. The other four are two-tone, diamond-shaped or triangular daybeacons of varying height (Figure 3.6a,b,c). The representation of the diamond-shaped red and white turning beacon is quite faithful to reality as shown in Figure 3.6d. There is a single long, thin, cylindrical support, on which a flat, white, diamond-shaped board rests, with the top and bottom portions of the diamond colored red. The only missing element is the white identification letter in the top red portion, which is not included in the simulation because the available area for drawing the number would render it illegible from the channel, even using binoculars. Besides, turn beacons of this kind are sparse in King's Bay compared to the buoys, so subjects should more easily be able to identify them from the virtual chart (see Section 3.3.8). Even though two of the turning beacons in King's Bay are of the black and white type in Figure 3.6b and another is of the triangular type in Figure 3.6c, the

single red and white visual representation below is used for all turning beacons that are not range markers, mainly because detailed photographs of these other types of beacons were not available until recently.

The description of a daybeacon in the input file requires exactly the same parameters as that of a range marker, since the heights may vary, and the diamond portion is flat and needs to face the point in the channel at which the submarine should begin its turn toward the next segment. This format begins with the word "NR," which is the code name for the red and white type turning beacon, followed by the height in yards, the X-Y location in yards, and finally the proper Z-rotation for facing the segment turning point. The end of the navaid input file in Appendix C lists the four turning beacons of this kind in the VE.

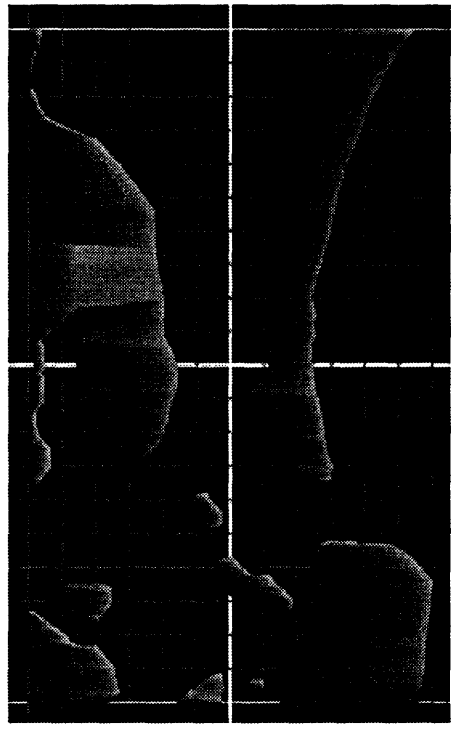


Figure 3.7 Polygonal land representation used in VE.

3.2.1.3 Land and Water Appearance

The visual representation of the land surrounding King's Bay is derived from a downloaded portion of a database on a World-Wide Web site maintained by US Geological Survey. The

data describes a 1° square area containing elevations above sea level in meters at a resolution of 3 seconds of arc (approximately 300 ft.) per datapoint. This data was converted from its native format into a form readable by our native graphics rendering system, *3d*, and later into a format readable by SGI's Performer. After this conversion, the result was a square area of sea and land that contained 57000 square, Gouraud-shaded polygons. From this large area, a much smaller square centering on King's Bay was retained (Figure 3.7), and from this new region the ocean polygons were clipped and many of the central land polygons were combined into large single polygons, since King's Bay is largely flat. This cut the number of polygons in the land down to 3500, or about 7000 after tesselation into triangles by Performer, allowing ample room for detail in the watercraft, navaids, and other objects in the scene without hampering performance.

To add perspective depth cues to the appearance of the land, a repeated texture map is applied, from a picture of the real King's Bay treeline taken from a distance. In addition, actual trees are drawn at various points on the land, using a rotating "billboard" technique of mapping a picture onto a single flat polygon that rotates to face the viewer.

The water surface is implemented as a flat, rectangular polygon covering the entire square region at a Z coordinate of zero. The land is drawn on top of this base surface, rather than fitting the shape of the ocean to match the boundaries of the land (as in a jigsaw puzzle). At first, this resulted in an some screen areas showing intermittent flickering between land and sea from far away viewpoints. This is actually a common graphics problem caused by inadequate precision in the Z-buffer depth comparisons, leading the frame buffer to oscillate between displaying land or sea at those pixels where the two objects are very nearly the same depth compared to the viewing distance. This problem was solved by a simple "decal" technique which gives priority to the land so that it is always drawn on top of the water regardless of the depth comparison in the Z-buffer. The computational costs of this solution are negligible and the gain in rendering time and generality (this ocean object may be used underneath any land object) from using a simple single-polygon ocean make this an attractive visual representation.

A static texture map (taken from a Performer demonstration of ocean waves) is applied to the water surface, giving it the appearance of rippled waves. The considerable computational expense of fluid dynamics make simulating moving waves above the surface a

performance-inhibiting option.

3.2.1.4 Water Depths

Another important database contains the depths of the bay at various points outside the harbor channel. Such a database is necessary if the simulation is to be able to discern when the submarine has run aground. These depths are taken directly from the DMA nautical chart. Each numerical depth is entered into an input file containing the X-Y location in yards of the datapoint, followed by the depth in feet as shown on the chart. Approximately 1200 points have been entered into the database, some of which are on the border of the channel to ensure that the sub does not run aground immediately after leaving the channel. The average resolution is three arc-seconds per datapoint, meeting the requirement given in Section 2.2.6. This resolution is maintained for the entire database, including areas up to 1000 yds. away from the channel boundaries. Appendix D lists a small portion of the depth entries from the input file. For all points inside the channel, a uniform depth of 45 ft. is returned without bothering to consult the depth database since all points in the part of the channel being simulated are known to be safe. Whenever the sub ventures from the channel and reaches an area of depth less than 28 ft. the simulation stops motion of the submarine, shakes the viewpoint several times back and forth, and plays the following message:

> "The submarine has run aground, sir. It has been towed to the edge of the channel and the engines have been shut off. Give speed and steering commands when you are ready."

The simulation then resumes with speed set to zero at this new start point close to where the submarine originally left the channel. This leaves time for the trainee to get his bearings from surrounding objects, consult the chart and course card, and plan a better course of action.

3.2.1.5 Watercraft

Version 1.0 is required only to include static watercraft in addition to the moving submarine. This defers the burden of creating pre-scripted paths or autonomous control mechanisms for other watercraft besides the sub till later versions of the simulator. To ensure that

the simulator will be interoperable with the latest military simulations, hooks using DIS (Distributed Interactive Simulation) communication protocols have been put in place for autonomous or external control of these watercraft [LS+93, Ins93]. The chief remaining action item for Version 1.0, then, was obtaining detailed visual representations for the static watercraft. Instead of producing polygonal models from scratch, a commercial option was sought, and in the end, several datasets were purchased from Viewpoint II, a company that issues a wide range of low-price datasets at various levels of detail. Among the watercraft selected were a sailboat, motorboat, yacht, freighter, and zodiac power-raft. A simple watercraft database is initialized from an input file containing entries that specify the vessel-type and X-Y location for each vessel. These watercraft are drawn at these same locations for the entire duration of the simulation. All locations are well outside the vicinity of the channel, to ensure that the trainee will not be unduly distracted from the chief task of guiding the submarine through the clear channel. The input file for the watercraft database is given in Appendix E.

3.2.2 Submarine

For the simulated fast-attack class submarine, we have obtained a detailed visual representation and implemented an accurate, albeit simplified model of the vehicular dynamics.

3.2.2.1 Visual Representation

Like the static watercraft above, a commercial dataset for a Los Angeles class fast-attack submarine was purchased from Viewpoint II. Using a 3D modeling package called Medit, from Medit Productions Inc., the polygons making up the rudder were isolated for articulated movement. This articulated rudder can be rotated about a vertical axis going through the point of connection between the rudder and hull (Figure 3.8). This allows the simulation to move the rudder in response to steering commands from the trainee, providing important visual feedback that the helmsman has carried out these commands correctly.

3.2.2.2 Dynamics

Software modules for computing the dynamics of the submarine and simulating its responses have been developed by BBN scientist William Levison and MIT testbed member Rakesh Gupta. Levison provided the prototype model in C code, which was recoded by Gupta

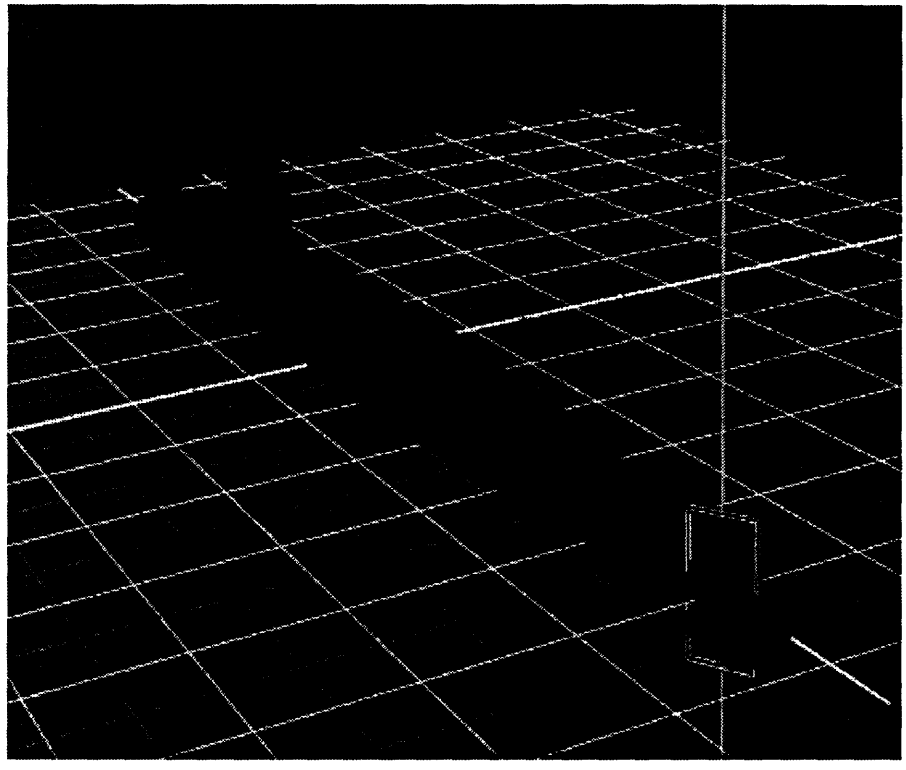


Figure 3.8 Visual representation of submarine and articulated rudder.

in C++ for OOD Version 1.0. The new implementation provides basic classes for time-steps, independent and dependent submarine parameters, channel points and segments, and methods for the automated helmsman. The submarine parameters and equations are derived in part from an earlier project that used a simple VE for visualizing a submerged submarine's path. For the OOD task any equations directly involving roll or pitch have been left out. Moreover, for Version 1.0, certain approximations have been employed, such as a turning radius depending only on rudder angle and not on speed, rudder movements occurring at the maximum slew rate, and an overall reduction of turning rate since only half of the rudder is submerged. Appendix F lists the differential equations governing the motion of the submarine, most of which have been derived from [War82]. For heading commands, the model of the automated helmsman uses a second order system with natural frequency of 1.0 radians/second and a damping factor of 2.5 to ensure the submarine will quickly stabilize on the commanded heading without significant overshoot (usually less than one

degree of overshoot for most turns). The maximum rudder setting used by the autohelm after a heading command is the maximum of the rudder angle of the most recent rudder command and 15° (standard setting) when the heading change is greater than 10°. Thus, the trainee can force a sharp turn to a specific heading by commanding a full or hard rudder and then commanding the new heading.

3.3 Logical Interface

The logical interface provides a means for the trainee to interact with the VE using natural input modalities such as speech, whole hand input, or body movements. The logical interface also includes the system's responses to the user's input, which also may be conveyed in a wide range of modalities, such as video, sound, or speech. Figure 3.9 shows a high-level diagram of the input modes and system responses used in the OOD simulation. The diagram also distinguishes between discrete types of input and output such as speech, and continuous input and output, such as head position or visual scene display. In Version 1.0 the primary input mode is speech; it is used both for commanding the helmsman with exactly the same phrases used in the real scenario and for invoking various physical aids such as the virtual chart and the course card. The Polhemus sensor on the HMD provides another input modality by monitoring the user's head position and orientation. In the submarine view, the HMD responds by displaying the visual scene as it would look from that viewpoint and viewing direction. Thus, the speech input and output constitutes an asynchronous, discrete-event feedback loop, while the position sensor and HMD display form a synchronous, continuous feedback loop. Continuous audio through the headphones on the HMD is yet another output mode, used to convey the sound of the ocean waves against the hull of the sub and the ringing of the two nearest buoys. The buoy sounds are part of another feedback loop that modulates the volume and spatial direction of the sounds according to the sub's position and the user's head orientation relative to the two nearest buoys.

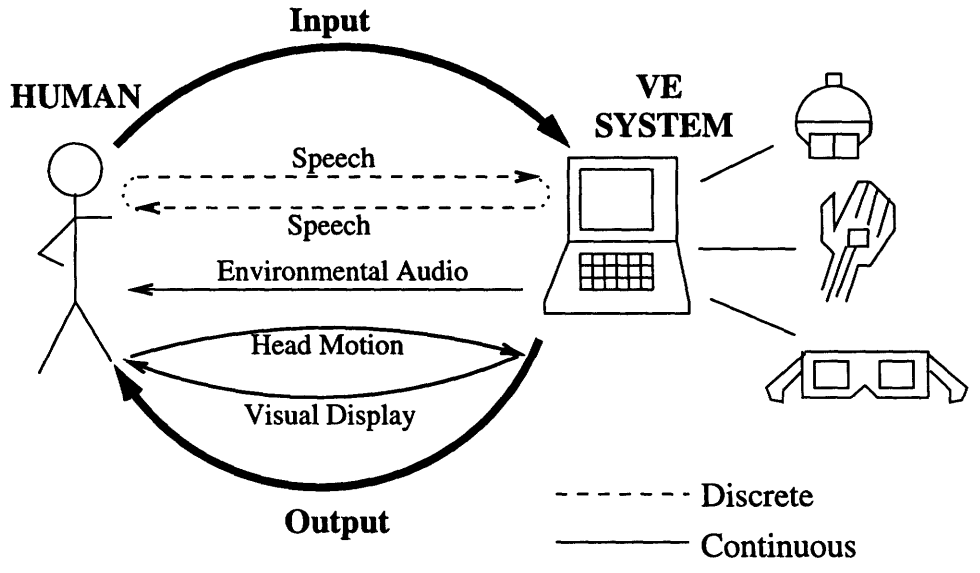


Figure 3.9 Input and output modes of the logical interface. The discrete feedback loop uses human voice commands and speech output from the VE system. The continuous feedback loop uses head movements and visual display updates. Environmental audio occurs as output only.

3.3.1 Speech Recognition

One of the most important components of the implementation is the speech recognition software. As mentioned in Section 2.4.4, recognition of speech should be independent of speaker, should achieve a 90% or better recognition rate, should recognize the command in less than two seconds, and should be insensitive to background noise. The first requirement is met by the HARK speech recognition system which was purchased from BBN for the OOD task and other VE applications. Informal tests of the HARK system in demonstration circumstances have verified this claim, as the system has worked well for a wide range of male and female users, even those with moderate foreign accents. The fourth requirement, insensitivity to background noise, is aided by the addition of a push-to-talk button on the microphone line to the audio amplifier box, effectively cutting out all input except when the trainee is giving a command. To reduce background noise during a command while the button is depressed, a Sennheiser 15 KHz directional noise-canceling headset microphone is used as the hardware voice input channel. The microphone is mounted on the left side of the HMD, while the push-to-talk button is held on the user's right hand. Each of these

requirements on the speech recognition system will be validated in detail in Chapter 4.

The HARK system uses a built-in dictionary of 100,000 words to enable recognition of any number of application-specific phrases specified in a user-defined grammar file. This grammar file's format is similar to Backus-Naur form (BNF) which uses production rules to produce correct statements [Pre93]. These rules are encoded in top-down fashion using combinations of nonterminals (lower-case terms representing a group of words or phrases) and terminals (actual spoken words). The grammar file for OOD Version 1.0 is as follows:

```
<DICT> "ood.hd";
<START> $oodsentence;
$oodsentence: $navcommand | $syscommand;
$navcommand: [ HELM ] [ BRIDGE ] $commandtype;
$commandtype: $rudder | $heading | $speed;
$syscommand: CHART VIEW | SUB VIEW | ZOOM | $binoccmd | $compasscmd |
        COURSE CARD | $buoycommand | $channelcommand | EXIT PROGRAM;
$binoccmd: BINOCULARS/binocs | BINOCS/binocs;
$compasscmd: COMPASS/comp;
$buoycommand: CHART BUOY $digit $digit;
$channelcommand: CHART LEG $digit $digit $digit | CROSS OUT $digit $digit $digit;
$rudder: $direction $angleextent RUDDER | RUDDER AMIDSHIPS/0;
$direction: RIGHT | LEFT;
$angleextent: FIFTEEN/15 DEGREES | STANDARD/15 | FULL/30 | HARD/35;
$heading: $motion $digit $digit $digit | STEADY/-1;
$motion: STEER [ COURSE ] | STEADY ON [ COURSE ];
$speed: [ ALL ] AHEAD $power;
$power: ONE THIRD/4 | TWO THIRDS/8 | STANDARD/12;
$digit: ZERO/0 | ONE/1 | TWO/2 | THREE/3 | FOUR/4 | FIVE/5 | SIX/6 |
        SEVEN/7 | EIGHT/8 | NINE/9;
```

The first line beginning with the symbol <DICT>, indicates the filename of a dictionary file that contains the phonetic spellings of special words in the grammar not found in the built-in dictionary (in this case "binocs" is the only such word). The next line marks the start of the grammar specification which is rooted at the nonterminal *$oodsentence.* An *$oodsentence* consists of either a *$navcommand* used to steer the submarine, or a *$syscommand* used to change views or invoke physical aids. A *$navcommand* is further broken down into *$rudder*, *$heading* or *$speed.* Rudder commands consist of a *$direction* followed by an *$angleextent* followed by the terminal word "rudder." *$Direction* is a nonterminal that is either "left" or "right." *$Angleextent* is a nonterminal which is any one of the terminals

"standard," "fifteen degrees" (same as standard), "full," or "hard." Notice the numerals following the slash at the end of the terminals in the definition of *$angleextent*. These are special "tags" used to provide extra information about the command. In this case, the numbers represent the rudder angle called for by the given rudder command. Returning to our grammar, heading commands are defined as the word "steer" followed by three occurrences of *$digit*, which is a nonterminal consisting of any of the terminals "one," "two," "three," "four," "five," "six," "seven," "eight," "nine" or "zero." Each terminal in *$digit* contains the tag number equal to the integer the word is describing. Finally, *$speed* commands consist of the optional terminal "all" (optional words are enclosed by square brackets), the terminal "ahead," and the nonterminal *$velocity*, which may be either "one third," "two thirds," or "standard." The *$syscommand* nonterminal includes view changes such as "chart view," which invokes the virtual chart; "sub view," which returns to the head-tracked submarine view; "course card," which invokes the course card; "binoculars" or "binocs," which toggles the binoculars on or off; "compass," which toggles the compass on or off; "chart buoy" *$digit $digit*, which centers on the buoy whose number matches the commanded number; "chart leg" *$digit $digit $digit*, which centers on the channel segment whose heading matches the commanded number; "zoom", which toggles a zoomed view within the chart buoy and chart leg displays; and "cross out" *$digit $digit $digit*, which draws a line through the course card row whose heading matches the commanded number. The final system command is "exit program," which causes the process to print out recognition statistics for the session and exit. Many of these system commands and their effects will be described in more detail later in this section where the implementation of physical aids is discussed.

3.3.2 Speech Output

The requirements of the task call for a pre-recorded library of human responses to simulate the helmsman. Such a library of recordings is straightforward to produce using the built-in audio tools on the Silicon Graphics Indy workstation. Using a tool called "soundeditor," the author used the Sennheiser headset microphone to record his own voice into compressed digital AIFF-C files. Each audio file can be played back at any time with the operating system command "playaifc." This can be achieved in C code by using a *system* command containing the string "playaifc <FILENAME>" where <FILENAME> is the name of the

aifc file to be played. To save disk space, variable words or phrases within commands are recorded separately rather than repeating the common phrases for sets of commands that differ by only a word or two. The words and phrases in this library are listed and categorized as follows:

- **Speed responses:** *all ahead, one third, two thirds, standard*
- **Rudder responses:** *left, right, fifteen degrees rudder, full rudder, hard rudder, rudder amidships*
- **Heading responses:** *steer, steady, on*
- **Numerals for headings:** *zero, one, two, three, four, five, six, seven, eight, nine*
- **Physical Aid Acknowledgment:** *binoculars, compass, course card, cross out, chart view, sub view, zoom, buoy, leg*
- **Error Feedback:** *illegal heading, say again sir*
- **Protocol Phrases:** *bridge helm aye, bridge helm maneuvering answers*

Using phrases from this library, each response from the right column of Table 2.1 can be constructed. To convey the speech output through the earphones on the HMD, the Indigo Extreme's audio signal is run through one channel of an eight-channel stereo mixer whose main output runs to the HMD headphones (see Figure 3.1).

If the voice recognition system cannot match a command with any of those given in the grammar from the last section, the phrase "Say again, Sir," is played to let the trainee know the command went unrecognized. Note that this is a different type of error than when a command or a digit within the command is misinterpreted as another phrase or number. In this case, the verbal output corresponding to the misinterpreted phrase is played. In either case, the trainee has received feedback about the error and can immediately try again. In the specific case of a heading that is out of the 0° to 360° range or is not an appropriate value for the type of the command, the phrase "Illegal heading, Sir" is played.

3.3.3 Physical Aids

The logical interface also includes the VE's methods for viewing the chief physical aids used by the OOD, such as the binoculars, compass, course card, or nautical chart. To simplify

the logical interface, all of the physical aids below can be invoked by a voice command. Where appropriate, a second means of invocation corresponding more closely to that of the task is also implemented.

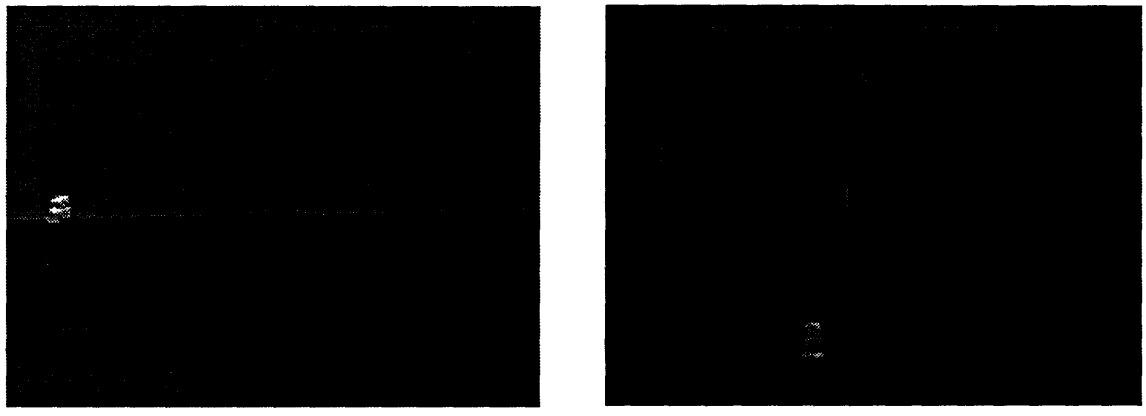


Figure 3.10a) OOD view from conning tower. **b)** Same view magnified using binoculars. The view shows Fort Clinch, a prominent landmark on the south shore of the bay.

3.3.4 Binoculars

Version 1.0 includes a voice command for invoking a magnified view that simulates real binoculars. The subject may say the word "binoculars" or "binocs" for short. The same command is used to turn the binoculars off, acting as a toggle switch. In an early prototype of the simulator, a more natural means of invoking the binoculars was attempted using whole hand input from a dataglove to detect a hand posture in which the fingers and thumb were curled into a "C" shape. This allowed the user to turn on the binoculars by raising his hands to his eyes as if holding a real pair of binoculars. Unfortunately, users occasionally turned on the binoculars inadvertently for short periods of time while moving their hands normally. The use of a voice command, though less natural than whole hand input, is a more reliable method than using the dataglove, and it helps simplify the logical interface by keeping speech as the main input channel.

Figure 3.10a shows a typical OOD view from the conning tower without binoculars, while Figure 3.10b shows the same view magnified by the binoculars. The binocular view magnifies the scene by a factor of ten. This is accomplished by using one-tenth the normal field of view, effectively 4.5°. Thus, as in real life, finding objects with binoculars may be difficult unless one first finds the object with the naked eye and then centers on the

object before using the binoculars (in this case, making sure it is inside a central rectangle one-tenth the dimensions of the entire display), to ensure that the object will be within the narrower magnified view. In early prototypes, subjects found they needed an unambiguous visual indication for binocular mode in addition to the magnification, which alone might not be noticeable in relatively featureless visual fields of regard. Therefore, a purple frame is drawn around the outer boundaries of the viewport (Figure 3.10b). Purple was chosen because it is not used anywhere else in the VE and because black would blend with the black surroundings inside the HMD, making the frame less apparent.

3.3.5 Compass

The suggestions in Section 2.3.2.2 regarding the compass have been followed closely in the implementation. By giving the voice command "compass," the trainee invokes a "heads-up display" of a three-digit number drawn at the top of the screen. The digits are black in color and are large enough to be easily legible in the HMD but still are unlikely to obscure any objects being looked at, since the top region of the screen is usually blank sky. As with the binoculars, the compass may be toggled on or off by using a voice command, in this case the single word "compass." As with onboard compasses, the virtual compass reads 0° at true north, counting up to 360° as the subject turns clockwise from north. Figure 3.10 showed an example of the compass readout when the subject is facing west and slightly south, at 257°. The compass is turned off by giving the same command a second time.

3.3.6 Course Card

The virtual course card also follows closely the requirements of the previous chapter. The course card is implemented as a large, fixed display covering the upper 75% of the field of view in the HMD. The lower quarter of the screen is used for the display of heading, speed, and last command, described below. The data in the course card is taken directly from the inbound and outbound King's Bay course cards (see Section 2.3.2.5), with the following exceptions:

- The "Rel Bg" and "Next Course" have been deleted since they contain redundant information found in other columns.

- The title of the "Course" column has been changed to "Heading" to reduce subjects' confusion of terminology during the training studies.
- The "Heading" values of the third and fourth segments have been changed from 302 to 303 and from 331 to 334 respectively, since these numbers more closely match the headings of these two segments as read from the DMA charts.
- The "Navaid" and "Turn Brng" columns have been switched so that the name of the turning aid and its associated turn bearing will appear consecutively, grouping together related information.

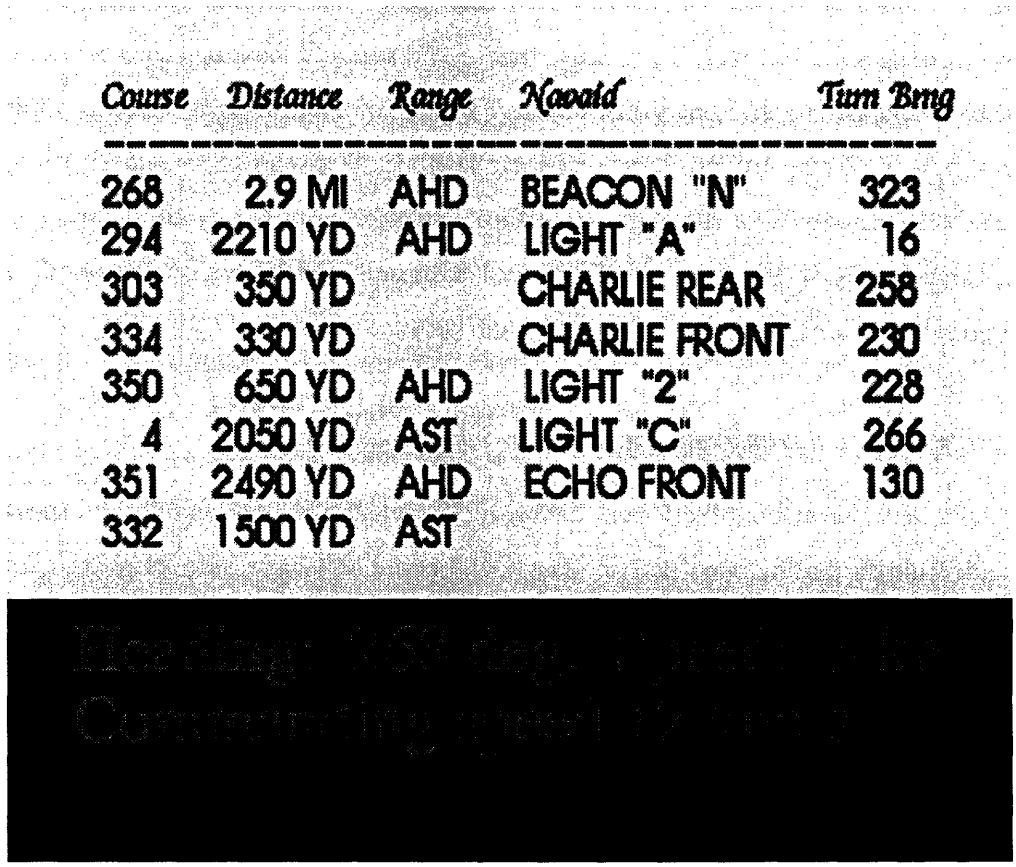


Figure 3.11 Inbound course card display. Bottom portion shows heading, speed, and last command.

The top portion of Figure 3.11 (white background) shows how the inbound course card appears in the HMD. There are two ways in which the subject can call up the course card. He may either look down past a pitch angle of 40° from the horizon, or he may say "course card" as a voice command which acts as a toggle switch, as with the binoculars and the compass. The first method is faster than the second and is a closer match to the action in the real scenario of glancing down at the hand-held card. However, since some subjects

reported that it was too distracting to have to look down for the course card, the second method was added. Since no head movement is required in this method, the trainee can keep objects in the field of view after looking at the course card, possibly preventing such disorientation.

To allow the OOD to cross out a row in the course card after navigating the corresponding channel segment, another system-specific voice command was implemented. The command consists of the phrase "cross out," followed by a three-digit heading (each digit is uttered separately). This heading must match the segment heading whose row is to be crossed out, that is, the first entry in that row. Using voice commands for such minor task elements instead of whole hand input or haptic feedback is a reasonable substitution, since it reduces the complexity of the logical interface. By offering only one input mode, namely speech, for both OOD commands and system commands, the effects on task performance of having to learn and control the subject's interface are minimized.

3.3.7 Display of Heading, Speed, and Last Command

To cut down on the number of different display modes available in the simulation, the display of the sub's heading and speed and the last OOD command have been combined with the course card display. The lower 25% of the field of view in the HMD shows this information, using a black background and a different font style and color to help distinguish this region from the course card information above (Figure 3.11). The speed is given in knots, rounded to the nearest integer rather than the nearest four-knot bell sounding, since this method might cause confusion between the sub's current speed and the last commanded speed. The last OOD command is displayed using the phrase "Now commanding" followed by either "rudder," "heading," or "speed," depending on the type of the last command. Following that is the relevant integer value of the command, either a rudder angle, a heading in degrees, or a bell sounding in knots. The display of the sub's current heading is rounded to the nearest degree. Since this is the most important of the three items, the heading is also displayed in the head-tracked submarine view, as a three-digit black numerical overlay drawn in a fixed position at the bottom of the viewing window (see Figure 3.10a). This extra heading indicator is removed in binocular mode, however, since it would overlap with the purple window frame drawn for the binoculars (see Figure 3.10b).

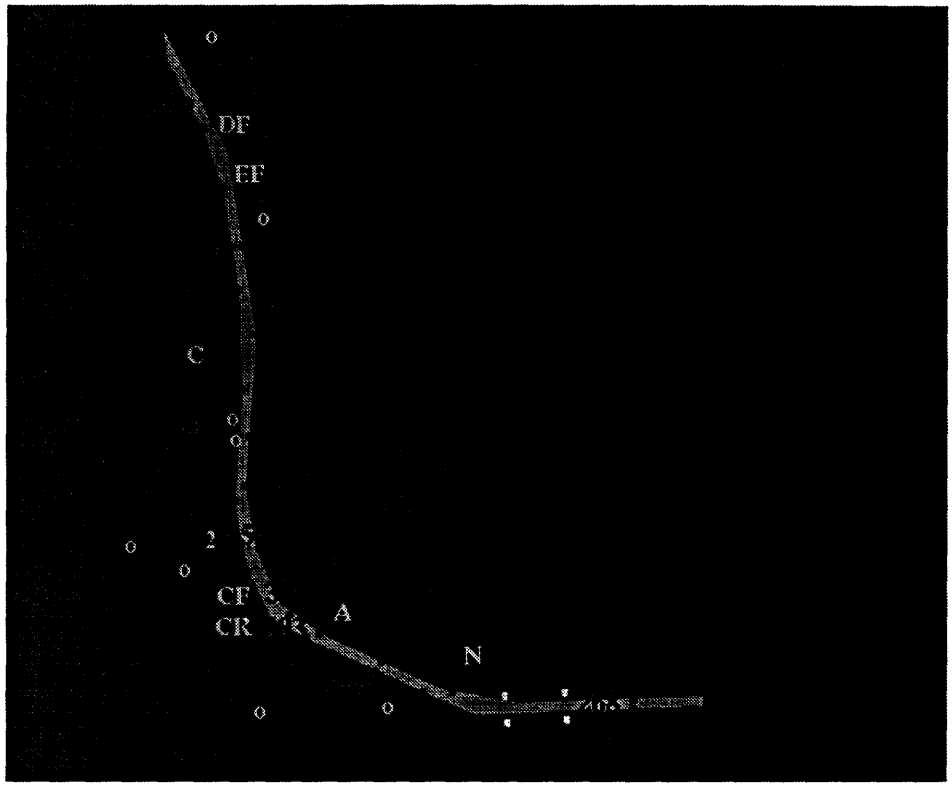


Figure 3.12 Virtual chart display. Buoys are denoted by squares along channel boundaries. Range markers are denoted by "o" symbols or by their two-letter initials if they are turning aids. Turning beacons are denoted by their one-character initial. The black numbers denote segment headings.

3.3.8 Virtual Chart

Limitations in HMD resolution and field of view make it difficult to display all of the information found on the paper DMA nautical charts in the simulation. However, all of the required items and some of the recommended items from Section 2.3.2.6 have been implemented in as clear and legible a manner as possible. Figure 3.12 illustrates the "virtual chart" of King's Bay used in Version 1.0. It is actually a plan view of the same graphics objects drawn in the submarine view, except at a much greater distance, although some of the objects, like the range markers and turning beacons are replaced by appropriate symbology. Also, the submarine is not drawn in the chart view, since this would be a strong positional cue not available in the real situation. All other objects in Version 1.0 are static, which makes feasible this dual use of the ocean objects for both submarine surroundings and simulated chart. The chart shows land regions colored green and water regions blue.

All of the buoys in the scene are scaled up from their normal size by a factor of 100 so that their tops are clearly visible as colored squares in the chart view. For each turning aid, a yellow one- or two-letter abbreviation of its name is displayed on the chart at the location of the actual object. The four turning beacons N, Light A, Light 2, and Light C are abbreviated by the characters "N" "A" "2" and "C" respectively. The four range markers that are used as turning aids, C Front, C Rear, E Front, and D Front are labelled "CF" "CR" "EF" and "DF" respectively. All other range markers are displayed as a yellow "o" in the corresponding map location. In addition, the harbor channel is delineated by a light blue color contrasting with the darker blue of the rest of the bay. The three-digit headings of each segment are drawn in black on top of the corresponding segment. If desired, the trainee may center the view on a particular channel segment by using the voice command "chart leg" followed by the three-digit heading for that leg. The trainee may then change the scale of the map by saying "zoom," which zooms in on the center of the picture so that the leg may be examined in more detail. The "zoom" command acts as a toggle switch, so saying it again brings the user back to the original chart view scale. A similar command allows the trainee to zoom in on a particular buoy. If the phrase "chart buoy" is said, followed by a two-digit buoy number, then the view will center on the buoy of that number, if such a buoy exists. After this, the "zoom" command may be used, having the same effect as above.

3.4 Sensors and Displays

Of the three salient components of VE simulation systems, presence is perhaps the most difficult to quantify since it is a subjective feeling expressed by the user of the VE. We define immersion, on the other hand, as an objective measure of the number and types of sensors and displays used in a VE. In this section, rather than attempt to derive an overall index of immersion, I will describe separately the sensors and displays in terms of the type, resolution, and accuracy of information. However, for each sensor or display, I will give a brief description of alternative sensors or displays that were considered, to help give some idea of where our implementation stands relative to the range of available technology.

3.4.1 Graphics Rendering Platform

The simulation uses a Silicon Graphics Onyx workstation to render the graphics scene, which is displayed in a 646 x 486 window on a color monitor and piped to the HMD via an RGB video connection. If desired, additional RGB cabling enables external viewing of the scene through the large-screen projection system. The Onyx has two processors running at 150 MHz. and a Reality Engine 2 graphics pipeline for Z-buffering, shading, anti-aliasing and color texture-mapping. An early prototype of the OOD simulation used the *3d* software system, originally developed under Prof. Zeltzer at the MIT Media Lab [CZ92]. The *3d* system is an interpreted scripting language for rapid prototyping of virtual worlds, built using Tool Command Language [Ous94] and compiled GL graphics calls. It provides high-level graphics operations such as multiple instances from a single polygonal input file, any affine transformation, and viewpoint control. The *3d* system provided a flexible, procedural, rapid development environment for early versions of the simulator, but as the scene became more complex the frame rate dropped as low as 6 Hz., well below the requirements. This led to a reimplementation of the graphics code using Performer, a library of special graphics commands embedded into C/C++ on Silicon Graphics machines. Although the Performer development environment operates at a lower level than *3d*, lengthening the development time, it is able to run at much faster speeds and offers additional built-in functionality such as control of time of day, fog, and other effects.

In Performer, the simulation exhibits a variable frame rate between 12 to 15 Hz, a significant improvement over *3d*. Unfortunately, the update rate of the submarine's position is noticeably lower, about 6 Hz, even though the dynamics process sends new values at 20 Hz. The cause of this loss of data is the blackboard process, which tends to repeat values on higher frequency reads exceeding 10 Hz., due to an automatic low-level buffering of socket data in the SGI's operating system. Section 5.1.2 suggests alternatives to the global blackboard process that would alleviate this bandwidth problem.

The end-to-end latency of the entire OOD graphics pipeline has not been directly measured, but it is estimated to exceed 110 ms., much greater than the required 50 ms. of Section 2.4.1. This value is based on tests run by VETT research assistant James Bandy. Bandy used timing circuits to record the time between the moment a bracket-mounted Polhemus sensor fell past an optical switch and the moment the VE scene changed in the

CRT, as measured by a light-sensing diode. The scene change was programmed to occur in the test software when the Polhemus sensor's values achieved that height. For an empty graphics scene with no additional computations beyond monitoring the Polhemus sensor's Z coordinate, the measured end-to-end delay was 43 ms. on average [Ban95]. Thus, if the average frame rate of the graphics process for the OOD is 15 Hz, then the additional latency due to the polygons in the scene and extra software computations within the frame would be 67 ms, for a total of 110 ms. This is a rough estimate since it assumes an entire frame's time in Performer should be added to Bandy's base value. In practice, a portion of the frame computations occur *before* the request for data is sent to the Fastrak, though most of the frame time is taken up by the frame render which does occur after the Fastrak request. However, from a user's standpoint, the viewpoint lag on sharp head movements is quite noticeable, indicating end-to-end latency may be well over the 110 ms. estimate. Section 5.1.1 suggests possible improvements for future versions that could help reduce lag to the required limit of 50 ms.

3.4.2 Head-Mounted Display

Even though HMD technology has evolved since the first well-known device was built by Ivan Sutherland in the 1960's, it is still quite limited compared to the quality of conventional monitors or large screen projection displays. Nevertheless, the need in the OOD task to quickly look around in any direction is suited by an HMD much better than by any fixed image projection or monitor. The HMD used in Version 1.0 is the VR4 model, built by Virtual Research for $8000 (Figure 3.13). Given the price constraints of $10000, we believe that this product represents one of the best available tradeoffs in chroma resolution, color quality, field of view, and ergonomics. The device weighs slightly over 2 lbs., well under the required limit. The device uses LCD displays at an effective resolution of 247 x 230 full-color hexagonally interlaced pixels (56,887 triads made up of 170,660 individual color elements), and a 60° diagonal field of view [Vir94]. This is slightly lower resolution than Virtual Research's previous HMD, called the Eyegen3. The Eyegen3 had 493 x 250 full-color pixel elements, (123,250 triads made up of 369,750 individual color elements) but had only a 40° diagonal field of view [Vir93]. The loss of resolution is significant, but is made up for by the improvement in field of view, which is an especially important

presence factor for the OOD task. The Eyegen3 used two miniature CRT's which produced a smoother picture than the grainy LCD pixels in the VR4. However, overall image quality is still better in the VR4 since it allows three-wire RGB component color signals, while the Eyegen3 is limited to one-wire NTSC composite signals which are color-biased for human flesh tones. Furthermore, the VR4, with its sleek, one-piece, adjustable visor that even fits over spectacles, is ergonomically more comfortable than the Eyegen3 which had two separate eyetubes that did not permit wearing spectacles.

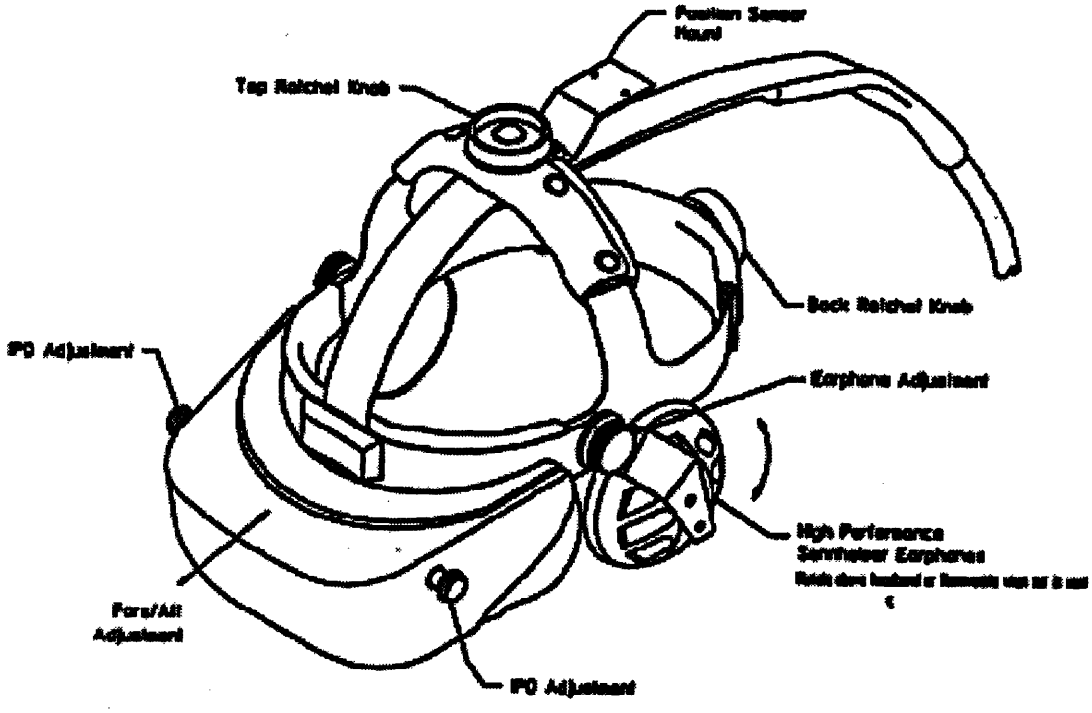


Figure 3.13 VR4 head-mounted display [Vir94].

Another helmet display evaluated by the testbed was the VIMM, built by Kaiser Electronics. This display combines four separated LCD screens to produce an effective resolution of 745 x 224 color pixels. Since the LCD screens are placed horizontally side by side, the horizontal field of view is much greater than that of most other HMD's. However, the vertical field of view is even smaller than the Eyegen3. And, though the two central screens are fully overlapping, the space between the outer screens and the overlapping region is apparent as two thick black lines, which seriously detract from one's sense of presence in the VE. Most other HMD's in this price range either failed to meet the weight criteria, as with the heavy LEEP flight helmet, or required hand-operated movement of the display,

deterring the subject from operating the push-to-talk button or feeling the physical rail, as with the Fakespace BOOM.

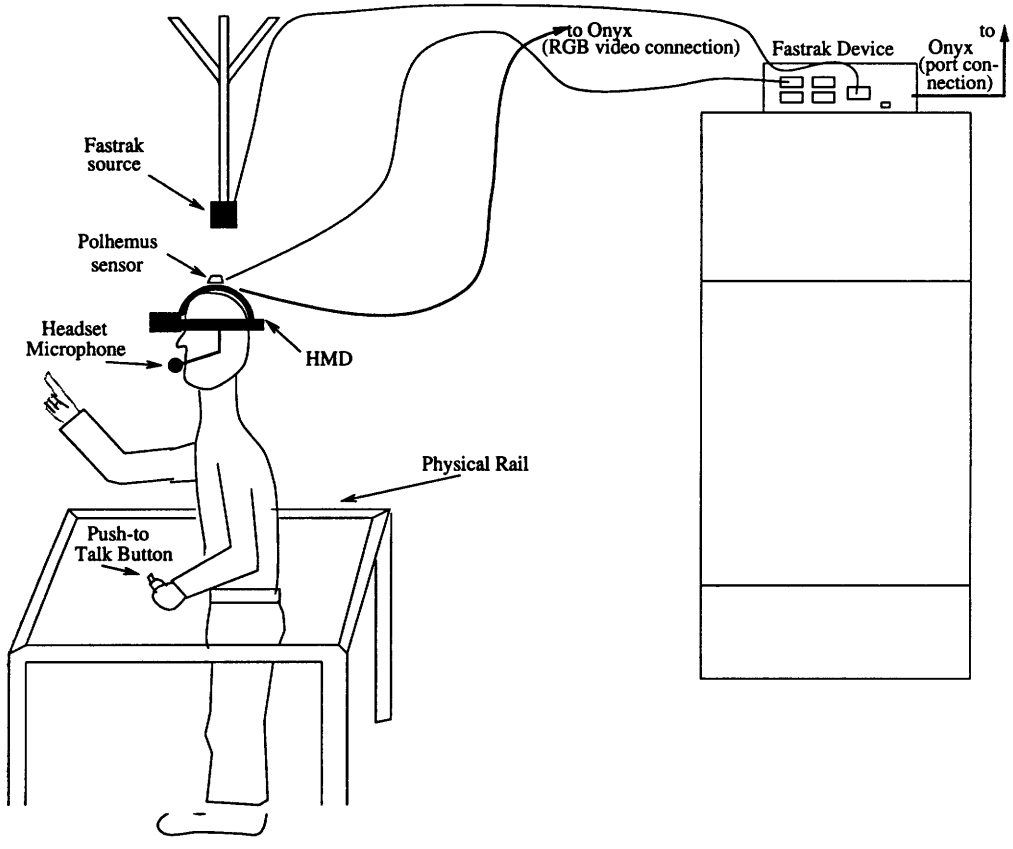


Figure 3.14 Polhemus Fastrak motion sensor. Figure also shows subject at physical rail station wearing HMD with microphone and push-to-talk button for voice commands.

3.4.3 Head Tracker

One advantage to boom-mounted displays is that the exact position and orientation of the display can always be directly measured from the configuration of the mechanical linkages. With a lightweight headworn display like the VR4, however, an external sensor must be mounted on the helmet to obtain this information. OOD Version 1.0 uses a six degree of freedom position and orientation sensor from Polhemus, called the Fastrak. This device uses a small one-inch sensor fixed via plastic screws to the top bar of the VR4. The sensor is connected to an interface box, which computes the sensor's coordinates in threespace and its Euler angles (pitch, yaw and roll) based on variations in an electromagnetic field emitted by a source that is mounted on the ceiling above the user's head (Figure 3.14). The accuracy

of the Fastrak device was measured using a calibration board by VETT team member Brett Reid, who found the data to be accurate to within .03 in. and .08°, well within the requirements for the OOD task. Reid also found that for source-to-sensor distances of less than 30 in., the jitter in the data remains within these ranges, but for distances approaching 3 ft. or more, the jitter and the spherical distortions in orientation angles become visually intolerable [Rei94]. This is not quite as large a range as hoped for, but it is larger in scope than most ultrasonic trackers such as the Logitech 3D mouse. The Polhemus sensor is less than a cubic inch in size and its weight is several ounces, thus the sensor's effect on the weight and center of gravity of the headgear is negligible. According to the manual, the latency of the Fastrak device in non-continuous polling mode (as it is used here) is 8 ms. per receiver. However, Reid measured the latency at 25 ms. per receiver, which is due mostly to transmission time of the data via serial cable from the Fastrak to the Onyx. This just meets the latency threshold requirement of Section 2.4.3. The update rate of the Fastrak has been measured at 40 Hz., from the average frame rate of a looping software driver with no graphics or additional computations. This is is well within the requirements, and well above the average graphics frame rate for the virtual ocean environment. The use of Euler angles to describe orientation leads to a singularity in the direction of positive Z, but in the OOD task and in most other VE systems, there is never a need to look straight up, so this should not constitute a problem.

An alternative position sensor, the Ascension Bird, is also supported in Version 1.0. To reconfigure the application for this sensor, a parameter in a header file needs to be changed and the graphics process must be recompiled. The Bird is less expensive than the Fastrak, but has slightly lower accuracy.

3.4.4 HARK Speech Recognition

The HARK speech recognition system, a commercial product available from BBN, is used to recognize verbal OOD commands and system commands spoken by the trainee throughout the simulation. Section 3.3.1 described the grammar that specifies the allowable commands, which has more to do with the logical interface. Here, the characteristics of the HARK system with respect to the requirements in section 2.4.4 will be discussed.

The HARK system uses Hidden Markov Models, essentially finite state machines with

probabilities on the transitions between states, to match acoustic and phonetic input to a set of standard phonemes. Then, compiled information about the restricted set of phrases encoded in the grammar file is used to pick the phrase in the grammar most closely matching the voice command. The first requirement of speaker-independence is met by the HARK system in that no pre-trial voice training is needed; the built-in HARK dictionary contains the spellings and pronunciations of 100,000 words, enabling it to compute phoneme information for a particular grammar during compilation. The recognition accuracy and time vary depending on the size and complexity of the grammar file. Except in rare cases, the HARK system has exhibited 90% or better recognition rate in demonstration circumstances. With the simple 51-word grammar file used for the OOD task, the average recognition time is about a fifth of a second, with a maximum time of about a half second. Overall time between the last spoken syllable and the first syllable of the pre-recorded response averages to about 1.5 seconds, with a maximum of 2.5 seconds, depending on the command.

3.4.5 Speech Output

The output of the speech recognition software is a string of recognized text, including any tag numbers contained in the words. Additional processing is needed to actually interpret the command into an action. In this case, there are two actions associated with each command. The first is the dynamic response of the submarine in the case of OOD commands or the view change in physical aid in the case of system commands. The second is speech playback of a verbal acknowledgment of the command. In the case of system commands, the response is always a repetition of the command. In OOD commands, the response is a repetition of the OOD command, followed by "Bridge, Helm, Aye," plus an additional acknowledgment for speed commands. All this is achieved using a small program in C that runs a HARK main loop that waits for a command that fits the OOD grammar. If a command is given, but does not match anything in the grammar, the phrase "Say again, sir" is played. If a good match occurs than the recognized text is parsed by checking for various key words, such as "Rudder" or "Ahead" or "Steer." Once the type of command and the setting of the command has been determined, the pre-recorded audio AIFF-C files that correspond to those phrases are played back, using a built-in IRIX operating system command called "playaifc." This occurs as soon as the text is recognized, usually within

2 seconds after the last syllable was uttered. Appendix G lists the C code for the voice recognition process, along with the grammar file from Section 3.3.1. The code makes use of the routine *playnumsound* which plays the audio file corresponding to the pronunciation of the given argument, which must be between 0 and 9.

3.4.6 Environmental Audio

We simulate the two major sounds in the ocean environment, ocean waves and ringing buoys, through the pair of Sennheiser headphones attached to the VR4 HMD. The audio signal to these headphones comes directly from one channel of an eight-channel stereo mixer, which may also direct the output to wall-mounted speakers near the trainee's station so that other people in the room can hear the sounds and the virtual helmsman's responses. The sounds are generated by a commercial MIDI package running on a MacIntosh, into which recordings of actual buoy ringing and ocean waves have been digitized. These MIDI events are triggered by a sound server process running on the Indigo Extreme, which plays both sounds continuously. For the buoy sounds, this sound server also sends head position data and the locations of the two nearest buoys (read over the blackboard from the graphics process) to the Beachtron device running on a dedicated PC. The beachtron also modulates the volume of each buoy according to the sub's distance from them.

3.4.7 Perceptual Cue Tuning

In a training situation, it is not always desirable to render a VE with the highest level of presence possible. For example, a VE trainer for firemen would not be very popular if it simulated the true heat of a raging inferno or the suffocation brought by smoke inhalation. Attention should be focused on the aspects of the task most important for training, accounting for safety and ergonomic factors as well. In this subsection I will rely on the notion of "selective fidelity" [Joh87]. This refers to the ability of the VE system to convey most accurately only those attributes of scene that are most important for training, leading in turn to the idea of "task presence" as a subset of the general notion of presence. For example, since most objects in the OOD task are at medium to long distances, the VE need not use stereoscopic displays, since these kinds of depth cues are only useful in small-range work volumes. Thus, we are using the HMD in biocular mode, which saves the rendering

time involved in computing a second, slightly displaced view.

Moreover, due to the relatively low resolution of the HMD many of the objects in the ocean VE must be visually tuned to more closely match their appearance in the real world. In particular, we are using a scaling algorithm on the buoys to ensure that they can be first seen at the same distance they would first be seen in the real world, which is approximately 2000 yds., called $ADIST$ below to denote actual viewing distance. Running visual tests on four subjects, I determined the average maximum distance at which a buoy in the HMD could be identified by its color, which is called $ODIST$ below to denote observed viewing distance. The scaling algorithm uses different scale factors depending on the distance of the buoy from the viewpoint. These distances are split into three different regions as follows:

- Region 1: Distance to buoy $< ODIST$
- Region 2: $ODIST <$ Distance to buoy $< ADIST$
- Region 3: Distance to buoy $> ADIST$

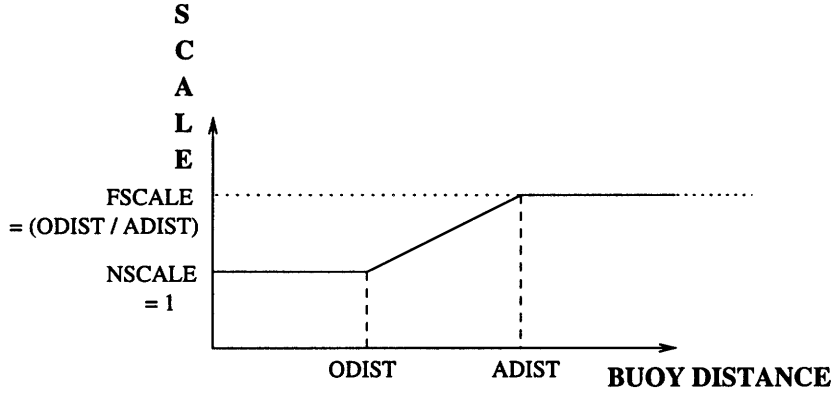


Figure 3.15 Graph of scale factor applied to buoys.

The goal here is to ensure that buoys are identifiable in the VE along with their colors to distances equal to or greater than $ADIST$ (see Section 2.2.2). In Region 1, since the buoys are already close enough to see well in the HMD, the algorithm uses a regular scale factor of one. Thus, when approaching a buoy its apparent size will increase normally after the distance has entered Region 1. In Region 2, the scaling varies linearly with the distance, beginning with a factor of one at $ODIST$, and ending at a factor of $(ADIST/ODIST)$. This linear scale factor keeps the apparent size of buoys constant in Region 2. In Region

3, the scaling is kept constant at ($ADIST/ODIST$). Thus, buoys will gradually fade away after they enter Region 3 and their distance continues to increase. Figure 3.15 shows a graph of the scale factor over its three piecewise linear regions.

Range markers are also scaled up using a simpler technique of entering their heights in the input file as twice the value shown on the DMA charts. Likewise, turning beacons have also been scaled up in the input files by a factor of two or more. Since the turning beacons and the range marker dayboards are modeled as flat rectangular surfaces, both types of objects are each rotated to perpendicularly face the associated channel segment, so that they will be most visible from the points where they are needed. Another example of perceptual cue tuning is the enlarged buoy numbers that fill the entire area of all four faces of the buoy rather than just one corner of one of the faces. The next chapter in part deals with validation that these visual cues have been tuned adequately for presenting the information in a way that offers as much "task presence" as possible.

Chapter 4

Validation and Verification

Alan M. Davis defines validation as "the process of checking the results of each stage of the software life cycle to see if it has the correct relationship to results from the previous stage" [Dav93]. Barry Boehm extends this definition to include a determination of the fitness or worth of a software product for its operational mission [Boe84]. Validation is distinguished from verification, which is the formal process of checking designs, code, test plans, and software products against requirements [Dav93]. The main focus of this thesis is on validating the results of the implementation phase with respect to the conceptual simulation model described in the requirements. Where appropriate, the validation may be supported by verification of data and algorithms in the implementation and a discussion of testing procedures used. Note that this validation does not attempt to show that the simulator can provide positive transfer of training. Such a validation would occur at a much later phase in the software cycle, after future prototypes have been designed and implemented and experiments on field usage of the simulator have been carried out.

For simulation software, formal assessment activities usually include five separate processes: conceptual model validation, software verification, operational validation, data validation, and internal security validation [KA93]. Conceptual model validation has already been dealt with in Chapter 2 when justifications for the requirements were given, with respect to the task analysis. Internal security verification, which involves testing for protection against viruses and illegal access to code, is not directly relevant to this open research effort of the OOD simulation. This thesis concentrates on the middle three processes. The

goal of software verification is to verify that the conceptual model has been implemented and tested. The goal of operational validation is to assure that the model compares well to perceived reality. Data validation deals with checking the source and consistency the data used in the computerized models. In the OOD simulator, these three processes are closely interrelated, since the requirements themselves include specifications for the data, computational models, and logical interface that attempt to match real world objects and interactions. Thus, rather than structuring the validation along these three separate processes, the taxonomy used throughout the two previous chapters will be retained, making more apparent the correspondence of validation activities to the associated components of the requirements and implementation. However, each validation activity's role with respect to software verification, operational validation, and data validation, will be discussed throughout the chapter.

4.1 Autonomy: Computational Models and Processes

Sections 2.2 and 3.2 discussed the requirements and implementation of the computational models used in the harbor VE. For coherency, descriptions of each type of object included both visual representation and locational database format. Here, I will first validate the visual representations for all types of objects and then verify their corresponding database descriptions, since the former is mainly an operational validation process and the latter is a combination of software verification and data validation processes.

4.1.1 Visual representations

Part of an operational validation involves checking that the computational models of objects in the simulation closely match their real world counterparts. Here, recent advances in computer graphics have made possible realistic rendering of complex scenes in real-time. Nevertheless, independent of the display or the graphics renderer used, level of detail available in large scenes is still limited by a rendering time proportional to the number of polygons in the scene, on top of the device and inter-process communication overhead associated with distributed multi-process VE's. Therefore, where possible, simplified visual representations of objects are used to keep the overall polygon count low enough to achieve the required

frame rate. However, care has been taken to retain those elements of objects' appearances most necessary for training. Such features have been specified in the requirements earlier in the thesis, and this section will validate that each has been adequately modeled in the visual representations while extraneous detail non-related to the task has been removed. Thus, this use of low model detail may have a second order effect of diminishing the sense of presence, but not to the extent where training is adversely affected (as it would be with a lower frame rate and longer latency). Future versions may benefit from the technique of using multiple visual representations at varying levels of detail to further enhance the sense of presence in the VE while still saving graphics rendering time.

4.1.1.1 Channel Segments and Centerlines

Version 1.0 uses no visual representation for the channel segment boundaries and centerlines, since displaying them as such would represent artificial cues appropriate only for an enhanced simulation. In the baseline simulation only real world cues are modeled, such as buoys denoting the outer edges of the channel or range marker pairs aligned with the centerlines.

4.1.1.2 Buoys

Figure 3.4 illustrated the appearance of a buoy in Version 1.0 as compared to a photograph of an actual buoy in King's Bay. The basic shape of the buoy, consisting of a rectangular base, thin bracketed supports, and a four-sided solid top with a small light attachment, matches that of the buoy in the photo. However, the four upper faces of the buoy have been enlarged relative to its height so that the buoy's color from a distance will still dominate one or more pixels in the display. Enlarged two-digit buoy numbers are texture-mapped onto each of these four faces, ensuring that at least one face will always be angled within 45° of facing the viewpoint. Using texture maps for the numbers instead of polygonal representations considerably reduces the complexity of the scene. Since exact buoy size is not an important characteristic for the task, all buoys are modeled at the same size, and are then uniformly scaled depending on distance from the viewpoint to make them more visible in the HMD (see Section 4.3.6). The most important feature, buoy color, has been modeled, using green for buoys on the left side of the channel, red for those on the right, and

yellow for the four basin markers. In the tests described below on perceptual cue tuning, subjects were also asked to identify the colors of approaching buoys as soon as they were apparent. Each subject was able to correctly identify the green and red colors at distances only slightly less than when the buoy itself first became apparent in the HMD.

4.1.1.3 Range Markers

The range markers' visual representation (see Figure 3.5b) is also simplified to be a slender white tower with flat, bright orange dayboards facing the channel centerline. Real range markers come in a variety of shapes, with some even protruding from a large tethered lightship [Mal77]. Since the common task-relevant attributes of all range markers are their brightly colored dayboards and their varying heights and locations, the use of one simplified tower-and-dayboard as a common representation is justified. The three alternating colored stripes on the dayboards have not been included since the boards would have to be scaled up beyond reasonable levels to make the stripes visible from most points on the corresponding centerline. Even these uniformly colored dayboards were initially quite difficult to distinguish in the HMD at normal scaling from the furthest points on the centerline. Using a scale factor of two for all range markers to compensate for this is justified since the visual cue of range marker alignment is a vital part of the task. Tests on human subjects to confirm the success of this perceptual cue tuning are given in Section 4.3.6.

4.1.1.4 Turning Aids

Four out of the eight turning aids in King's Bay are range markers which have just been described above. Of the other four turning aids in King's Bay, three are diamond-shaped daybeacons (two with black fields and one with red) and one is an orange-red triangular marker. Unfortunately, detailed photographs of the black and white beacons and the triangular marker were not available till after implementation of Version 1.0, in which only the red and white variety of turning beacons were used. However, since the main goal of the simulator is currently to support learning the general harbor navigation task, this simplification is justified as long as one of the valid appearances of turning beacons is used. Comparison of Figure 3.6a to Figure 3.6d shows that the VE's visual representation for this first type of turning beacon is quite faithful. Likewise, the results of the pilot experiments (see

Section 4.6) should not be affected by the simplification of using this single representation. However, if future versions of the simulation are used in transfer of training experiments involving real vs. virtual navigation of King's Bay, then accurate representations of each individual turning aid must be included.

As with the range markers, turning beacons in the VE were initially too small to identify from many of the necessary points in the channel segment. A similar solution was used, scaling the sizes of all turning beacons by a factor of two or more. Tests verifying the successful identification of these scaled beacons are described in Section 4.3.6.

4.1.1.5 Land and Water Appearance

Since King's Bay is largely low in elevation and flat in shape, rendering detailed geographic relief for the land is secondary in importance to using an accurate shoreline shape. This validates the retention of the full 3 arc-second resolution for polygons on and near the shorelines and the combining of large groups of inland polygons into uniform flat regions. The great reduction in polygon count and the resulting improvement in update rate further validate this implementation. A green "grass" texture map applied over the entire land region gives the land an appearance of vegetation similar to that seen in a videotape of King's Bay. Randomly placed rotating "billboard" polygons with pictures of trees texture mapped onto them further add to the realism of the scene and also give 2D perspective depth cues (see [DJ92, Pfa94, Nag91] for simple experiments showing the relative value of these and other types of depth cues). Similarly, the large flat polygon used for the water surface is colored light blue and continuously texture-mapped with a rippled wave pattern. Comparing the apparent size and differences in motion between ripples and in the pattern also gives perspective depth cues for location and speed much like a real ripple pattern on the ocean surface. Each of these features fulfills the requirements for land appearance in Section 2.2.5.

4.1.1.6 Submarine

As mentioned in Section 3.2.2.1, a commercial model for a Los Angeles class fast-attack submarine was purchased from Viewpoint II Datalabs. On this model the polygons in the rudder have been successfully isolated and the rudder has been rotationally articulated,

as confirmed by informal tests of the simulator in which all possible rudder commands were given and each of the seven possible angular extents were easily distinguishable to subjects. Furthermore, in the last visit by an officer from the Groton School, the height of the submarine in the water was validated as accurate compared to real harbor situations.

4.1.1.7 Other Watercraft

All of the external watercraft for Version 1.0 have also been purchased from Viewpoint II. Six different vessels covering a wide variety of sizes and classifications have been chosen. An input file is used to specify static locations for any number of these watercraft in the VE. Each location currently used is sufficiently close enough to the channel to ensure that it will be easily visible but not so close as to constitute a potential collision threat.

4.1.2 Environmental Database

This section performs data validation on the descriptions of objects stored in the environmental database. Most items in the environmental database have been derived from the DMA nautical charts of King's bay. The data was obtained initially by using the map to estimate the lat/long coordinates of each object to the nearest arc-second. Then, the lat/long values were converted to X-Y coordinates in yards using a conversion formula derived from the scale bars on the side of the chart. Finally, these values were mathematically checked for satisfying certain angular or distance criteria, depending on the type of object, and modified accordingly to meet the criteria. These criteria, along with techniques for verifying that the recorded locations satisfy them, are given below for each type of object. The comprehensive results of these tests are given in Appendix H. In most cases, automated test modules performing the calculations below could be written to speed the process of updating and validating the database for each prototype.

4.1.2.1 Channel Segments and Centerlines

Each channel segment is stored as a quadrilateral with the left and right sides parallel (see Section 3.2.1.1). Centerlines are stored as a pair of X-Y points indexed by the number of the corresponding segment. One condition that must be met by both the segments and the centerlines is that the geometric angle (with respect to true north) of the centerline and

the left and right sides of the segment must match the "Heading" entry in the course card for that segment. This can be checked mathematically by simply taking the arc-tangent of the ΔY over the ΔX for each of these three line segments, converting the result from a geometric angle to a true compass heading, and comparing it to the value in the course card. An example using the first inbound segment is given below:

Segment 0:
- Heading from Course Card: 268
- Left side:
  - $L_0 = (8417.0, 5082.0)$
  - $L_1 = (5020.0, 4951.0)$
  - $\Delta X_L = -3397.0$
  - $\Delta Y_L = -131.0$
  - $\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 182.208$ (Quadrant III)
  - True Heading $= 90 - \theta_L = -92.208 = 267.792$
- Right side:
  - $R_0 = (8417.0, 5246.0)$
  - $R_1 = (5676.0, 5148.0)$
  - $\Delta X_R = -2741.0$
  - $\Delta Y_R = -98.0$
  - $\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 182.048$ (Quadrant III)
  - True Heading $= 90 - \theta_R = -92.048 = 267.952$
- Centerline:
  - $C_0 = (8057.0, 5151.0)$
  - $C_1 = (5676.0, 5062.0)$
  - $\Delta X_C = -2381.0$
  - $\Delta Y_C = -89.0$
  - $\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 182.141$ (Quadrant III)
  - True Heading $= 90 - \theta_C = -92.141 = 267.859$

All three of the computed headings for the centerline and channel edges round to 268, which matches the heading given in the first column of the course card for Segment 1.

Another criterion for the centerlines is that they each lie along the geometric lengthwise bisector of the corresponding channel segment. This can be checked by first finding the midpoints M_0 and M_1 of the entrance line and finish line of the segment. Convert this to a line in point-slope form:

$$(y - M_{0_y}) = \frac{M_{1_y} - M_{0_y}}{M_{1_x} - M_{0_x}}(x - M_{0_x})$$

This can be rewritten to solve for y as:

$$y = \frac{M_{1_y} - M_{0_y}}{M_{1_x} - M_{0_x}}(x - M_{0_x}) + M_{0_y}$$

If the endpoints C_0 and C_1 of the centerline both satisfy this equation within a 5-yard margin, then the centerline is along the geometric bisection of the segment. For the first centerline, we check this as follows:

$M_0 = \frac{L_0+R_0}{2} = \frac{(8417.0+8417.0, 5082.0+5046.0)}{2} = (8417.0,\ 5164.0)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(5020.0+5676.0, 4951.0+5148.0)}{2} = (5348.0,\ 5049.5)$

Centerline Equation: $y = 0.0373(x - 8417.0) + 5164.0$

Test $C_0 = (8057.0,\ 5151.0)$ in equation:
$y = 0.0373(8057.0 - 8417.0) + 5164.0$
$y = 0.0373(-360.0) + 5164.0$
$y = 5150.6$

Test $C_1 = (5676.0,\ 5062.0)$ in equation:
$y = 0.0373(5676.0 - 8417.0) + 5164.0$
$y = 0.0373(-2741.0) + 5164.0$
$y = 5061.8$

The equation yields Y-coordinates for the centerline endpoints that round to those in the database, verifying the correct placement of this centerline along the segment's lengthwise bisector.

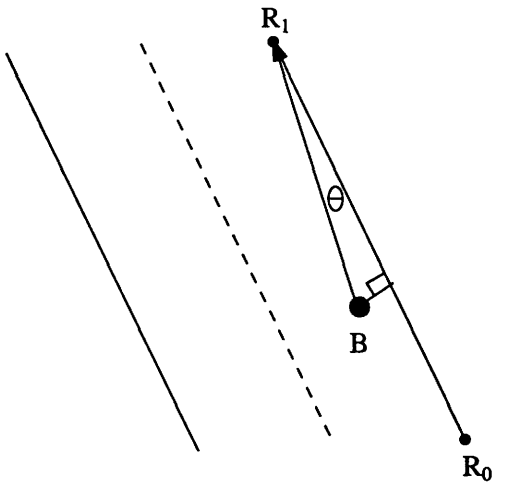
Finally, we must check that $|M_0 - C_0|$ and $|M_1 - C_1|$ are greater than 200 yds.:

$|M_0 - C_0| = \sqrt{(8417.0 - 8057.0)^2 + (5064.0 - 5151.0)^2} = \sqrt{360^2 + (-87)^2} = 370.36$
$|M_1 - C_1| = \sqrt{(5348.0 - 5676.0)^2 + (5049.5 - 5062.0)^2} = \sqrt{(-328)^2 + (-12.5)^2} = 328.23$

For the verification of the remaining channel segments and centerlines, see Appendix H.1.

4.1.2.2 Buoys

At the highest level, the function of the buoys is to mark the boundaries and vertices of the harbor channel. Since the channel segment data and buoy location data were derived independently of each other from different markings on the DMA charts, each buoy location should be checked for its proximity to the associated segment boundary. However, even the DMA map shows that not all buoys lie exactly on the artificial lines denoting boundaries. Therefore, an allowance of 20 yds. (one-eighth of the average channel width) on the perpendicular distance of each buoy from its associated channel can be used. To find this perpendicular distance we first find the angle θ between the channel boundaryline $\overline{R_0R_1}$ and the line $\overline{BR_1}$ from the buoy to the upper vertex of the boundaryline using vector dot-products:



$$\overline{BR_1} \bullet \overline{R_0R_1} = \cos\theta\,|\overline{BR_1}||\overline{R_0R_1}|$$

$$\theta = \cos^{-1}\frac{\overline{BR_1} \bullet \overline{R_0R_1}}{|\overline{BR_1}||\overline{R_0R_1}|}$$

Figure 4.1 Verification of buoy locations.

The perpendicular distance from the buoy B to the channel boundary is simply $|\overline{BR_1}|\sin\theta$. If this distance is greater than 20 yds., this datapoint must be moved along the perpendicular toward the channel boundary to the point that is 20 yds. away on that side (giving credibility to the lengthwise location of the original map datapoint). See Appendix H.2 for the comprehensive results of these buoy data validation tests.

4.1.2.3 Range Markers

It is critical that each range marker pair in the database exactly aligns with the centerline of its corresponding channel segment, otherwise risking negative transfer of training. Thus, using a technique similar to that described above for checking the centerlines themselves, each of the two range marker locations Rg_1 and Rg_2 must be substituted into the equation for the centerline. One extra computation is performed here to derive the perpendicular distance d of the range marker from the centerline extension. If this distance is less 2 yds., the markers are sufficiently aligned. Otherwise, the datapoint needs to be moved to the vector projection into the extended centerline of a vector from C_1 to the original datapoint (see Figure 4.2) and this new point should be rechecked. Refer to Appendix H.3 for results of these data validation tests on all range markers in the environmental database.

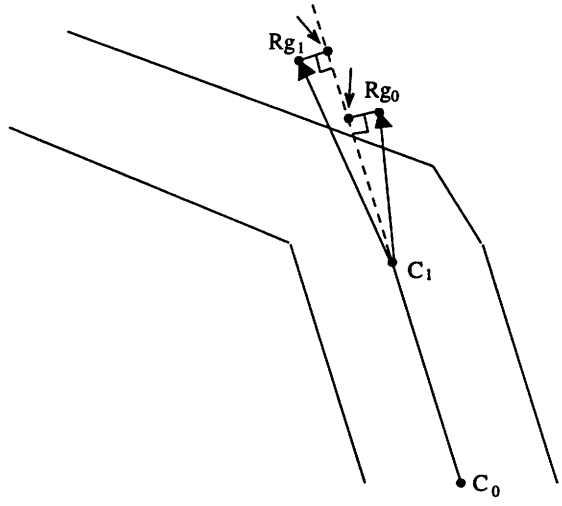


Figure 4.2 *Verification of range marker locations.* Erroneous locations Rg_0 and Rg_1 must be moved to the vector projection of vectors $\overline{C_1 Rg_0}$ and $\overline{C_1 Rg_1}$ into the extended centerline.

4.1.2.4 Turning Aids

While range markers are important for keeping the submarine on course in the interior portions of channel segments, turning aids provide vital cues for guiding the submarine's path during transitions between segments. Data validation of the locations of turning aids

must occur simultaneously with that of the fourth column of the course card, since this column contains the turning aid bearings at which turns are begun. Several methods of validating these turn bearings are described in this chapter. Here, I use geometric models of the optimal curved path of the submarine using equations from the dynamics process. Later, in Section 4.4.1, I use integration tests involving both human and computerized piloting to operationally validate these locations. The geometric data validation here is somewhat less reliable than the other methods because timing and style of turn execution varies somewhat depending on the OOD, and because simplifications in the application of the dynamics are employed. For instance, some OOD's may give a single heading command, leaving the rudder extents to the helmsman, while others may first give a rudder command followed later by a heading command. However, in cases where turning aids have been significantly mislocated, these test results should indicate the approximate direction and magnitude of the error.

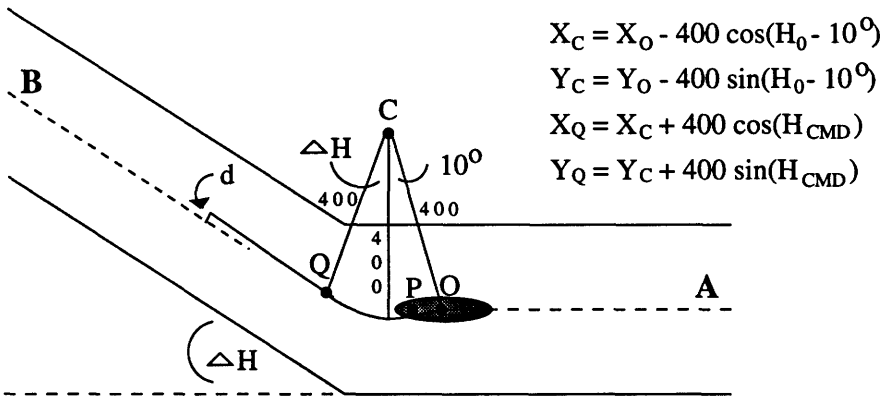


Figure 4.3 Simplified geometric modeling of a submarine turn.

Appendix H.4 gives the results of geometric model tests of the turning aids in the environmental database. Here, a hypothetical turn in a generic harbor from Segment A to Segment B is illustrated (Figure 4.3). The geometric model starts from the assumptions that the OOD is exactly on the centerline of Segment A and gives a standard rudder command at the moment the bearing of the turning aid has reached the angle read from the course card. To allow time for the OOD to finish speaking the command and the modeled delay in the automated helm it is assumed the helmsman begins his response 4 seconds later. Thus,

if the sub is traveling at 12 knots and the command is given from Point O, the OOD will be at a point P (26.6 yds. further along the segment) when the rudder begins to move. Because the conning tower is actually 25 yds. away from the modeled center of the submarine, the submarine position will have just reached Point O when the OOD is at Point P, so we will use Point O at as the start of the turn in the analysis.

Empirical tests revealed that the turn radius of the submarine at a rudder setting of 15° is approximately 400 yds.. After the rudder has turned, the $-u \cdot r$ term in the normal component of the submarine's motion (see Appendix F) causes a significant initial sideslip away from the intended direction of turn. For rudder settings of 15° this sideslip induces approximately an extra 10° of arc just to regain the original direction of travel, even though the heading of the submarine has increased continuously (Figure 4.3). Thus, for a 20° turn the sub must go through 30° of arc on a circle of radius 400 yds., starting from an angular direction 10° less than the initial heading H_0. The endpoint of this arc, Point Q in the figure, can be computed trigonometrically in two steps: first find the center of turn C from the initial angle of motion ($H_0 - 10°$) and then find Point Q from the goal angle at the commanded heading H_{cmd}. At this point on, the sub will be aligned with the heading of the centerline. The subsequent path would be straight, so we now need only compute the perpendicular distance of Point Q to the centerline extension. Anything less than 20 yds. is acceptable since such deviations could be quickly corrected using the range markers for that segment (for those segments without range markers, a smaller threshold of 10 yds. should be used). Otherwise, a new turning point O' should be computed by moving the entire curve forward along Centerline A until point Q is on the Centerline B (see Section 4.4.1). Note that the source of these deviation errors cannot be assumed to be only in the location of the turning aid, since certain simplifications in Version 1.0 in the submarine dynamics and the lack of water currents could lead to different turn paths than in real situations. In addition, some turning aids are range markers which should not be moved because they have already been verified as perfectly aligned with their associated centerline. Thus, rather than moving the turning beacon to a new location, the associated turn bearing in the virtual course card should be updated to reflect the new angle defined by the vector from O to the turning aid. As long as the system is not used for onboard rehearsals of particular harbors, this will not lead to negative training transfer since a new course card with different values would

be available for study and use in the real situation. See Appendix H.4 for the computed points O, C, and Q and the resulting deviation d for each of the seven turns in the virtual channel. Where d is above threshold, the point O' and the new suggested turn bearing is given as well.

4.1.2.5 Land Database

The elevation data used in the visual representation of the land and shoreline has been derived from a reliable outside agency, namely the United States Geological Survey (USGS). The resolution of three arc-seconds per datapoint (approx. 100 yds.) of this data is more than sufficient for accurate modeling of the shoreline. The combination of large inland areas into large single flat polygons should not affect the fidelity of the visual scene since King's Bay is almost entirely flat. This minor sacrifice in polygonal detail is more than made up for by the resulting increase in frame rate, which was seen to be on the order of 4-5 Hz. in informal tests.

4.1.2.6 Water Depths

The water depths outside the channel have been read in from the DMA charts in lat/long and converted to X-Y in yards for the depth database. The 3 arc-second resolution maintained throughout the database meets the resolution requirements from Section 2.2.6. A threshold of 28 ft. is used for detecting collision with the ocean floor, as recommended by naval officers familiar with the Los Angeles class submarine. Since the portions of the channel used in the simulation are all uniformly safe and free of dangerous sand bars and hills, modeling all points inside the channel at 45 ft. (the value noted on the DMA charts) is sufficient for preventing bottom collisions while inside the channel. The only tests necessary on the depth database are correlation of each nonzero datapoint with the land elevation database to check that such a datapoint lies in water and not on the land. All such inconsistencies should be set to zero to ensure that the submarine will always run aground and reset before reaching the shore.

4.1.3 Submarine Dynamics

Appendix F discusses the equations governing the dynamics of the submarine, which are based on a simplified model derived from [War82], leaving out equations dealing with roll or pitch. These simplifications are justified, since the OOD task occurs entirely at the surface, and at low speeds for which the roll and pitch of the real vessel are negligible. Other approximations such as the use of a turning radius depending only on rudder angle are minor and should be sufficient for the purposes of the training study, which "did not warrant the implementation of a high-order high-fidelity model" [WHLG94]. Furthermore, the naval officers with OOD task experience who evaluated the system (see Section 4.5) indicated that the submarine's responses were realistic.

4.2 Interaction: Logical Interface

The main goal here is to validate that the interactions afforded by the logical interface are appropriate to the application. This occurs along three guiding principles:

1. Support just those actions required for the task.
2. Hide the VE system from the user.
3. Evaluate inherent hardware and software tradeoffs where necessary and implement carefully to avoid negative transfer of training.

The logical interface for OOD Version 1.0 is chiefly driven by speech input, therefore this section begins with a validation of the different uses of speech recognition in the simulation. After this, each of the physical aids used in the simulation will be operationally validated, both qualitatively and using tests that verify that they convey the associated information.

4.2.1 Speech Input and Output

The use of voice recognition to allow the trainee to give verbal OOD commands to the simulation is easily validated, since this implementation exactly matches the means used on the bridge for communicating such commands to the helmsman. The use of voice input for system commands that switch views or invoke physical aids is not as easily justified, since

it differs fundamentally from the physical methods used in the real situation. However, simulating most of these physical objects and actions, such as holding a handheld card or grabbing and looking through a pair of binoculars, would be difficult with the given hardware available. Though possible to implement, extra devices like the Phantom or the dataglove would be required for haptic input and feedback, and these would add to the complexity and awkwardness of the logical interface. Therefore, using Principle 2 above, it was decided that extending the use of voice input to the invocation of physical aids and alternate views would be the best way to keep the interface from excessively intruding on the user, as long as the number of system commands remains manageable (high memory loads could adversely affect task performance). Where possible, if an alternate method of invocation is easily implemented with the devices already in place, then this is allowed in addition to a voice command.

4.2.1.1 OOD Grammar

The HARK system provides a function that takes in a phrase and returns true or false depending on whether it is a valid phrase in the given grammar. This function was used to unit test the grammar for OOD commands in Section 3.3.1 against each command in Table 2.1. All required commands successfully passed this test. After every update to the grammar, regression tests using an automated script containing the commands in Table 2.1 should be run using this HARK facility to reverify the correctness of the grammar (with updates to the script, too, as new commands are required).

4.2.1.2 Verbal and Encoded Response

Validating the input grammar is only half the job, however, since none of the actual effects of the commands have been tested. Next, the voice recognition process should be run in a mode that prints out each recognized phrase as it is received. Then, for OOD commands the program should print out the command type and value to be sent to the dynamics process over the blackboard. For system commands, the program should print out the command code being sent to the graphics process. This output should be checked, ideally by automatic test drivers, for validity with respect to a predefined list of test outputs. Throughout the tests the speech output simulating the verbal response of the helmsman should be compared

to the right column of Table 2.1 for correctness. Furthermore, to be consistent with the verbal acknowledgment of OOD commands, repetition of each system command should be checked to occur simultaneously with the invocation of the corresponding physical aid (for example, after the OOD says "Compass" the system should repeat back "Compass" while invoking the compass display).

4.2.2 Binoculars

The binoculars were implemented by simply reducing the field of view in the graphics window from 45° to 4.5°, which results in a factor of ten size magnification of objects in the display, matching exactly the magnification in the binoculars used by OOD's. Unfortunately, this also amplifies the apparent jitter in the head position data sent by the Fastrak device, but this is allowable since the level of jitter is still low enough (within 0.3 in. and 0.8° for distances less than 30 in. from the source) to allow identification of different navaids. Furthermore, real handheld binoculars are also sensitive to slight movements. The use of a voice command to invoke the binoculars is justified by Principles 2 and 3 above, in that a voice command is natural, easy to remember, and matches the logical interface for OOD commands, while the use of devices that simulate the haptic qualities of binoculars could complicate the interface to the point of interfering with the task. Moreover, the simulation is aimed more at teaching when and how to functionally apply these aids toward the main task of navigating the harbor channel, rather than teaching how to physically use the aids themselves.

4.2.3 Compass

The implementation of the compass as an overlaid numerical display at the top of the submarine view meets the requirements of Section 2.3.2.2. This display, and the use of a voice command to toggle it on and off was described by one naval officer as a "good substitute for the onboard device," since approximately the same time and effort is involved in using the onboard device. Validating the accuracy of the compass to within 1° is fulfilled by the earlier validation of the orientation accuracy of the position tracker (see Section 4.3.3), since compass headings are derived directly from the azimuth (yaw) component of the tracker data.

4.2.4 Display of Heading, Speed and Last Command

The display of heading, speed, and last command have been combined into the same fixed view mode that shows the course card. This decision is validated by Principles 1 and 2 in that it reduces the number of modes and commands the trainee will have to learn in the interface, allowing him to concentrate on the task. Furthermore, making a readable display of this data that is drawn or texture mapped onto a virtual plug-in console would be difficult due to the low HMD resolution and the jitter in the position sensor. This would constitute a noticeable intrusion of the VE system into the task. As implemented, all subjects were able to easily read all three parts of this display correctly in independent readability tests and in the pilot experiments described below in Section 4.6.

4.2.5 Course Card

The implementation of the course card leaves out two redundant columns used in the original course card, namely the column labeled "Rel Bg" which contains the turn bearing relative to the segment's heading and the "New Course" column which give the heading of the next segment. The first column is not necessary since the "Turn Bg" column gives the information more directly as it would appear in the compass. The "Next Course" column is redundant since it can also be found in the "Heading" column of the next row. Therefore, eliminating these columns does not remove any necessary information for the current subset of the task and leaves more space to display the remaining five columns and maximize their readability. Use of a real naval course card for King's Bay as the source of the data in the virtual course card validates all columns except the column containing turn bearings, since these may be affected by the simplifications employed in Version 1.0. Other sections in this chapter deal with methods of checking the correctness of these values and adjusting for errors (Section 4.1.2.4 and Section 4.4.1). The minor adjustments in the headings listed for the third and fourth segments (inbound) are justified since both segments have no annotated heading on the DMA chart, being too short to use range markers.

The use of a voice command to invoke the course card is qualitatively validated by the same reasoning used in the previous sections. In addition, an alternate method is also used in which the course card is automatically displayed when the subject looks down past a

certain pitch angle. This is justified because it is a closer substitute to the real world action than a voice command and it has a quicker response. However, the voice command is more useful in situations where the trainee does not want to have to lose sight of an object to invoke the course card.

To test the readability of the course card in the HMD, subjects were asked to read aloud all entries on the course card after invoking it with a voice command. All subjects were able to correctly read all entries, except for one subject who had set too small an interocular distance for the two LCD displays, cutting off the edges of the text. After widening this distance the subject could accurately read the entire field of text.

4.2.6 Chart View

The virtual chart discussed in Section 3.3.7 has all five of the necessary features specified in 2.3.2, including demarcation of land and water, buoys drawn as colored dots along the channel boundaries, location of turning aids denoted by their initials, and special voice commands for centering on a particular buoy or channel segment. The chart even includes some of the recommended features such as symbols denoting locations of range markers and highlighting of the harbor channel with numbers shown for each segment's heading.

Using voice commands to pan to particular target areas of interest and zoom for a more detailed inspection may not be as natural as allowing head-controlled panning and zooming relative to a fixed virtual "screen" map. But, as with the electronic suitcase display, jitter in the head tracker data would likely make the text and symbols in such a chart unreadable, and the origin of this head-position-slaved map would have to be calibrated for users of different heights. Such an implementation may be more feasible as the technology improves, however.

As an initial check on the readability of the text in this chart view (which is smaller than the text in any other display mode), a comparison with some of Huey's suggested standards for readability of printed text was performed. The two most relevant standards include minimum size of characters (at least 1.5 mm.) and minimum vertical line thickness in fonts (0.3 mm.) [Hue68]. A factor of two was applied to these thresholds since the HMD's resolution is significantly poorer than that available with printed text or digital typeface for CRTs. Upon examination of the pixel widths, heights, and vertical line thickness of fonts

in both the chart view and course card, all symbols on these textual aids were found to comfortably exceed the thresholds. For example, the horizontal and vertical dimensions of each of the two LCD displays in the VR4 are 2.64 cm. x 1.98 cm. Since the pixel chroma resolution is 247 x 230, the actual size of a pixel is 0.106 mm x 0.086 mm. But, according to Virtual Research, the lenses scale up the apparent image by approximately a factor of nine, so the apparent pixel size is 0.96 mm x 0.77 mm. The smallest alphanumeric character in the chart view is drawn at pixel dimensions of 6 x 4.5, which yields a character size of 4.32 mm. x 4.62 mm. This comfortably exceeds the chosen standard of 3.0 mm. Another standard mainly relevant to the course card was spacing between lines of text (at least 2.5 mm.). The course card display uses 4 vertical pixel-lines of whitespace between each row, for an apparent distance of 3.08 mm., which meets this standard. Further discussion of digital text legibility at various resolutions and other issues in digital typography can be found in [BD87].

Subsequently, empirical tests were performed to verify the readability of the text and symbols in the chart view. Each subject was asked to read the eight segment heading numbers, the four initials of the turning beacons, and the initials of the four range markers used as turning aids. Also, they were asked to describe locations of range markers from the "o" symbols and count the number of red and green buoys. Each subject read and counted all objects correctly, except several who occasionally misread one of the digits in the heading numbers and misread "CF" as "CP." Most of these misreadings occurred on text that partially overlapped the channel or other underlying objects. In future versions, this text should be either slightly enlarged or moved off the channel or underlying object. Afterwards, a new set of readability tests should be performed to confirm the success of the fix.

4.3 Presence: Sensors and Displays

This section validates the implementation's choice of sensors and displays, based on the inherent tradeoffs among the available options. Where possible, tests on the sensors and displays to verify the requirements in Section 2.4 are described. Some of these tests generalize well to any VE application, while others, such as the verification of the perceptual cue

tuning, use objects specific to the OOD task.

4.3.1 Graphics Renderer

The combined choice of SGI's Onyx as the hardware graphics workstation and SGI's Performer as the graphics software package meet most of the requirements of Section 2.4.1. The Onyx has a multi-channel option which can output an RGB color signal of a 646 x 486 pixel window to the HMD, meeting the color and resolution requirements. Performer supports Gouraud-shaded polygonal objects with ambient, diffuse, and specular components. Light sources may be ambient or directional and may be assigned an RGB color. The Reality-Engine 2 greatly improves the base performance of the graphics rendering while enhancing the image by anti-aliasing edges and texture-mapping color images onto polygons. Performer also has built-in capabilities for displaying performance statistics on the graphics screen; these statistics indicate that Version 1.0 runs with a frame rate of 12-15 Hz. on average, depending on the complexity of the view. The lowest frame rate seen in these on-line statistics was 10 Hz., just meeting the minimum frame rate requirement. As a second check, overall average frame rate is printed out at the end of the simulation and these varied over a series of runs from 12-15 Hz. When external load from remote users and irrelevant processes is removed, these averages reliably lie on the 15 Hz end of the range. The only aspect of scene rendering not meeting the requirements is the update rate of the submarine, which occurs at about 6 to 8 Hz., making for a noticeably choppy motion. The reason for this lies not with the limitations on the graphics renderer, but on the poor bandwidth capabilities of the blackboard process, which is losing over half of the writes of the sub position coming from the dynamics process. Suggestions for fixing this problem are given in the beginning of the next chapter.

End-to-end latency is another requirement still unmet in Version 1.0. Timed tests by James Bandy using optical circuits and operational observations of visual responses to quick movements indicate end-to-end lag of at least 110 ms. Although subjects easily notice lag of this magnitude, they have not complained that it has affected their ability to perform the task. One subject in the pilot experiments had to abort the first trial due to motion sickness but did not mention head tracking lag as the cause. Future implementations may be able to make use of predictive filtering techniques and/or pipelining of device polling to

reduce this high latency. The next chapter describes possible ways to improve the overall latency to help meet the 50 ms. requirement.

4.3.2 Head-Mounted Display

The Virtual Research VR4 HMD meets all of the specifications in Section 2.4.2 except horizontal resolution. It offers 247 x 230 full-color pixels, a 60° diagonal field of view, RGB component input, weight of well under three pounds, and a cost of $8000. Despite the loss of resolution compared to the Eyegen3, the improvement in color quality and field of view represent a better choice for the OOD task. The VR4 also offers a much wider vertical field of view and better color quality than the Kaiser VIMM HMD, and lacks the black vertical lines separating the two extra LCD displays exhibited in the VIMM. Ergonomically, the VR4 is also lighter and contains less moving parts than many other HMDs in its price range. Both interpupilary distance and eye-to-display distance are externally adjustable, while each component of color may be internally adjusted via potentiometers next to the eyetubes. This allows a near perfect matching of the colors in the HMD with those on the CRT display. The choice of the VR4 is further validated by the successful readability tests above and the tests on visual cue tuning discussed in Section 4.3.6 below.

4.3.3 Head Tracker

In general, both the Polhemus Fastrak and the Ascension Bird are appropriate head trackers for the OOD task, since they measure both position and orientation in three-space. Early tests by Reid verify the Polhemus Fastrak device as meeting most of the requirements given in Section 2.4.3. The position resolution of .03 in. and orientation resolution of 0.08° easily satisfy the respective requirements of 0.1 in. and 0.5° for the OOD task. Also, the jitter range stays comfortably within these limits as long as the source-to-sensor distance does not exceed 30 in. The exact range at which the jitter fluctuations exceed 0.1 in. or 0.5° has not been determined, but jitter becomes uncomfortably high for distances of 36 in. or more, so the effective range is at least close to the 3 ft. requirement. Reid's latency measurement of 25 ms. just meets the required maximum, while the current measurement of the maximum update rate at 40 Hz. comfortably meets the corresponding 20 Hz. requirement. The one singularity of the Fastrak's Euler angle specification of orientation does not constitute a

problem for the OOD task in which the subject never needs to look straight up or straight down. Finally, the mounted Polhemus sensor itself is so small and light that its weight is negligible toward the weight of the overall helmet.

The Bird, from Ascension Technology, is the head tracker currently used by the NAWC/TSD sponsors on their own copy of the simulator. The code within the graphics process dealing with trackers has been isolated into a submodule that can read data from either tracker depending on the value of a certain flag in the Makefile the last time the code was compiled. No formal tests have been done to measure the latency, frame rate, or accuracy of the Bird, but the manual indicates its performance in these areas is comparable or slightly better than the Fastrak. The range of the device is listed at 36 in. with a positional resolution of 0.03 in. (for source-to-sensor distances within one foot) and the average RMS positional accuracy is 0.1 in. The angular resolution and average RMS accuracy are listed as 0.1° and 0.5°, respectively. The manual lists the update rate for one sensor at 100 Hz. [Asc94]. All of these specifications meet the requirements of Section 2.4.3. However, tests similar to those of Reid and Bandy above should be performed to verify that these requirements are met in practice with the OOD application.

4.3.4 Speech Recognition and Output

The accuracy of the HARK speech recognition system was tested on several subjects of a wide variety of voice ranges. Both males and females were tested, including one who had a moderate foreign accent. Each subject used the push-to-talk button and Sennheiser microphone to give voice commands to the OOD voice recognition process. Each possible OOD command and system command (not including different heading and buoy numbers) was spoken at least twice. If a misrecognition occurred on a given command, the command was not repeated (despite the "Say again, sir" response) and the subject moved on to the next command in the list. Table 4.1 gives the overall recognition rate for each subject, and the breakup into percentage of commands not recognized and percentage of commands misinterpreted. The overall average rate was 97 % while the lowest rate was 89.8 % for one of the female subjects. The table also gives the average and maximum time between the last syllable of the command and the first syllable of the response. Overall, this response time averaged to 1.45 seconds, with a maximum of 2.41 seconds, slightly above the required time

from Section 2.4.4. The vast majority of commands however, were answered well within 2 seconds. These tests should be repeated in further prototypes as the size and complexity of the grammar grows.

Table 4.1: Results of voice recognition tests using OOD grammar.
With a few exceptions, the recorded data includes **A:** Number of failed recognitions, **B:** Misinterpreted recognitions, **C:** Total Errors, **D:** Recognition Rate (percentage), **E:** Average Response Latency (seconds), and **F:** Maximum Response Latency (seconds).

Subject	Gender	# of Cmds	A	B	C	D	E	F
1	Female	48	0	0	0	100.0	–	–
2	Female	45	0	0	0	100.0	1.49	2.39
3	Female	49	2	3	5	89.8	1.47	2.41
4	Male	101	0	3	3	97.0	–	–
5	Male	44	0	0	0	100.0	1.47	1.67
6	Male	48	3	5	8	83.33	1.36	1.81
Tot / Avg	–	335	5	11	16	95.2	1.45	2.19

4.3.5 Environmental Audio

Naval personnel with OOD experience consistently list buoy sounds as being of marginal use at best in the OOD task. These cues are relied on much more strongly by part of the navigation team below deck. Likewise, the sound of the ocean is faint and has no value as a task aid. However, subjective comments from many subjects indicate that the sounds included in Version 1.0 add considerably to the sense of presence and make the demonstration of the system quite compelling. One person even admitted to getting motion sickness in the HMD with sounds turned off after she had been used to running the system with sounds. In some cases, the directional buoy sounds could possibly alert a trainee using binoculars that a collision with the buoy is possible unless corrective action is taken (even though such collisions are not modeled in Version 1.0, subjects are discouraged from allowing them). This further validates the decision to include environmental sounds.

4.3.6 Perceptual Cue Tuning

Given the inherent limitations in resolution, color, and field of view imposed by VE displays such as HMD's, artificial tuning of cues such as size, shape, or color of objects may be

necessary to convey sensory information comparable to the real situation. Being primarily a visual task, the OOD simulation applies this technique especially toward making objects suitably visible from a distance. In particular, the three types of navaids, buoys, range markers, and turning beacons, cannot be adequately identified in the HMD from moderate to great distances without some form of size scaling. This section validates qualitatively the different types of visual cue tuning described in Section 3.4.7 and verifies quantitatively that they enable each object to meet its respective requirements described in Section 2.2.

4.3.6.1 Buoys

The scaling algorithm for buoys described in Section 3.4.7 involved a piecewise linear scale factor depending on distance from viewpoint, applied to each buoy in the scene. The scale factor was set to one until the distance exceeded $ODIST$, the maximum distance at which an unscaled buoy could be seen in the HMD. Then the scale was linearly increased with distance until the distance exceeded $ADIST$, the desired maximum distance for identifying a buoy. Any distance greater than $ADIST$ used $ADIST/ODIST$ as the scale factor. This three-region adaptive scale factor was used, rather than a uniform scale factor, so that buoys near the submarine would have realistic sizes but would still be visible from a distance.

To determine $ODIST$, the observed maximum distance for recognizing a buoy and its color in the HMD, an experiment was performed on six subjects in which the viewpoint was slowly moved away from a single unscaled buoy centered in the field of view. The time at which the subject could no longer positively distinguish the object as a buoy of that color was recorded. This was done for both red and green buoys and the minimum of the two distances was recorded as that subject's $ODIST$ value. Incidentally, the minimum distance for all subjects occurred with the green buoy, reflecting the better contrast and visibility of red objects in the sea than green objects. The average of this minimum distance over all subjects was 1256 yds.. For good measure, the algorithm was implemented with $ODIST$ set to 1000 yds., since we need to begin applying the scaling *before* the buoy becomes invisible. Thus, the maximum scale factor, $ADIST/ODIST$, was set to two and used for distances greater than 2000 yds.. With this scaling algorithm in place, a second experiment was later performed on the same subjects. This time the viewpoint slowly moved toward the buoy (as it would in the simulation), and the time at which the subject could first identify the buoy

and its color was recorded. The minimum of the recorded distances for red and green buoys was recorded as the subject's $ODIST'$ value, representing the new distance of maximum visibility in the HMD. For all subjects, $ODIST'$ was greater than $ADIST$, verifying that the algorithm meets the distance visibility specification for buoys. The average $ODIST'$ for green buoys was 3553 yds. and for red buoys, 4241 yds., averaging to 3897 yds. overall. Both values are comfortably above the goal of 2000 yds.. The likeliest explanation for the lower results for green buoys is that the green is less distinct from the light blue color of the ocean surface than the red. One reason the results were unexpectedly high above the desired threshold is liberalness in subjects' decisions about inability to see a color. Another reason may be the conservative value used for $ODIST$ in the algorithm, which was set more than 200 yds. below the measured averages to help ensure the algorithm's success. See Appendix I for a full listing of test results.

Similar experiments were done to determine the maximum distance at which a buoy number could be read using binoculars. With the field of view decreased by a factor of 10, the viewpoint was slowly moved toward an unscaled buoy centered in the view (no buoy scaling was used since the simulation does not use it in binocular mode). One of the four numbered buoy faces perpendicularly faced the user, for an angle of maximum readability. The user was told to read the number on the buoy as soon as it became clearly distinguishable. On average, for green buoys, subjects first read the number "51" correctly at a distance of 798 yds.. Red buoys were read correctly from 682 yds. on average. Both numbers easily exceed the minimum distance of 540 yds. specified in Section 2.2.2, indicating that the trainee should be able to read the nearest buoy number from all points in the channel. Again, see Appendix I for the results of each subject.

4.3.6.2 Range Markers

Early tests of the simulation revealed difficulties in using the range markers from long distances. Even with appreciable deviations from the centerline, the separation between the range markers was imperceptible because they were so far away that both towers were being displayed in the same pixels in the HMD. In the extreme cases the separation was not readily apparent even with binoculars, and without binoculars the sky dominated the range marker towers to the point of rendering them invisible. To compensate for this poor

distance resolution, a uniform scale factor of two was applied to each range marker height read originally from the DMA chart. A uniform scale was chosen rather than an adaptive regional scaling like the buoy scaling algorithm because range markers are always seen from moderate to long distances and not at close distances like the buoys. This visual cue tuning of the range markers was tested experimentally by placing subjects' viewpoints at the furthest end of each centerline from its associated range marker pair. Each such point represents the maximum distance that pair of range markers would need to be visible in the task. An intentional 20-yard perpendicular deviation from this centerline startpoint was employed, since this is the approximate threshold of acceptable deviation cited by experienced navy personnel and it is the cutoff used in one of the performance measures for the pilot experiments (see Section 4.6.1).

Each subject was asked if he or she could notice any horizontal separation between that segment's range markers and then use it to guess the direction of the deviation (left or right), first without binoculars and then with binoculars. All subjects were able to correctly guess the correct direction of offset using the binoculars. Without binoculars subjects were able to guess correctly about three of the six deviations on average. This represents a substantial improvement over the unscaled situation in which almost all pairs were too short to be useful from far ends of the segment. Furthermore, the test results verify that the perceptual cue tuning of the range markers enables them to meet the requirements of the task from all relevant locations in the channel.

A second set of tests on the range marker scaling was performed, in which subjects were placed 4000 yds. away from the closer range marker in a range marker pair, also 20 yds. to the right of the centerline. Binoculars were not allowed, thus the dayboard separation at this starting point was imperceptible, as were their orange colors. The viewpoint was continuously moved toward the range markers, maintaining the same 20 yard distance from the centerline. The subject was told to say when the separation between the markers first became noticeable in the HMD, and also when one of the orange dayboards first became apparent as well. The average distance for the first measure was 2224 yds., while the average distance for the second measure was 1838 yds.. Both are close to or above one nautical mile (2000 yds.), a distance at which OODs must often resort to using binoculars for range markers, according to experienced naval personnel.

4.3.6.3 Turning Aids

The red and white turning beacons, like the range markers, were also quite difficult to identify at normal scaling. Thus, by the same reasoning as above, a uniform scale factor of two was applied in the input file on each turning beacon. To verify that this visual cue tuning allowed users to identify the turning beacons adequately, an informal experiment was performed involving placement of the viewpoint near the middle of each segment (close to where an OOD would typically begin to look for a turning aid), telling the subject a general direction in which to look, and then asking them to find the turning beacon for that segment. Reference to the virtual chart view was allowed if requested. This was also carried out for the four segments that use range markers instead of daybeacons. In each case, the subject was able to identify the correct turning aid within ten seconds of being given a direction. Thus, all turning aids in the simulated bay have been verified as satisfying their visibility requirements.

4.4 Integration Validation and Verification

Having qualitatively validated each of the above components and verified their implementation with respect to the requirements, further work must be done to validate and verify the system as a whole. This involves a series of integrated tests, in which most or all components of the integrated simulation are run using human or automatic control (or a combination of the two).

4.4.1 Turning Aids

Earlier, in Section 4.1.2.4, a geometric verification of the turn bearings was given, using approximations from the submarine dynamics equations. This gave a rough indication of whether the the turn bearings gave accurate indications of when to turn so that the submarine would end up close to the next segment's centerline after each turn was completed. However, some of the assumptions used in this method may not be equally appropriate for all turns. For example, slight turns may require the rudder to begin centering again before it has even reached the standard angle of 15°. Thus, a more effective way to verify the turn bearings is to use a computer-controlled "autopilot" that begins on the centerline of each

segment and continually monitors the turn bearing to the segment's turning aid, issuing a steer command to the next segment's heading at the exact moment the bearing reaches the specified value given in the fifth column of the course card (Point O). After the turn is complete, this test driver can then automatically check the deviation from the centerline against a desired threshold (a minor deviation is allowable, especially for those segments with range markers to assist in a finer alignment). For any turns not within this threshold, an improved turn bearing value can be estimated offline, based on the printed deviation and a display of the sub's path overlaid on the chart view with channel and centerline. For example, a right turn with the turning aid on the right that ended up to the right of the centerline indicates that the turn was made too early, thus the new turn bearing value should be increased (Figure 4.4a). The actual amount added can be estimated in a similar manner to the mathematical validation above, by translating the entire curved path of the turn forward along the direction of the starting centerline until the end of the turn aligns with the new centerline. This distance can be computed in the same way as the geometric model approach of Section 4.1.2.4 by dividing the deviation d resulting from the autopilot's turn by $\sin \Delta H$, where $\Delta H = H_{cmd} - H_0$ is the overall heading change. This distance can be used to compute a new point of turn O' on the original centerline (Figure 4.4b).

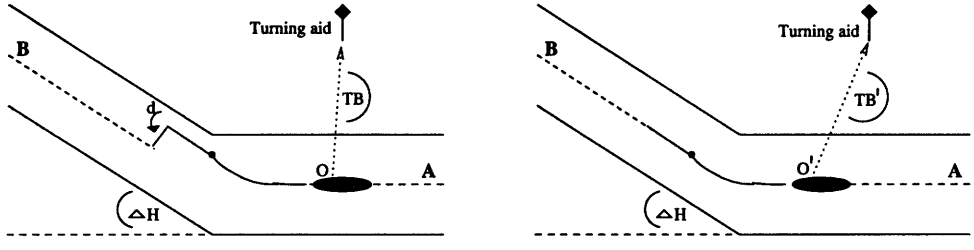


Figure 4.4
a) Turn bearing TB leads to deviation error d. **b)** New turn bearing TB' yields 0 error.

The bearing to the turning aid from this new point of turn should be used in an updated course card. The results of these "autopilot" tests for OOD Version 1.0 are given in Appendix J, which shows that six out of seven of the turns yield unacceptable centerline deviations. The appendix also summarizes the computation of updated turn bearings for these six turns.

After updated turn bearings were calculated, based on the deviations from the first set of tests, the autopilot tests were run a second time using the new turn bearing values. The new results indicate that these new bearings each place the sub very close to the next centerline, well within a 10-yard threshold, as shown in Appendix J.

Since this automated turning methodology is not a completely accurate model of the timing of human-commanded turns, a further test was performed in which an expert user (the author of the thesis) navigated the entire simulated channel using the update turn bearings. No corrections were made using range marker alignment in between turns, since one goal of the test was to see if small errors in the turns would accumulate to unacceptable levels by the end of the trial. Appendix J gives the results of this test, which also yielded very small centerline deviations. This successful verification of the new turn bearings using both computer-controlled and human-controlled turns indicates that the course card should be updated with these values as soon as possible, before the next series of pilot experiments.

4.4.2 Range Marker Alignment

Another important test is to verify operationally that the range markers serve their purpose adequately as alignment aids. Using subjects immersed in the running VE simulation, the instructor's interface should be used to place the submarine on various points on each centerline (determined from the environmental database). At each point the user should be asked whether the range markers appear perfectly aligned in the HMD (using binoculars if the dayboards are not visible without them). If not, the direction of misalignment should be noted and the data validation process above for aligning the range markers should be rechecked for each misaligned pair. If an updated range marker also serves as a turning aid in the simulation, the process above for verifying the corresponding turn bearing in the course card should be repeated, possibly arriving at a new turn bearing to reflect the changed location. After each range marker pair in the scene passes this test, the verification

of the functional requirements for range markers will be complete. Results of these tests on several subjects revealed that all range marker pairs appeared aligned for all centerline points tests (for each centerline, both endpoints and the midpoint were tested).

4.4.3 Rudder Response

Part of the purpose of the above tests is to operationally validate the relationship between an object's visual representation and its environmental database description (ie, validate the consistency of the overall computational model). Similar tests need to be done to correlate visual representations of dynamic objects such as the submarine with the underlying computational processes governing their motion. For example, while immersed in the VE simulation, a subject should give all seven possible rudder commands and look back at the rudder to verify that it is moved to the correct angular extent with the correct slew rate of 4° per second. If the rudder cannot be seen because it is submerged or because the periscope casing is always obscuring it, the visual representations should be altered to alleviate the problem by raising the height of the sub (or the rudder) in the water or reducing the thickness of the periscope casing. If any angular extent seen does not correspond to the commanded value than the code in the graphics process for articulating the rudder should be examined. If the problem is not found here, the underlying dynamics code should be reexamined, starting with the interface to the graphics (point of communication of rudder commands) working back to the interface to the voice recognition process (point of receipt of the rudder command). Several subjects were tested here, and all were able to easily look aftward and see the rudder move correctly to all seven positions.

4.4.4 Testing for Special Cases

Finally, all special case situations included in the requirements phase must be tested. For the OOD task, the most notable of these is the event of grounding the submarine. First, the response of the logical interface to this event must be verified to meet the specifications given in Section 2.2.6. While running the integrated simulator, the user should intentionally steer the submarine out of the channel to an area of shallow water (which can be determined beforehand from the DMA chart). The visual response (shaking of the viewpoint) should be qualitatively verified as dramatic enough to be interpreted as a forceful collision, and

the feedback from the verbal message played after the collision should be verified as well. In particular, the point chosen by the dynamics process to instantly "tow to the edge of the channel" should be verified as near the point the submarine originally left the channel. Since this can only be done to a rough approximation from the submarine view (the user can only estimate from the buoys where the channel boundary lies), a special chart view with the submarine's current location and the path taken so far should be examined. After verifying the correct response, the depth database should be verified operationally by steering the sub off key points of the channel and comparing the eventual point of grounding to what would be expected from the depths given on the DMA chart, using 28 ft. as the threshold. If possible, a screen dump of the virtual chart with paths of the repeated groundings should be overlaid on a copy of the DMA chart that has been reduced/enlarged to the same size. Any instances in which the simulation grounded too early or too late (past some acceptable threshold) or failed to ground on land indicate the need to modify the depth database in that vicinity. When the test runs with no failed grounding points, the correlation of the depth database, land elevation database, and visual representations of the land will be fully operationally validated.

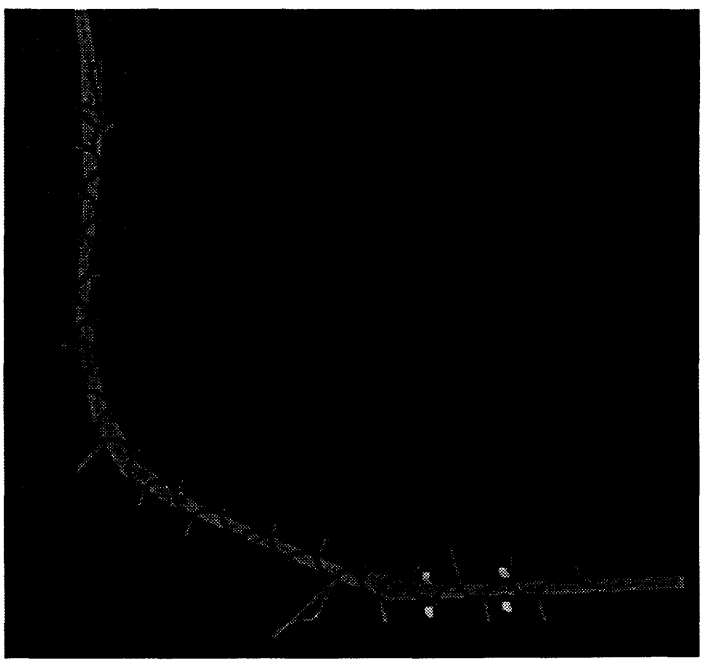


Figure 4.5 Paths showing grounding points in VE.

Figure 4.5 shows a comprehensive set of tested grounding points covering most of the virtual channel. These grounding locations were looked up on the DMA chart showing nearby water depth readings (see Figure 2.8 for a portion of this chart covering most of the tested areas). Each grounding point was either on or slightly further out from the 28-foot "contour" of the DMA chart, which validates the accuracy of the depth database. The last two segments were not tested for grounding, but they immediately fall off to unsafe depths outside their boundaries, like the previous segment, so their accuracy in the depth database can be inferred from the successful tests on the preceding segment.

4.5 Domain Expert Evaluation

One of the most important elements of a validation of any simulation system, or any large engineering effort for that matter, is evaluation by domain experts. No matter how accurately one builds the objects in the VE, the dynamics of the simulated vehicle, or the multimodal interface to the VE, a valid appraisal of how closely the simulation approximates the real task can only come from one who has truly performed it many times themselves. Unless a prototype is fortunate enough to achieve glowing reviews on the first try, these evaluations should be iteratively repeated after each series of updates to the implementation are made, until the domain experts' ratings are satisfactory.

For this purpose, several junior officers at the Naval Officers' School in Groton, CT were contacted in November, 1994, to visit the testbed and try out the OOD simulator. Each had previously navigated several harbors as an OOD, and one had actually performed the task in King's Bay. Their visit helped identify strong and weak points of the simulation, and gave valuable additional insight into how OOD's are currently trained. All four of the officers were able to perform nominally well in the simulator, although one officer still relatively new to the task did run aground on one occasion. The main criticism of the simulator was in the following areas [PL94b]:

1. Buoys could not be seen at the proper range, requiring atypical use of the binoculars. (Normally, they are used only once or twice per trip).
2. The chart view should show true bearings and other typical markings.
3. The chart view should not show the position of the boat.

4. The view from the bridge needs adjustment.

Despite these problems, all four of the officers expressed enthusiasm about the simulator's potential for training the OOD task. They even suggested that it might also be useful for onboard mission rehearsals, to help familiarize an OOD with the idiosyncrasies of a new channel before entrance into the real harbor. The above criticisms were accounted for in an updated set of requirements for Version 1.0 (see Chapter 2), and led to the inclusion of the buoy scaling algorithm (Section 3.4.7) and additional symbols and numerical headings for the chart view (Section 3.3.8).

One drawback of the first officers' visit was the lack of an organized record of objective ratings on different aspects of the simulator. Such a record would be helpful in identifying specific problem areas and suggestions for how to solve them, as well as for setting objective criteria for final acceptance of Version 1.0 based on the overall approval ratings. This idea led to the compilation of a six-page written survey to be administered in future OOD visits. The last such visit, in February, 1995, was by one of the four officers, Lt. Allen Andrew, when he was called back to reevaluate the system. After he ran through the entire channel in the simulation, he filled out the written survey which asked for numerical ratings on a wide range of aspects of the simulator and asked open-ended questions about the trial and about task-related quantities useful for future development (see Appendix K for a complete reproduction of the written survey and Lt. Andrew's answers). Most of the numerical ratings were either 5 (Extremely Satisfactory), or 4 (Moderately Satisfactory). The only marks of 2 (Moderately Unsatisfactory) were on the size of the range markers and visibility of distant range markers, which led to the decision to double their heights in the simulation. The only criticisms in the open-ended section were that the task could not be performed as well as in the real world because of lack of advice from the chief navigator (which is deferred till later versions) and a rare problem with the speech recognition system repeatedly misinterpreting the word "two" as "seven" (all other commands worked fine). After he completed the survey, Lt. Andrew expressed praise for the overall improvement in the visuals since November.

While thorough written feedback on the simulation is important, so is the careful planning of the expert's time spent using the simulator itself. Exposure to each feature that

might benefit from expert advice must be included in one or more trials of the simulation. Any missed detail could lead to negative consequences during later experiments. For example, the MIT evaluation sessions with OODs did not include specific checks on the turn bearings of the course card; the error in some of these values was caught later by naval officers evaluating the system at NAWC/TSD in Orlando. This led to the above application of off-line geometric verification and integrated autopilot testing of the turns, but the corrections that were derived were not in time for the first series of pilot experiments. Thus, subjects also needed to "learn" their own corrections for some of the turn bearings in the course card.

4.6 Pilot Experiments

In any long-term simulation project using evolutionary prototypes, each prototype or part-task trainer must not only be tested for meeting the requirements, but also should be evaluated through experiments involving users in its target training class (in this case novice OOD's). Such experiments are valuable in determining whether subjects are able to learn to effectively perform the simulated task, and if so, what the "time constant" is for the average learning curve and at what level the curve tops out. If the subjects are not able to show improved performance after several trials, these experiments serve as a sanity check on the design and implementation on the current prototype, and may help identify those areas in most need of improvement, based on experimenter's observations and subject's comments.

4.6.1 Experimental Procedures

The first set of pilot experiments on OOD Version 1.0 were scripted by the BBN training team. The experiment consisted of five 2-hour sessions. Each session began and ended with a survey to determine if the subjects were undergoing any symptoms of motion sickness. After each trial a survey about the subjective strategy and difficulty of the navigation was also administered, with room for written comments at the end.

The first session was for pre-trial training only. In this session, the subject filled out a set of general surveys on previous boating experience, past VR game or application usage, and current health. Then the subject was shown a 20-minute videotape illustrating the

navigation of two segments of a fabricated harbor channel. This video also explained the VE equipment used in the simulation as well as the major voice commands (both OOD and system commands). Then the subject was allowed time to read a training manual containing pictures and explanations of different situations from the fabricated channel. Finally, the subject donned a headset microphone to practice giving voice commands (only the automatic verbal response was given; otherwise, the simulation was still "turned off").

The second session began with a brief review of the voice commands, leading into a short simulation using the fabricated channel to accustom the subject with a generic ocean VE and the responses of the submarine to OOD commands. Afterwards, the subject was given extra instruction by the experimenter about the King's Bay channel and the configuration of its segments and navaids. The subject was then allowed to review the course card and nautical chart for King's Bay and ask questions about the task. Then the first full simulation trial was administered, using just segments 2 through 6 of the inbound course to reduce the time spent in one session in the HMD.

The three remaining sessions each included three full trials. Each trial began at one of four startpoints some distance to the left or right of the center of the start of Segment 2 (either 35 or 40 yds. off center). The startpoint was specified along with subject and experimenter name and trial number through the experimenter's interface (EI). The EI also communicated with the dynamics process to record the submarine's location every second of the simulation. This data was written into a periodic datafile containing the subject's name and run number in the filename, with extension .per. The EI also recorded aperiodic data from the dynamics program, including rudder, heading, and speed commands with their associated simulation times, as well as special case events such as running aground. This data was written to an aperiodic datafile of the same name with extension .apr. Appended to the end of this file was a tabular summary of performance measures both for the overall channel and for each individual segment. The main performance measures were percentage of time spent out of the channel and percentage of time greater than 20 yds. away from the centerline. Other related measures included mean, standard deviation, and RMS deviation from the centerline (not including turn segments or the 200 yard buffers at the ends of each segment) as well as number of times the sub ran aground. These performance measures are based somewhat on suggestions by the naval officers who evaluated the system, but are not

officially endorsed by the navy since there are no standard evaluation criteria and actual shipboard ratings of OOD performance vary greatly depending on the style of the captain.

After each trial the subject was shown the entire path of the submarine overlaid on the chart view and was told the percentages of time spent out of the channel and outside the 20-yard deviation threshold for each segment and for the overall channel. After filling out the post-trial questionnaire, they were given the opportunity to review the paper course card and chart. However, they were not allowed to ask questions about course strategy or receive any verbal feedback on their performance.

4.6.2 Results

The pilot experiments included seven different subjects, five male and two female, all undergraduates at MIT who had never performed the OOD task before and had only moderate sailing experience at best. Appendix L lists the scores of each subject for the entire channel over all ten trials. Each subject exhibited dramatic learning curves in almost all performance categories, especially in category A (percentage of time spent $>$ 20 yds. off center). The end of Appendix L shows graphs of the scores in category A for the six subjects. Subject 4 was able to achieve perfect scores of 0% by the second trial. Most others bottomed out close to 0% by the third or fourth trial.

Throughout the trials, statistics on voice recognition rates were also kept. Table 4.2 gives the average recognition results over each session (except for the initial classroom training session) for each subject, as well as the overall averages by subject. The results may be a bit higher than the true recognition rate achieved because commands that were recognized but misinterpreted were not counted as mistakes. All subjects' overall rates were comfortably above 90 % except for Subject 1, a female whose rate was about 76 %. The system in particular had trouble recognizing the commands "All ahead two-thirds" and "Compass" for this subject, also occasionally misrecognizing digits in numerical commands. The experimenter often had to enter in commands through the instructor's interface if these commands were missed a second time in a row. Overall, however, the voice recognition system worked well for the task.

Readability of the course card was also confirmed for each subject at the beginning of each daily session by briefly having them look down and verify with a "yes" or "no" whether

Table 4.2: Recognition rates (percentages) by subject and session for the pilot experiments.

Subject	Gender	Session 2	Session 3	Session 4	Session 5	Average
1	Female	58.6	76.8	76.8	79.7	75.7
2	Male	99.1	97.1	100.0	100.0	98.8
3	Male	95.3	98.4	95.4	98.6	96.9
4	Male	95.7	99.0	95.1	89.1	94.7
5	Male	93.9	96.8	100.0	96.8	96.8
6	Female	88.8	97.3	93.3	87.3	92.3
Avg		91.3	93.4	91.9	90.9	92.0

they could read the text. Not once did a subject answer "no" for this test. In addition, they were asked before the start of each daily session how many green and red buoys they could see to make sure the display was working adequately. All subjects were able to see the correct number of buoys on both sides of the segment ahead of them.

4.6.3 Interpretation of Results

The fact that all subjects showed significant improvements in performance indicates that the simulator adequately provides the required information to learn the simplified OOD task with no currents or intervention of external traffic or interaction with the navigator. In fact, the short learning curves seem to indicate that these unmodeled task elements may constitute a majority of the workload in the task. The next logical direction would be to specify a new set of requirements for the next prototype to include models of water currents and their associated visual and internal dynamics representations. Interaction with other water traffic may be reasonable for Version 2.0 also, although the issues of how to design relevant performance measures or uniform experimental conditions are more complex here. Due to the short learning curves, adding artificial cues to the experiment would not be advisable at this point since subjects seem to do fine without extra cues (unless, of course, the other added task elements can be shown to add sufficiently to the difficulty of the task).

Another possible explanation for the rapid mastering of Version 1.0 is that the simulation gave too *much* information to the subjects, allowing them to perform better than they otherwise would in a real situation without currents, water traffic, or interaction with the navigator. However, even though some of the methods through which the simulation conveys

information necessarily differ from that of the real world, the validation discussions of earlier sections indicate that the information itself is the same as that given in the real task and the time and effort involved in obtaining it is close if not identical to the real situation. Testimony from the domain experts in the last evaluation session also tend to defeat the possibility of over-cuing. Therefore, the pilot experiments, in exhibiting respectable learning curves, favorable recognition rates, navaid visibility, and text readability, represent further validation of the prototype's usability and potential training effectiveness.

Chapter 5

Future Work

This chapter first discusses some of the remaining problems with the current OOD implementation, and possible methods for improving them in future versions. The remainder of the chapter discusses VE validation and verification for more general VE training issues not previously discussed, such as the use of instructional features and artificial cues not available in the real situation, the use of intelligent tutoring agents, and more complex VE's with large numbers of human or autonomous participants.

5.1 Problems

The two most prominent areas in need of improvement in the OOD Version 1.0 implementation are the high end-to-end latency, and the low communication bandwidth limits available from the blackboard process causing choppiness in updates of the submarine position in the graphics scene. Possible fixes or alternative solutions to these problems are suggested below.

5.1.1 Latency

The most outstanding requirement from Chapter 2 not met in Version 1.0 is the overall latency, as measured between time of a head movement and the completion of the corresponding graphics frame drawn in the HMD. Part of the source of this is from the signal latency of the Fastrak device in transmitting the six floating point values for position and orientation to the Onyx. This contributes a substantial part of the latency–about 43 ms. according to performance tests run on an empty graphics scene. The additional latency due

to the time to render the frame buffer on the Onyx is 67 ms. on average (since the average frame rate is 15 Hz.), for a total of 110 ms., approximately one-ninth of a second.

The latency due to the Fastrak device is only about 25 ms. (which is still much higher than that specified by the manual, which cites an 8 ms. latency) for each Polhemus sensor. Future versions should attempt to cut down on this part of the lag by exploring other modes of data transmission. For example, the Fastrak can run in continuous mode rather than polling mode. In continuous mode, data values are read at the maximum possible and at the time of each request from the graphics software the more recent data record is returned, while in polling mode the device waits idly until a request comes in so the graphics process must wait for a full Fastrak cycle before going on. Additional speed may also be gained by running the device in binary mode, which is a more compact data format than the ASCII mode currently being used in the simulation.

The additional latency due to the graphics update could possibly be further reduced by simplifying further some of the polygonal models of objects such as the static watercraft, or the land object. Another alternative is to use the level-of-detail capabilities of Performer to render distant objects using a simplified 2D representation, further speeding the rendering time. If the overall frame rate is still over 50 ms., then predictive filtering techniques based on first and second derivatives of the most recent Fastrak records should be explored [She92].

5.1.2 Inter-process Communication

As mentioned in Chapter 3, the blackboard process for communication between the different modules in the simulation does not allow enough communication bandwidth to meet all the performance requirements specified in Section 2.4.1. In particular, the effective frame rate of the graphics update of the submarine position suffers most severely, reaching values as low as 5 Hz. Part of this is due to the fact that the dynamics process sends updates of the submarine's X-Y position over the blackboard at a rate of 20 Hz., which is a greater rate than the underlying IRIX operating system's implementation of Unix socket writes can accommodate. As a result, a large percentage of these writes are lost and the previous writes are seen two, sometimes three, times in a row by the reading process. This problem is peculiar to Unix platforms, possibly due to an automatic buffering that occurs on incoming socket data when the context switches to another process. The implementation of the

blackboard on DOS does not exhibit this doubling up of data records and supports greater throughput.

One major drawback of using the blackboard is that its performance degrades in proportion to the number of data objects posted, as well as the length (number of fields) of the objects. In the OOD simulation, there are approximately five of these objects, with field lengths ranging from one to ten, since the simulation uses several independent processes on several different machines. However, the two largest software components, the graphics process and the dynamics process normally run on the same machine, the Onyx. Thus, one alternative that would at least reduce the overall load on the blackboard is to use shared memory blocks for data passed between these two processes. In particular, the latest data record with the submarine position and heading would be immediately available for the graphics. Since the dynamics process frame rate of 20 Hz. exceeds that of the graphics process, 15 Hz., the graphics would be virtually guaranteed to render a new submarine position each frame, resolving the most outstanding communication problem. Other data exchange, such as the transmission of a voice command, or the recording of data by the experimenter's interface (which typically occurs at 1 Hz.), or the sending of buoy locations and head orientation for the sound server, is either aperiodic or very low frequency and therefore can be adequately supported by the blackboard.

The ideal solution to the communication problem would be something that offers the speed and ease of access of shared memory, but also allows other processes on other machines to read the shared data. One such technology called *Scramnet*, by Systran Corp., has been developed using special memory extension cards with ultra-thin fiber-optic connecting cables between different machines. The end result is a global shared memory area that is accessed on each host just as if it were regular physical memory. Write and read times using this cables are on the order of a millisecond or less, regardless of the machines (as long as they are connected via the Scramnet cables), much faster than with the blackboard.

5.2 Artificial Instructional Cues

All of the simulation components discussed so far in this thesis for the OOD task VE have been carefully modeled and perceptually tuned to match as closely as possible the

corresponding interactions or objects of the real situation. However, one of the most unique characteristics of VE over conventional simulators is the capability to present artificial cues and sensory input not available in the real world. These could be used as instructional features in the OOD task, for example, by highlighting the harbor channel and centerlines or displaying the predicted path of the submarine (based on the heading, speed, and rudder angle) as a curve overlaid on the surface of the water (Figure 5.1). In training situations, the use of artificial cues introduces possible risk of negative transfer from the simulated task to the real task. This could occur because a student may have come to rely on artificial cues so heavily in the simulation that his or her performance in the real situation without the cues is severely hampered. To prevent these extra cues from becoming a "crutch" their use must be carefully managed according to a student's level of mastery, gradually removing each cue as the student becomes proficient at the task element for which that cue was designed.

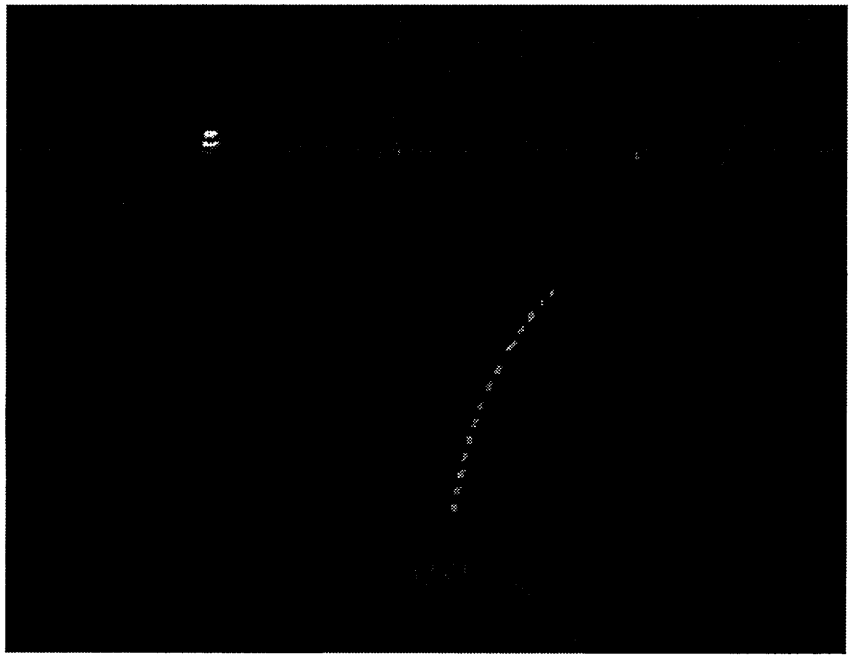


Figure 5.1 Display of submarine's predicted path.

The notion of operationally validating an artificial cue seems contradictory, since there is no corresponding real world object that this cue is trying to represent. However, every cue in the VE used for instructional purposes is likely to be tied in some way to an abstract idea

or invisible entity that is used by domain experts to help them perform the task. During an evaluation session by these experts, if any cue sufficiently deviates from the experts' "mental model" of this task-aiding idea entity, then the model of the cue must be changed accordingly. Other artificial cues will often have a mathematical description based on data already on hand describing real objects used in the task. Thus, before the above expert evaluation is carried out, data validation should be performed on these cues, checking for each instantiation that the correct corresponding data and derivation from baseline objects was used. Of course, the validity of any new sources used in the acquisition or derivation of data for the artificial cues must also be checked.

Another aspect of operational validation of instructional cues is checking that they do not unduly distract a subject from the baseline cues modeled upon real objects, which could occur through inappropriate use of color, contrast, or shape. Worse yet, the implementation of a cue might actually obscure an important baseline cue, or reduce the effectiveness of the perceptual cue tuning that had been verified before these cues were added. The following informal guidelines can help detect these problems:

1. Compare results of presence questionnaires with and without each cue to help identify possible distractions.
2. Check if cues obscure baseline perspective depth cues such as repeated texture maps.
3. Perform regression tests on the perceptual cue tuning to check if a cue is adversely affecting visibility.
4. For textual displays with added cues, perform regression tests on readability.

After any deficiencies revealed by the above tests have been fixed, the cues themselves should again be tested for readability and/or visibility from required distances and angles.

5.3 Intelligent Tutoring

Another feature not yet discussed that could be useful in VE training systems are intelligent tutoring components. Conventional Intelligent Tutoring Systems (ITS) compare an online model of the student's performance with an expert model of encoded domain knowledge and administer effective coaching techniques via an instructional planning component.

Most existing ITS offer only conventional 2D textual or mouse-based HCI, although recent research efforts have begun to explore effective use of multi-media as well [GKL94]. VE has the potential to offer more effective coaching through simulated human agents that interact with the student as a human teacher would, except that, instead of blackboard and chalk, these agents have the ability to render 3D illustrations in real-time with auditory and haptic feedback included. Decisions to include ITS in a simulation, especially one using VE technology, should be weighed carefully with the time and manpower constraints of the project. The ITS's relationship with the VE and its interactions with the student must be operationally validated, perhaps through comparison studies of real task training versus VE task training, closely involving domain experts.

5.4 Complex Virtual Worlds

The OOD task represented a single-user virtual environment in which the scenery and interactions were all relatively straightforward and deterministic. However, new capabilities in network technology and distributed simulation have made possible the construction of VEs with large numbers of human and computer-simulated participants using a wide range of sensors and displays. In these VEs each step of the simulation can exponentially increase the number of possible states since each user can perform any of a number of actions. Reasoning about the effects of these actions on the current and future states of other participants may seem almost intractable. For this reason, separate validation and verification of each participating agent's allowed interactions and VE-presented scene may not be sufficient to validate the VE system as a whole. In some cases it may be necessary in the requirements to enforce standardization of components, as with DIS, which requires each participant to use the exact same terrain database and each vehicle to communicate using pre-specified identification codes listed in the current IEEE standard [LS+93]. These environments, however, are more like a set of loosely-coupled standalone simulators. Other VEs for training team cooperation may involve direct multi-modal contact and/or communication between participants. To encourage creative team solutions to tasks, there are likely to be fewer constraints on an individual's capabilities and roles within the scenario. Examples of such VEs include using 3D graphics for group visualization of multi-dimensional scien-

tific data, or a virtual "easel" on which multiple artists simultaneously draw in 3D, or a multi-user dungeon (MUD) loosely simulating a real world group problem or task. One possible approach to experimentally validate such systems with respect to a desired or required performance goal is a "hill-climbing" technique. First, establish a "control" version of the VE by performing applicable unit validation and testing for each VE component, using the autonomy/interaction/presence taxonomy. Then, record performance data for several trials of the simulation with a full team of users. Next, make some small change in the autonomy, interaction, or presence component of the VE, based on the data and written comments of participants. Record data for a new set of trials using the changed parameter. If the data does indeed show improved team performance toward the desired goal, this indicates a positive directional derivative in the "AIP cube," so the next trial should continue changing the component in this direction to maximize performance. If a decrease in performance was seen, then a change in the opposite direction should be used for the next trial. This technique has two major caveats, in that finding an optimal configuration is a painstakingly slow process, and the performance levels of this configuration may only be a local maximum in the AIP cube.

Chapter 6

Conclusions

Even though many of the tests for verification of components of the OOD simulation were highly specific to that particular task, the validation and verification methodology used throughout the thesis should generalize well to other VE simulations for training. The outline below summarizes this methodology, emphasizing the Autonomy, Interaction, and Presence taxonomy within each phase.

6.1 VE Validation Outline

I. Task analysis

A. Decide on type (flowchart, questionnaire, outline, or other method).

B. Isolate separate subtasks (horizontal modularity)

C. Include hierarchical level of detail (vertical modularity).

II. Requirements

A. Decide from task analysis which subtasks to model first.

B. Autonomy

1. What computational models and processes are needed?
2. What are the behaviors of objects in the VE?

C. Interaction

1. What is the means through which the user must be able to interact with the VE?
2. Give mapping of task-level user actions to VE responses.

D. Presence

1. What visual/auditory/haptic channels must be used and to what level of fidelity/resolution?
2. May specify a certain device type (eg. HMD vs screen projection) but not a certain product.

III. Implementation description

A. Autonomy

1. Give visual/auditory/haptic description of computational models.
2. Specify sensory representations and behavioral models.

B. Interaction

1. What are the primary and secondary input and output modes?
2. How are these modes similar/different from those of the real task?
3. List each task-level command and its VE response(s).
4. Again say how they are similar/different from real task.

C. Presence

1. Give relevant parameters of sensors and displays.
2. Specify perceptual cue tuning methods and give reasons.

IV. Validation and Verification

A. Autonomy

1. Validate the level of detail of visual representations.
2. Validate data for behavioral models.
3. Verify through unit tests that required behavior occurs.

B. Interaction

1. Validate the choice of input and output modes.
2. Verify that the task-level commands elicit the required VE response(s).
3. Validate differences between VE actions and real task actions using these principles:
   i. Support only those actions needed for the task.
   ii. Hide the VE system from the user.
   iii. Evaluate inherent HW/SW tradeoffs and implement carefully to avoid negative transfer of training.

C. Presence

1. Verify that sensor and display parameters meet the requirements.
2. In cases where they don't, due to inherent HW tradeoffs, validate that the implementation tradeoff is optimal for the task, given the alternatives.

D. Integration tests

1. Check all channels and types of data communication between software modules.
2. Use a computerized operator to exercise the simulation and determine if the information provided to a human operator supports successful completion of the task.
3. Run single-trial tests for usability.
4. Test for robustness, ie that all the documented special cases are handled correctly, and those that aren't documented don't crash the system.

E. Evaluation by domain expert

1. Schedule periodic evaluations by domain experts.
2. Carefully plan evaluation sessions to ensure domain expert encounters all those aspects of the simulation for which expert evaluation is sought.
3. Use a written questionnaire to rate satisfaction with the objects in the scene and the means of interaction.
4. Leave room for comments on improvements for any unsatisfactory elements.
5. Include a section asking for task information relevant to next prototype.
6. Ask about effectiveness of any artificial cues used.

F. Pilot experiments (for each prototype)

1. Run multiple trials on novices.
2. Initial session for classroom familiarization with task.
3. Second session for familiarization with simulation.
4. Validate performance criteria as meaningful to task.
5. Compare performance with vs. without artificial cues.

6.2 Summary

The OOD simulation has been the largest full-scale VE project so far in the VETT program. Choosing a naval application area in need of a formal training program has helped us gain new insights into the design, implementation, and validation of VEs for training, despite the OOD task's lack of haptic interaction and sensorimotor skill involvement. Initial work on task analysis and requirements by the BBN training team, based on research in the field and correspondence with domain experts, helped ease the implementation phase tremendously. Subsequently, the implementation description for each VE component in the three-dimensional taxonomy enabled a well-defined validation and verification testing methodology for each component, speeding the process of making code fixes. Finally, integration validation of OOD Version 1.0, using integrated tests, domain expert evaluation, and pilot experiments on novice subjects, has indicated our readiness to begin the next iteration of the simulator, while identifying remaining areas for improvement.

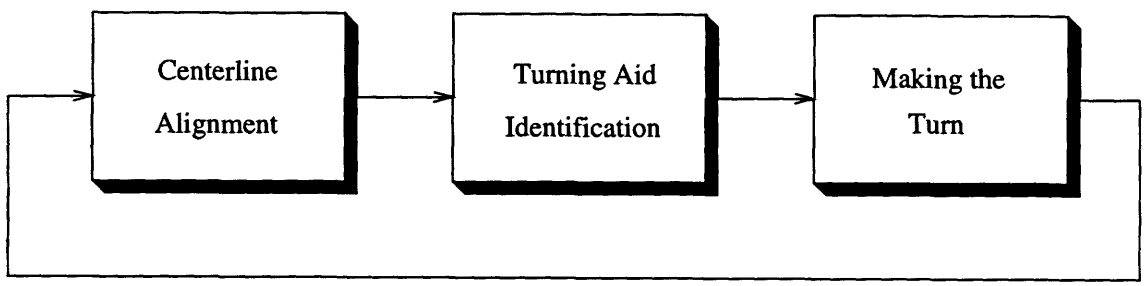
Although research on the OOD simulation and on other VE's for training is still in its early stages, the enthusiasm of the naval personnel and other visiting scientists for our training systems reflects the need for more interactive, immersive simulations. For many tasks, VE has the capability to offer more natural, task-level interfaces than conventional simulators, matching more closely the interactions involved in the real task. VE technology presents more than just a baseline simulation – it can empower enhanced simulations with perceptually tuned objects for optimal "task presence," adaptive artificial sensory cues, intelligent tutoring using multi-modal teaching agents, and complex scenarios combining multiple human and computer-simulated participants. As we explore these immersive systems for training, validation and verification of the VE will be an integral part of the research. Incorporating these techniques throughout the development cycle of the VE simulation is necessary to maximize positive transfer of training. The results in the long run will be saved money, vehicles, equipment, and most importantly, saved lives.

Appendix A

Task Analysis Flowcharts

OOD TASK

Sequential Task Elements



Parallel Task Elements

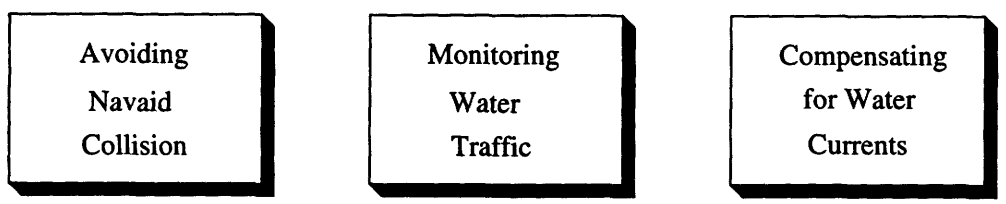


Figure A.1 Sequential and parallel task elements for the OOD.

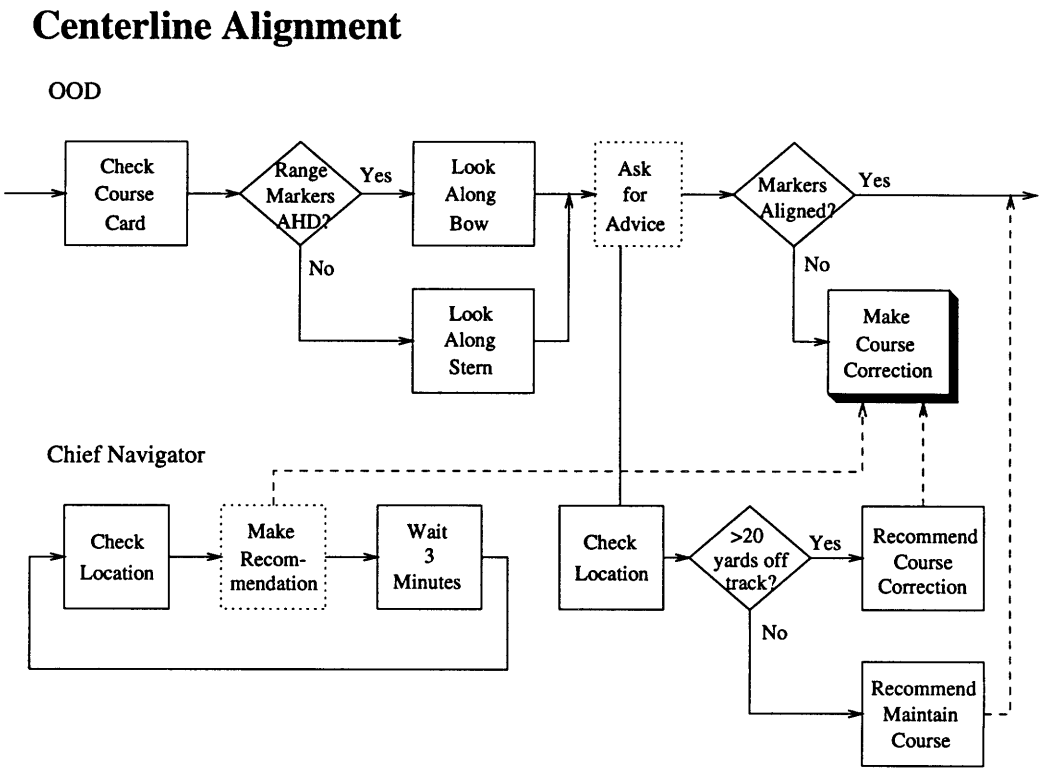


Figure A.2 Centerline alignment subtask using range markers.

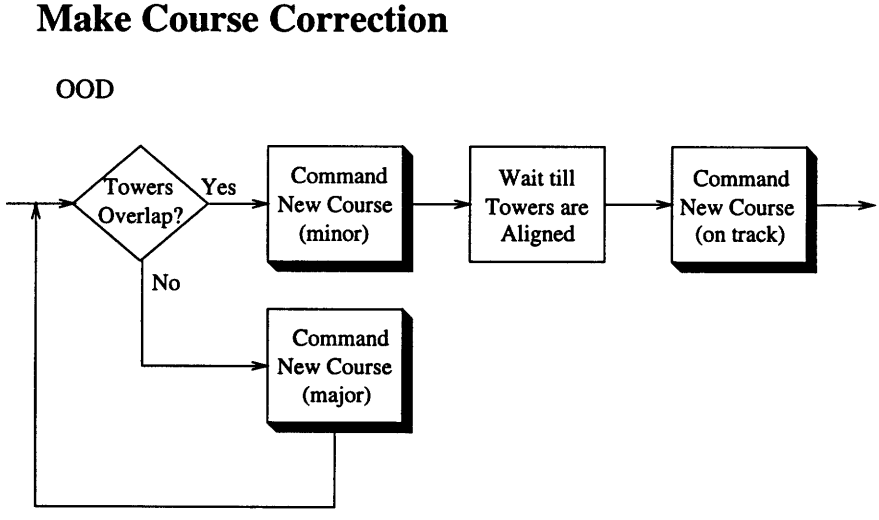


Figure A.3 The OOD finds the turning aid for the current segment after consulting the course card.

Command New Course

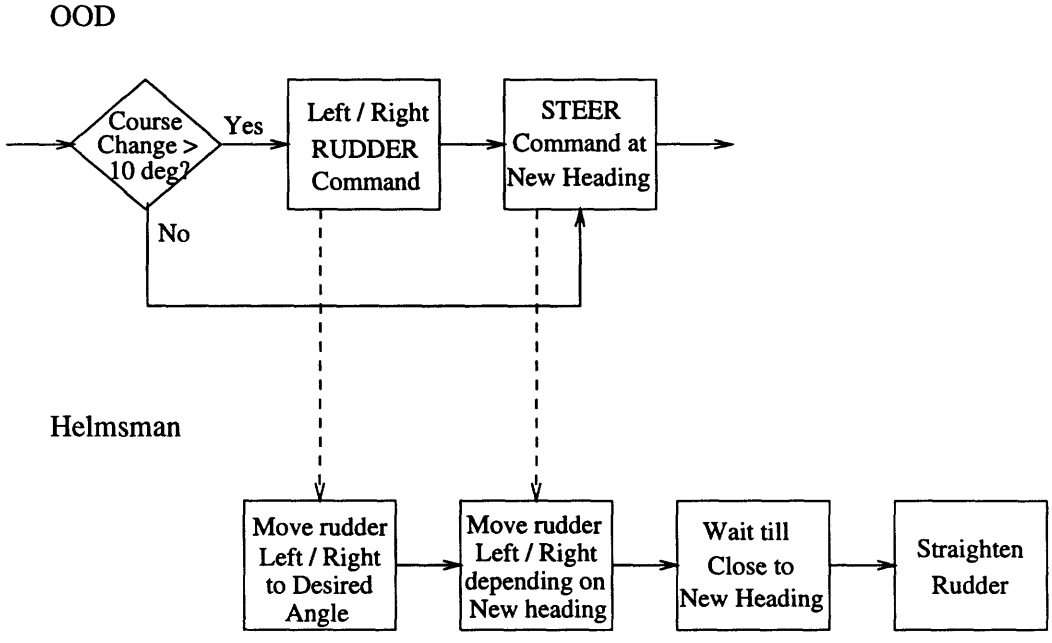


Figure A.4 The OOD can command a new course by directly issuing a STEER command or issuing a RUDDER command followed by a STEER command.

Turning Aid Identification

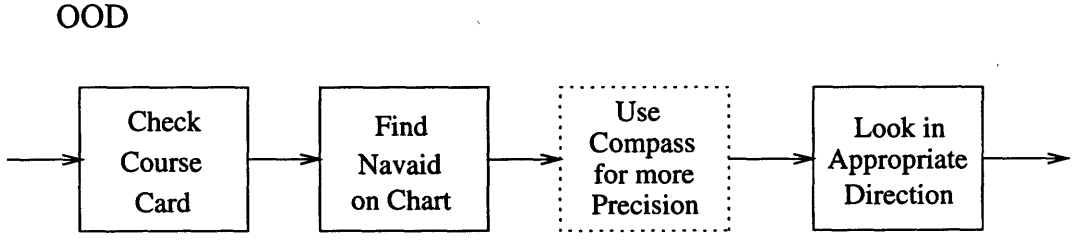


Figure A.5 The OOD finds the turning aid for the current segment after consulting the course card.

Making the Turn

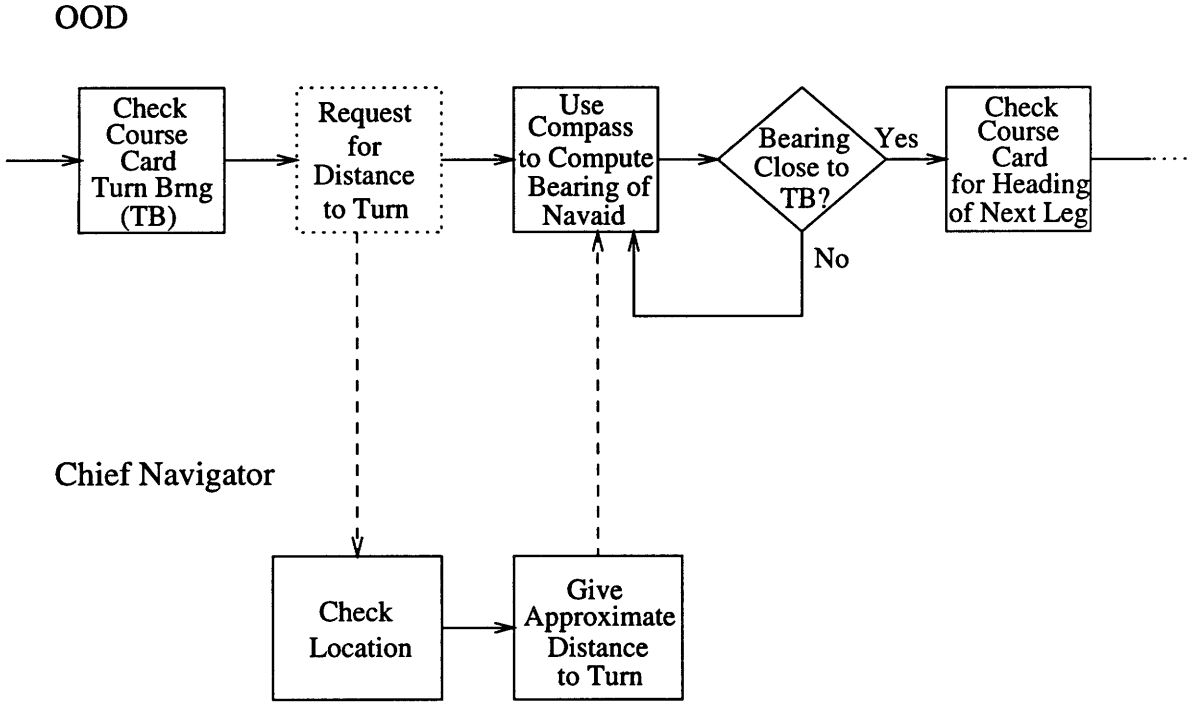
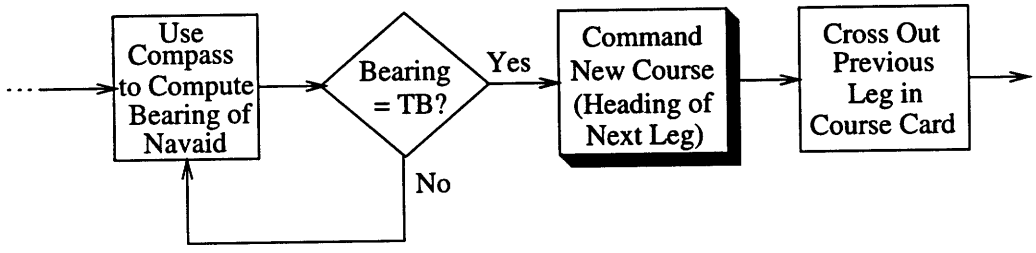


Figure A.6 The OOD commands a turn to the next segment after the turning aid's bearing reaches the value given by the course card. This figure is continued on the next page.


Use Compass to Compute Bearing of Navaid
Bearing = TB?
Yes
No
Command New Course (Heading of Next Leg)
Cross Out Previous Leg in Course Card

Avoiding Navaid Collision

OOD

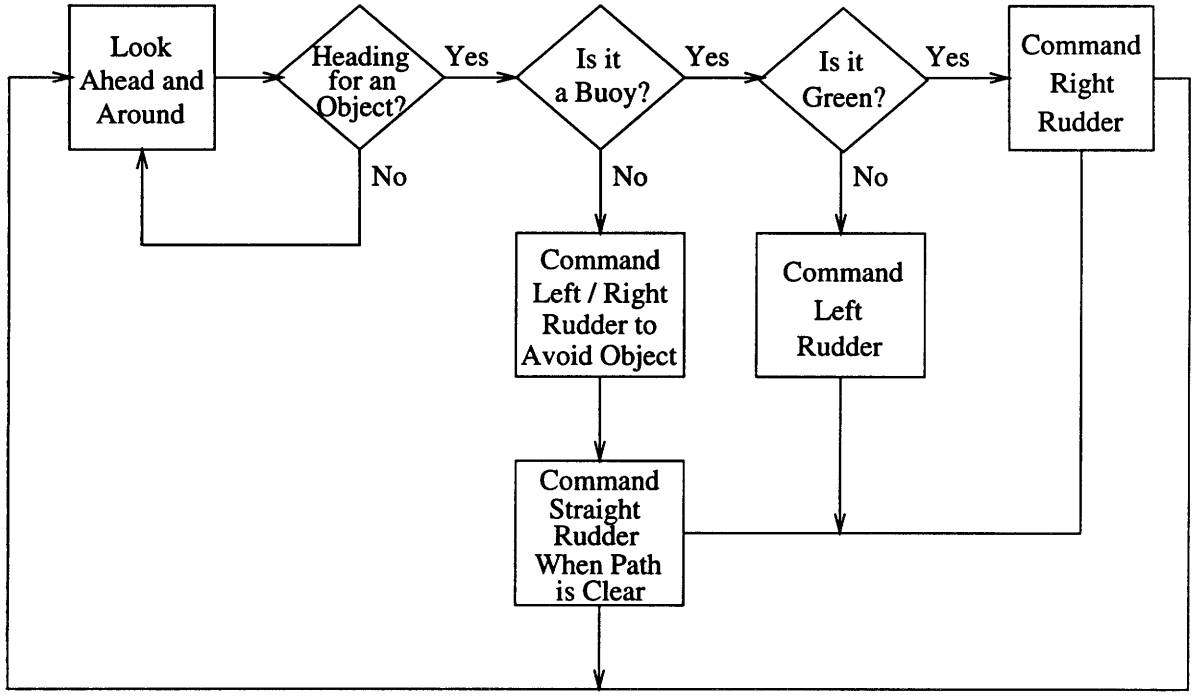


Figure A.7 The OOD must avoid collision with navaids and other surface objects.

Monitoring Water Traffic

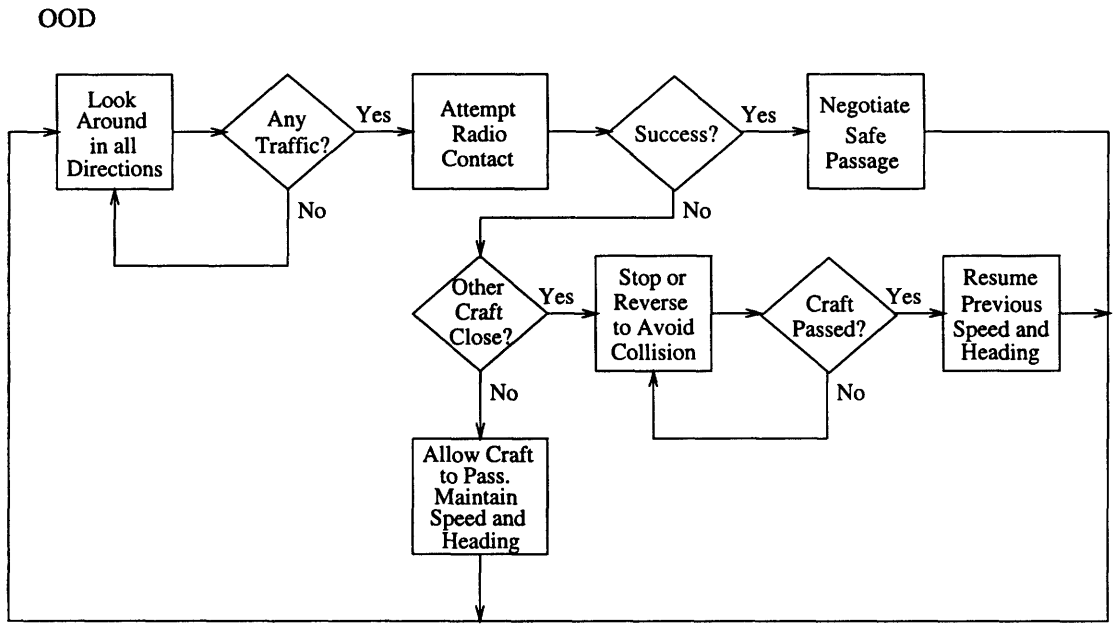


Figure A.8 The OOD must negotiate mutually safe passages with water traffic.

Compensating for Water Currents

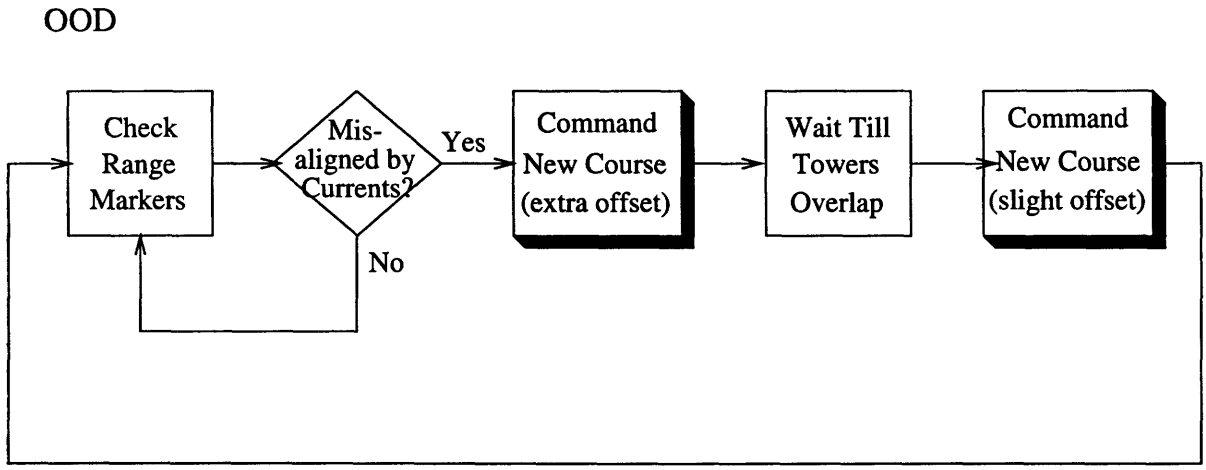


Figure A.9 The OOD must issue heading commands with slight offsets to compensate for water currents.

Appendix B

Channel and Centerline Database

The data below for the 8 straight segments and 3 turn segments in the VE simulating King's Bay is located on beamish.mit.edu in the directory /usr2/vett/ood/data:

FILE: channel.dat

```
CHANNEL_BOUNDARIES 12
8417. 5082. 8417. 5246.      % Format:   X_Left   Y_Left   X_Right   Y_Right
5020. 4951. 5676. 5148.
4987. 4968. 4495. 5361.
2574. 6050. 2820. 6115.
1967. 6427. 2131. 6591.
1639. 7050. 1751. 7426.
1443. 8082. 1607. 8362.
1541. 9788. 1771. 10541.
1492. 11263. 1771. 10558.
1262. 12723. 1344. 13296.
951. 13641. 1328. 13313.
410. 14690. 361. 15149.

CENTERLINES 8
0 8057. 5151. 5676. 5062.      % Format:   Segment   X_C0   Y_C0   X_C1   Y_C1
2 4118. 5444. 2879. 6001.
3 2530. 6193. 2258. 6372.
4 1940. 6734. 1804. 7013.
5 1652. 7484. 1568. 7976.
6 1551. 8606. 1664. 10281.
8 1620. 10986. 1314. 12937.
10 1108. 13538. 631. 14450.
END
```

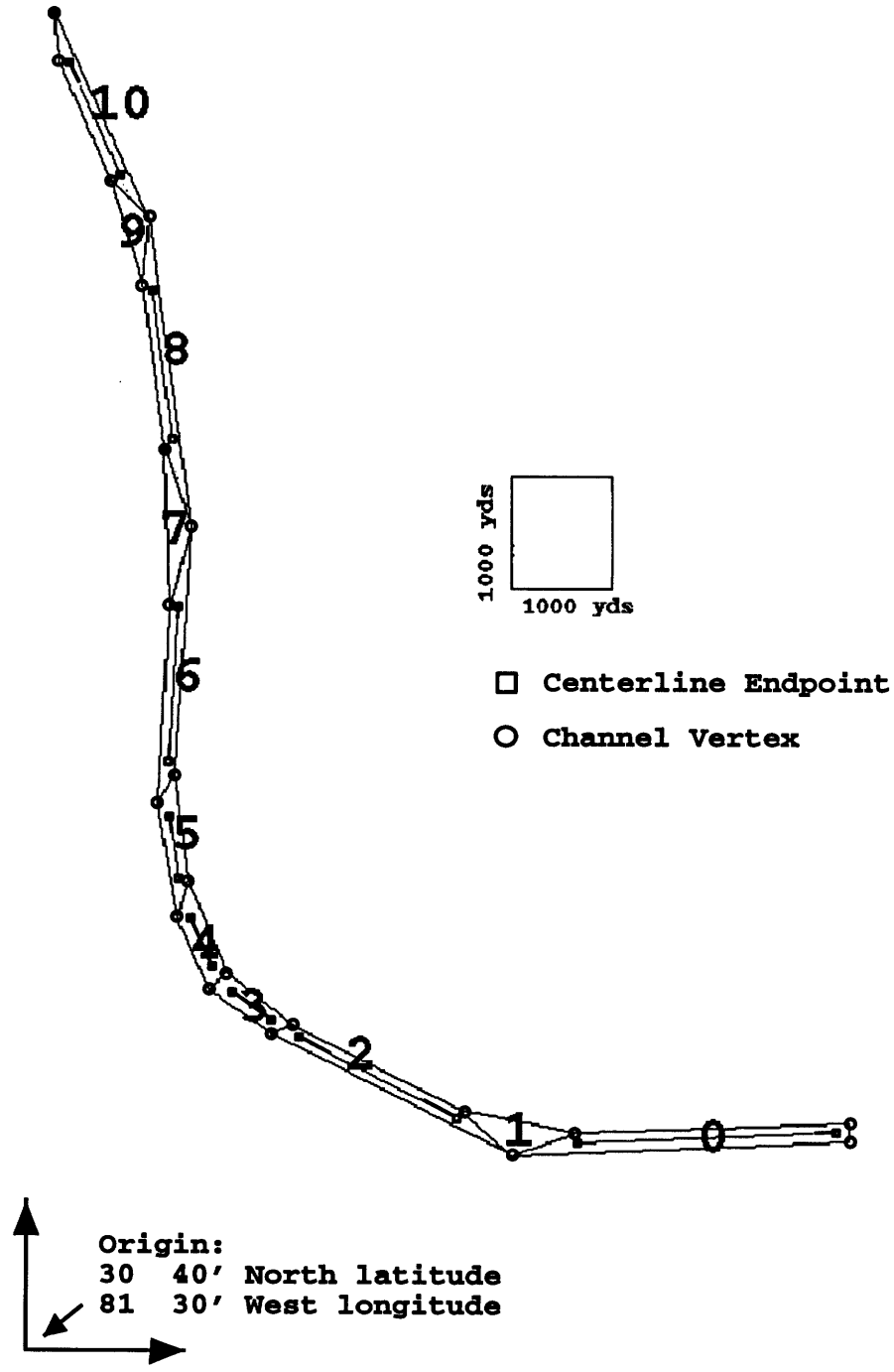


Figure B.1 Configuration of the segments and centerlines in the VE, numbered from 0 to 10 in the inbound direction. The placement of the origin is also shown. The same origin is used in the other environmental databases.

Appendix C

Navaid Input File

The input file for loading the three major types of navaids into the VE is given below. The format for describing a buoy consists of the string "BUOY," followed by the buoy number or letter, a string for the color of the buoy, and the X-Y coordinates of the buoy in yards. Range markers start with the string "RANGE," followed by the marker's height in yards the X-Y coordinates of the marker, and the heading toward which the dayboard must be oriented. The format of turning beacons is similar except for the initial string, "NR" (the naval code for a diamond-shaped red and white daybeacon). The data for these navaids was derived directly from the DMA nautical charts. This file can be found on beamish.mit.edu in the directory /usr2/vett/ood.

FILE: buoys.map

```
BUOY 25 green 7367.0 5050.0
BUOY C yellow 6380.0 4902.0
BUOY E yellow 5495.0 4853.0
BUOY 29 green 4233.0 5295.0
BUOY 31 green 3544.0 5607.0
BUOY 33 green 2508.0 6099.0
BUOY 35 green 1623.0 7050.0
BUOY 37 green 1443.0 8116.0
BUOY 39 green 1508.0 9083.0
BUOY 41 green 1541.0 9771.0
BUOY 43 green 1475.0 11263.0
BUOY 45 green 1393.0 11886.0
BUOY 47 green 1262.0 12755.0
BUOY 49 green 967.0 13657.0
BUOY 51 green 770.0 14018.0
```

```
BUOY 24 red 7367.0 5214.0
BUOY D yellow 6348.0 5295.0
BUOY F yellow 5462.0 5246.0
BUOY 28 red 4692.0 5328.0
BUOY 30 red 3626.0 5755.0
BUOY 32 red 2705.0 6164.0
BUOY 34 red 2262.0 6443.0
BUOY 36 red 1903.0 7092.0
BUOY 38 red 1607.0 8345.0
BUOY 40 red 1672.0 9083.0
BUOY 42 red 1754.0 10591.0
BUOY 44 red 1557.0 11902.0
RANGE 66 1754.0 4915.0 268.0
RANGE 28 3659.0 4986.0 268.0
RANGE 15 622.0 7014.0 294.0
RANGE 30 -168.0 7369.0 294.0
RANGE 10 1401.0 8939.0 350.0
RANGE 20 1346.0 9256.0 350.0
RANGE 10 1416.0 6607.0 4.0
RANGE 20 1386.0 6163.0 4.0
RANGE 10 1209.0 13607.0 351.0
RANGE 30 1002.0 14934.0 351.0
RANGE 20 1451.0 12882.0 332.0
RANGE 34 1793.0 12226.0 332.0
NR 30 4841.0 5718.0 315.0
NR 30 2934.0 6361.0 240.0
NR 30 1016.0 7427.0 0.0
NR 30 754.0 10189.0 0.0
```

Appendix D

Depth Database

The first 15 lines of the water depth database are given below. The first line contains a single integer indicating the number of datapoints to be loaded. Each subsequent line describes a datapoint by giving its floating point X-Y coordinates followed by the depth of that location in feet. Each of the datapoints in this file have been taken directly from numerical depth values indicated on the DMA nautical charts (see Figure 2.8). The approximate resolution of the values is 3 square arc-seconds per datapoint (approx. 100 yds.). The datafile is located on beamish.mit.edu in the directory vett/ood/data.

FILE: depthdata.dat

```
1221
8174.000000 4840.000000 34
8000.000000 4806.000000 38
8000.000000 4873.000000 34
8319.000000 5042.000000 45
8145.000000 5025.000000 48
8000.000000 4974.000000 43
7855.000000 4887.000000 35
7855.000000 5025.000000 45
7681.000000 4991.000000 46
7565.000000 4873.000000 39
7536.000000 5021.000000 49
7333.000000 4890.000000 41
7072.000000 4873.000000 37
8348.000000 5277.000000 34
```

Appendix E

Watercraft Database Input File

The watercraft database describing locations of various types of seagoing vessels external to the submarine is given below. The first field is a string containing the filename of the Performer .flt file to be loaded. Any legal .flt file may be specified, thus other static objects besides watercraft could be loaded here, such as floating cans, hazard markers, or landmarks. In fact, the last line loads a model of Fort Clinch onto the tip of the south bay; this fort is a major landmark used by OODs in King's Bay. The two fields after the filename specify the desired X-Y coordinates of the object in yards. The watercraft below have been placed away from the immediate vicinity of the channel so that they will not cause any undue distraction or concern for collision. The database input file and all of the .flt files specified therein can be found on beamish.mit.edu in the directory vett/ood/data.

FILE: traffic.map

```
sailboat 9855.000000 6554.000000
sailboat 8261.000000 4369.000000
sailboat 6956.000000 6453.000000
sailboat 10087.000000 3025.000000
sailboat 8609.000000 4201.000000
sailboat 6319.000000 7731.000000
galleon 6608.000000 4705.000000
knoxfls 1449.000000 15966.000000
yacht 7275.000000 6050.000000
zodiac 1304.000000 11966.000000
fort 4579.000000 4537.000000
```

Appendix F

Submarine Dynamics Equations

The dynamics model for the Los Angeles fast-attack submarine in Version 1.0 was developed by Levison in [WHLG94] and recoded in C++ by Gupta. Citing directly from this source, the state and control variables included in the equations of motion are:

r	Turn rate, in radians/second
T	Engine thrust
u	Speed of the submarine along the longitudinal body axis, relative to earth, positive forward, in feet/second
v	Speed of the submarine normal to the longitudinal body axis, relative to earth, positive rightward, in feet/second
x	Location east, in feet
y	Location north, in feet
r	Rudder angle, positive for right turn, in radians
Ψ	Compass heading, in radians

The differential equations of motion for the submarine are given below. They have been simplified slightly to take out the effects of water currents since they are not part of Version 1.0.

$$\dot{u} = -X_{u2}u^2 - X_{u3}u^3 + v \cdot r + X_T T$$

$$\dot{v} = -Y_{v2}v^2 - u \cdot r + Y_{\delta r}u^2\delta r$$

$$\dot{r} = -N_r ur + N_{\delta r}u^2\delta r$$

$$\dot{x} = u\cos\Psi - v\sin\Psi$$

$$\dot{y} = u\sin\Psi + v\cos\Psi$$

The $X_{(.)}$, $Y_{(.)}$, and $N_{(.)}$ quantities are constants that depend on the submarine configuration. For the pilot experiments, the following values were used:

$$X_{u2} = 0.0007$$
$$X_{u3} = 0$$
$$Y_{v2} = 1.0$$
$$Y_{\delta r} = 0.001$$
$$N_r = 0.008289$$
$$N_{\delta r} = 4.6e - 07$$

Appendix G

Voice Recognition Software

G.1 HARK Grammar File

FILE: ood.hg

```
<DICT> "ood.hd";
<START> $oodsentence;
$oodsentence: $navcommand | $syscommand;
$navcommand: [ HELM ] [ BRIDGE ] $commandtype;
$commandtype: $rudder | $heading | $speed;
$syscommand: CHART VIEW | SUB VIEW | ZOOM | $binoccmd | $compasscmd |
        COURSE CARD | $buoycommand | $channelcommand | EXIT PROGRAM;
$binoccmd: BINOCULARS/binocs | BINOCS/binocs;
$compasscmd: COMPASS/comp;
$buoycommand: CHART BUOY $digit $digit;
$channelcommand: CHART LEG $digit $digit $digit | CROSS OUT $digit $digit $digit;
$rudder: $direction $angleextent RUDDER | RUDDER AMIDSHIPS/0;
$direction: RIGHT | LEFT;
$angleextent: FIFTEEN/15 DEGREES | STANDARD/15 | FULL/30 | HARD/35;
$heading: $motion $digit $digit $digit | STEADY/-1;
$motion: STEER [ COURSE ] | STEADY ON [ COURSE ];
$speed: [ ALL ] AHEAD $power;
$power: ONE THIRD/4 | TWO THIRDS/8 | STANDARD/12;
$digit: ZERO/0 | ONE/1 | TWO/2 | THREE/3 | FOUR/4 | FIVE/5 | SIX/6 |
        SEVEN/7 | EIGHT/8 | NINE/9;
```

The OOD simulation uses a commercial software voice recognition package called the HARK system, from BBN. HARK consists of a run-time library of recognition commands and a built-in dictionary containing over 100,000 words and their pronunciations. Program-

mers may build their own voice recognition applications by specifying combinations of these dictionary words in a grammar file. The grammar file for the OOD simulation, which can be found on tumtum.mit.edu in the directory /usr/people/vett/hark/ood, is given on the previous page.

G.2 Voice I/O Application Interface

The above grammar is utilized in a C program that serves as the OOD application interface for voice input and output. This program uses the HARK runtime library to set up a waiting loop with callbacks that are executed each time a voice command is recognized or misrecognized. If the voice command is misrecognized, an appropriate message is played asking for a repetition. If recognition is successful, the code extracts the text and numerical tags from the command and parses it into a numerically encoded format to be sent via the blackboard to either the dynamics process or the graphics process. In general, OOD commands for steering the submarine are sent to the dynamics process, while system commands are sent only to the graphics process. The program also plays back the appropriate pre-recorded AIFC file(s) to simulate the helsman's response. The library of AIFC files, located on tumtum.mit.edu in /usr/people/vett/hark/ood/wordsounds, is as follows:

accelerate.aifc	eight.aifc	left.aifc	steadyasshegoes.aifc
allahead.aifc	endsim.aifc	leg.aifc	steer.aifc
allstop.aifc	exit.aifc	nine.aifc	stop.aifc
binoculars.aifc	fifteenrudder.aifc	off.aifc	subview.aifc
bottom.aifc	five.aifc	on.aifc	tenrudder.aifc
bottomold.aifc	fiverudder.aifc	one.aifc	thirtyrudder.aifc
bridgehelmaye.aifc	flank.aifc	onethird.aifc	three.aifc
bridgehelmman.aifc	flybackward.aifc	repeat.aifc	twentyfiverudder.aifc
buoy.aifc	flyforward.aifc	resetviewpoint.aifc	twentyrudder.aifc
channel.aifc	four.aifc	right.aifc	two.aifc
chartview.aifc	full.aifc	rudderamidships.aifc	twothirds.aifc
come.aifc	fullrudder.aifc	seven.aifc	waves.aifc
compass.aifc	hardrudder.aifc	six.aifc	zero.aifc
coursecard.aifc	heading.aifc	standard.aifc	zoom.aifc
crossout.aifc	illegalheading.aifc	steady.aifc	

The C header file and code for the voice interface program, also found on tumtum in the directory /usr/people/vett/hark/ood, are given below:

FILE: ood.h

```
#define TRUE 1
#define FALSE 0

#define NUMLEGS 8
#define TNUMLEGS 2

/* need headings of King's Bay segments for certain commands */
static int inleg[] = { 268,294,303,334,350,4,351,332};
static int outleg[] = { 152,171,184,170,154,123,114,88};

/* need headings of the training bay */
static int tinleg[] = { 90,70 };
static int toutleg[] = { 250,270 };

/* Defines for OOD command types and values */
#define RUDDER_CMDTYPE  0
#define HEADING_CMDTYPE  1
#define SPEED_CMDTYPE  2
#define STANDARD_R  15
#define FULL_R  30
#define HARD_R  35
#define STEADY_HEADING 400
#define ONETHIRD_SPEED  4
#define TWOTHIRDS_SPEED  8
#define STANDARD_SPEED  12
#define FULL_SPEED  16
#define FLANK_SPEED  20
#define STEADY_CMD  -1

/* Defines for graphics command types */
#define STOP_FLYING_CMD  0
#define FLY_FORWARD_CMD  1
#define FLY_BACKWARD_CMD 2
#define CHART_VIEW_CMD  3
#define RESET_VIEW_CMD  4
#define BINOCULARS_CMD  5
#define ACCELERATE_CMD  7
#define SUB_VIEW_CMD  8
#define COMPASS_CMD  9
#define CHANNEL_CMD  11
#define ZOOM_CMD  12
#define WAVES_CMD  13
#define COURSE_CARD_CMD 14
#define CHART_BUOY_CMD 100
#define CHART_LEG_CMD 1000
#define CROSS_OUT_CMD 2000
```

FILE: ood.c

```
/*-------------------------------------------------------------
#
# FILE:   ood.c
#
# HISTORY: created by   Nick Pioch            September 25 1994
# Updates: 9-29-94  N. Pioch    Add "flying" commands for moving the viewpoint
#          11-30-94 N. Pioch    Fix numerical values of rudder commands
#          2-5-95   N. Pioch    Restricted ood.hg grammar to mimimal list of
#                               commands for pilot experiments, taking out
#                               flying as suggested by Levison.  This code
#                               was not changed in case we replace the original
#                               grammar.
#          2-28-95  N. Pioch    Added commands suggested by visiting OOD
#          3-10-95  N. Pioch    Added code to print recognition stats at end
#          4-28-95  N. Pioch    Added code to print avg and max recogn times
#
# PURPOSE: Voice recognition process for Officer of the Deck simulation.
#          Recognizes commands specified in HARK grammar file ood.hg and
#          sends a numerical encoding of the command to the dynamics
#          process and the graphics process via the blackboard.
#
# INSTRUCTIONS:  Make sure the following environment vars are set:
#                  setenv HARK_HOME /usr/local/hark
#                  setenv PATH ${PATH}:$HARK_HOME/bin
#              Type "ood" at the UNIX prompt.
#              Say "EXIT PROGRAM" to exit with recognition stats printed.
#              After making a change in the HARK grammar file ood.hg:
#                  hcompile -m senn16k -o ood-senn16k -r 90 ood.hg
#              After making any corresponding changes to this file:
#                  make ood
#              See ood.params for settable recognition parameters.
-------------------------------------------------------------*/

#include <stdio.h>
#include <sys/types.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include <sys/time.h>
#include <gl/gl.h>
#include <gl/device.h>
#include <hfi.h>
#include "vettnet.h"
#include "ood.h"

#define ABS(x) ((x > 0) ? x : -1.0 * x)

/* blackboard variables */
struct vnt_link   *theLink,*vcmdLink;
struct vnt_class  *theClass,*vcmdClass;
struct vnt_object *theObject,*vcmdObject;
static char buf[VN_STR_LENGTH];

/* for recognition rates */
int total_tries = 0;
```

```
int total_accepts = 0;

/* for keeping track of time between end of speech and successful recog */
double start_time = 0.0;
double end_time = 0.0;
double recog_time = 0.0;
double total_time = 0.0;
double max_time = 0.0;

void after_startup_cb();
void got_recognition_cb();
void no_recognition_cb();
void speech_end_cb();

HFI_RECOGNIZER hr;

double gettime() {
  struct timeval tm;
  double t,dec,sec,temp;
  gettimeofday(&tm);
  sec = (double)(tm.tv_sec);
  dec = (double)(tm.tv_usec);
  temp = dec/1000000.0;
  t = sec+temp;
  return(t);
}

void play_num_sound(int num) {
  switch (num) {
    case 0: system("playaifc wordsounds/zero.aifc"); break;
    case 1: system("playaifc wordsounds/one.aifc"); break;
    case 2: system("playaifc wordsounds/two.aifc"); break;
    case 3: system("playaifc wordsounds/three.aifc"); break;
    case 4: system("playaifc wordsounds/four.aifc"); break;
    case 5: system("playaifc wordsounds/five.aifc"); break;
    case 6: system("playaifc wordsounds/six.aifc"); break;

    case 7: system("playaifc wordsounds/seven.aifc"); break;
    case 8: system("playaifc wordsounds/eight.aifc"); break;
    case 9: system("playaifc wordsounds/nine.aifc"); break;
  }
}

void send_dynamics_command(int cmdtype, int cmdval) {
  char buf[80];

  sprintf(buf,"%d %d\n",cmdtype,cmdval);
  printf(buf);
  vnPut(theLink,theObject,buf);
  end_time = gettime();
}

void send_graphics_command(int cmdtype) {
  char buf[80];

  sprintf(buf,"%d\n",cmdtype);
  printf(buf);
```

```
  vnPut(vcmdLink, vcmdObject, buf);
  end_time = gettime();
}

void play_speed_response(int bellsounding) {
  system("playaifc wordsounds/allahead.aifc");
  /* The full and flank speeds are not used in Version 1.0 */
  switch (bellsounding) {                                                    120
  case ONETHIRD_SPEED: system("playaifc wordsounds/onethird.aifc"); break;
  case TWOTHIRDS_SPEED: system("playaifc wordsounds/twothirds.aifc"); break;
  case STANDARD_SPEED: system("playaifc wordsounds/standard.aifc"); break;
  case FULL_SPEED: system("playaifc wordsounds/full.aifc"); break;
  case FLANK_SPEED: system("playaifc wordsounds/flank.aifc"); break;
  }
}

int check_segment_heading(int *in, int *out, int len, int hdg, int cmd) {
  int c,ileg,oleg;                                                           130
  for (c=0; c<len; c++) {
    ileg = in[c]; oleg = out[c];
    if (hdg==ileg || hdg==oleg) {
      send_graphics_command(cmd + hdg);
      return(TRUE);
    }
  }
  return(FALSE);  /* not found */
}                                                                            140

main(int argc, char *argv[]) {
  /* Allow NAWCTSD to compile to use their own blackboard host */
#ifndef NAWCTSD
  /* First link is for OOD commands, read by dynamics process */
  theLink = vnAddLink("beamish.mit.edu","beamishvoice");
#else
  theLink = vnAddLink("","localvoice");
#endif
  theClass = vnAddClass(theLink,"VoiceClass","integer integer");
  theObject = vnAddInstance(theLink,theClass,"VoiceCommand");                150
  /* Second link is for system commands, read by graphics process */
#ifndef NAWCTSD
  vcmdLink = vnAddLink("beamish.mit.edu","beamishvcmd");
#else
  vcmdLink = vnAddLink("","localvcmd");
#endif

  /* Set up blackboard variables, initialize to -1 (no command yet) */
  vcmdClass = vnAddClass(vcmdLink,"VcmdClass","integer");
  vcmdObject = vnAddInstance(vcmdLink,vcmdClass,"Vcmd");                     160
  vnPut(theLink,theObject,"-1 0");

  /* Set callbacks for the three interesting event types */
  hfi_set_event_callback(HFI_EVENT_STARTUP_DONE, after_startup_cb, NULL);
  hfi_set_event_callback(HFI_EVENT_RECOGNITION, got_recognition_cb, NULL);
  hfi_set_event_callback(HFI_EVENT_NO_RECOGNITION, no_recognition_cb, NULL);
  hfi_set_event_callback(HFI_EVENT_SPEECH_END, speech_end_cb, NULL);

  /* Spawn recognizer and enter main loop to dispatch events to callbacks */
```

```
 hr = hfi_spawn_recognizer("ood-senn16k","ood.params",0,NULL);
 hfi_main_loop();
}

void speech_end_cb(HFI_EVENT event, void *ignore)
{
 start_time = gettime();
}

void after_startup_cb(HFI_EVENT event, void *ignore)
{
 printf("Say next ood command:\n");
 hfi_send_one_shot_cmd(hr,NULL);
}

void no_recognition_cb(HFI_EVENT event, void *ignore)
{
 /* Got voice input, but did not recognize command */
 total_tries++;
 printf("   I didn't understand that\n\n");
 end_time = gettime();
 recog_time = end_time-start_time;
 total_time += recog_time;
 if (recog_time > max_time) max_time = recog_time;
 system("playaifc wordsounds/repeat.aifc");
 printf("Say next ood command:\n");
 hfi_send_one_shot_cmd(hr,NULL);
}

void got_recognition_cb(HFI_EVENT event, void *ignore)
{
 /* Successful recognition of a valid phrase. */
 char tags[80];
 int num,num2,num3,sign,hdg,found;
 char *recognition = hfi_recognition_text(event);

 /* inc the counters */
 total_tries++;
 total_accepts++;
 /* extract the recognized words */
 strcpy(tags,hfi_recognition_tags(recognition));
 printf("Tags are:  %s\n",tags);
 printf("  Command is:  ");

 /* OOD COMMANDS */
 /* Check for rudder command */
 if (strstr(recognition,"RUDDER")) {
  sscanf(tags,"%d",&num);
  if (num==0) {
   send_dynamics_command(RUDDER_CMDTYPE,0);
   system("playaifc wordsounds/rudderamidships.aifc");
   system("playaifc wordsounds/bridgehelmaye.aifc");
  } else {
   if (strstr(recognition,"LEFT")) {
       send_dynamics_command(RUDDER_CMDTYPE,-num);
       system("playaifc wordsounds/left.aifc ");
   } else {
```

```
        send_dynamics_command(RUDDER_CMDTYPE,num);
        system("playaifc wordsounds/right.aifc ");
    }
    /* The 5,10,20, and 25 degree settings are not used in Version 1.0 */
    switch (num) {
    case 5: system("playaifc wordsounds/fiverudder.aifc"); break;
    case 10: system("playaifc wordsounds/tenrudder.aifc"); break;
    case STANDARD_R: system("playaifc wordsounds/fifteenrudder.aifc"); break;
    case 20: system("playaifc wordsounds/twentyrudder.aifc"); break;
    case 25: system("playaifc wordsounds/twentyfiverudder.aifc"); break;
    case FULL_R: system("playaifc wordsounds/fullrudder.aifc"); break;
    case HARD_R: system("playaifc wordsounds/hardrudder.aifc"); break;
    }
    system("playaifc wordsounds/bridgehelmaye.aifc ");
  }

/* Check for general command to come left or right (not used in V1.0) */
} else if (strstr(recognition,"COME")) {
  if (strstr(recognition,"LEFT")) {
    send_dynamics_command(RUDDER_CMDTYPE, -5);
    system("playaifc wordsounds/come.aifc");
    system("playaifc wordsounds/left.aifc");
  } else {
    send_dynamics_command(RUDDER_CMDTYPE, 5);
    system("playaifc wordsounds/come.aifc");
    system("playaifc wordsounds/right.aifc");
  }

/* check for speed command */
} else if (strstr(recognition,"AHEAD")) {
  sscanf(tags,"%d",&num);
  send_dynamics_command(SPEED_CMDTYPE, num);
  play_speed_response(num);
  system("playaifc wordsounds/bridgehelmaye.aifc ");
  sginap(100);  /* Pause before extra response for maneuvering answer */
  system("playaifc wordsounds/bridgehelmman.aifc ");
  play_speed_response(num);

/* check for heading command */
} else if (strstr(recognition,"STEADY") || (strstr(recognition,"STEER"))) {
  sscanf(tags,"%d",&num);
  switch (num) {
  case STEADY_CMD:
    send_dynamics_command(HEADING_CMDTYPE, STEADY_HEADING);
    system("playaifc wordsounds/steady.aifc");
    system("playaifc wordsounds/bridgehelmaye.aifc");
    break;
  default:
    sscanf(tags,"%d %d %d",&num,&num2,&num3);
    hdg = num*100+num2*10+num3;
    if (hdg>360 || hdg<0) {
        end_time = gettime();
        printf("Illegal heading!\n");
        system("playaifc wordsounds/illegalheading.aifc");
    } else {
        send_dynamics_command(HEADING_CMDTYPE,hdg);
        if (strstr(recognition,"STEADY")) {
```

```
        system("playaifc wordsounds/steady.aifc");
        system("playaifc wordsounds/on.aifc ");
      } else {
        system("playaifc wordsounds/steer.aifc ");
      }
      play_num_sound(num);
      play_num_sound(num2);
      play_num_sound(num3);
      system("playaifc wordsounds/bridgehelmaye.aifc ");
    }
  }

/* SYSTEM COMMANDS */
/* Check for chart buoy or leg  commands */
} else if (strstr(recognition,"BUOY")) {
  sscanf(tags,"%d %d",&num,&num2);
  num3 = num*10+num2;
  if (num3<1 || num3>100) {
    end_time = gettime();
    printf("Illegal buoy number!\n");
  } else {
    send_graphics_command(CHART_BUOY_CMD + num*10+num2);
    system("playaifc wordsounds/buoy.aifc");
    play_num_sound(num);
    play_num_sound(num2);
  }
} else if (strstr(recognition,"LEG")) {
  /* Make sure the leg's heading is valid heading on chart */
  found = FALSE;
  sscanf(tags,"%d %d %d",&num,&num2,&num3);
  hdg = num*100+num2*10+num3;
  /* search thru both the training headings & Kings bay headings */
  found = check_segment_heading(inleg,outleg,NUMLEGS,hdg,CHART_LEG_CMD);
  if (!found)
    found = check_segment_heading(tinleg,toutleg,TNUMLEGS,hdg,CHART_LEG_CMD);
  if (found) {
    system("playaifc wordsounds/leg.aifc");
    play_num_sound(num);
    play_num_sound(num2);
    play_num_sound(num3);
  } else {
    /* Play error feedback if not valid heading */
    end_time = gettime();
    printf("Illegal segment heading!\n");
    system("playaifc wordsounds/illegalheading.aifc");
  }

/* Check for crossing out a course card line */
} else if (strstr(recognition,"CROSS")) {
  /* Make sure heading is valid heading from course card */
  found = FALSE;
  sscanf(tags,"%d %d %d",&num,&num2,&num3);
  hdg = num*100+num2*10+num3;
  /* search thru both the training headings & Kings bay headings */
  found = check_segment_heading(inleg,outleg,NUMLEGS,hdg,CROSS_OUT_CMD);
  if (!found)
    found = check_segment_heading(tinleg,toutleg,TNUMLEGS,hdg,CROSS_OUT_CMD);
```

```
  if (found) {
    system("playaifc wordsounds/crossout.aifc");
    play_num_sound(num);
    play_num_sound(num2);
    play_num_sound(num3);
  } else {
    /* Play error feedback if not valid heading */
    end_time = gettime();
    printf("Illegal segment heading!\n");
    system("playaifc wordsounds/illegalheading.aifc");
  }

/* Check for flying commands to move viewpoint (not used in Version 1.0) */
} else if (strstr(recognition,"STOP")) {      /* not used in Version 1.0 */
  send_graphics_command(STOP_FLYING_CMD);
  system("playaifc wordsounds/stop.aifc");
} else if (strstr(recognition,"BACKWARD")) {  /* not used in Version 1.0 */
  send_graphics_command(FLY_BACKWARD_CMD);
  system("playaifc wordsounds/flybackward.aifc");
/* NOTE: BACKWARD must come before FLY in the if statement since
        FLY alone means go forward */
} else if (strstr(recognition,"FLY")) {       /* not used in Version 1.0 */
  send_graphics_command(FLY_FORWARD_CMD);
  system("playaifc wordsounds/flyforward.aifc");
} else if (strstr(recognition,"ACCELERATE")) { /* not used in Version 1.0 */
  send_graphics_command(ACCELERATE_CMD);
  system("playaifc wordsounds/accelerate.aifc");

/* Check for view change commands and physical aids */
} else if (strstr(recognition,"CHART")) {
  send_graphics_command(CHART_VIEW_CMD);
  system("playaifc wordsounds/chartview.aifc");
} else if (strstr(recognition,"SUB")) {
  send_graphics_command(SUB_VIEW_CMD);
  system("playaifc wordsounds/subview.aifc");
} else if (strstr(recognition,"VIEWPOINT")) { /* not used in Version 1.0 */
  send_graphics_command(RESET_VIEW_CMD);
  system("playaifc wordsounds/resetviewpoint.aifc");
} else if (strstr(recognition,"binocs")) {
  send_graphics_command(BINOCULARS_CMD);
  system("playaifc wordsounds/binoculars.aifc");
} else if (strstr(recognition,"comp")) {
  send_graphics_command(COMPASS_CMD);
  system("playaifc wordsounds/compass.aifc");
} else if (strstr(recognition,"CHANNEL")) {   /* not used in Version 1.0 */
  send_graphics_command(CHANNEL_CMD);
  system("playaifc wordsounds/channel.aifc");
} else if (strstr(recognition,"ZOOM")) {
  send_graphics_command(ZOOM_CMD);
  system("playaifc wordsounds/zoom.aifc");
} else if (strstr(recognition,"WAVES")) {     /* not used in Version 1.0 */
  send_graphics_command(WAVES_CMD);
  system("playaifc wordsounds/waves.aifc");
} else if (strstr(recognition,"CARD")) {
  send_graphics_command(COURSE_CARD_CMD);
  system("playaifc wordsounds/coursecard.aifc");
} else if (strstr(recognition,"EXIT")) {
```

```
    printf("EXIT\n");
    end_time = gettime();
    system("playaifc wordsounds/exit.aifc");                                    400
    /* Print out recognition statistics */
    printf("Commands given:      %d\n",total_tries);
    printf("Commands recognized:      %d\n",total_accepts);
    printf("Average recognition time:      %f\n",total_time/total_tries);
    printf("Maximum recognition time:      %f\n",max_time);
    if (total_tries)
      printf("Recognition rate:  %3.2f per cent\n",
             100.0*((float)total_accepts/(float)total_tries));
    exit(0);
  }                                                                             410
  /* Compute recognition time for this command; add to total, check if max*/
  recog_time = end_time-start_time;
  total_time += recog_time;
  if (recog_time > max_time) max_time = recog_time;
  printf("\nRecognition was:  %s\n",recognition);
  printf("\nSay next ood command:\n");
  hfi_send_one_shot_cmd(hr,NULL);
}
```

Appendix H

Environmental Database Validation

H.1 Channel Segments and Centerlines

This section verifies the locations of the channel segments and centerlines in the environmental database. Only segments with centerlines are verified; turn segments 1, 7, and 9 are not included since they have no corresponding rows in the course card. Refer to Appendix B for a printout of the channel and centerline database.

H.1.1 Verification of Segment and Centerline Headings

The calculations below verify that the centerlines and channel boundaries in the database lie at an angle equivalent to the corresponding course card heading. Errors of two degrees are allowed for the channel boundaries, while the computed headings of centerlines must round to the integer value of the course card. All computed headings below meet these criteria.

Segment 0:
Heading from Course Card: **268**
Left side:
$L_0 = (8417.0, 5082.0)$
$L_1 = (5020.0, 4951.0)$
$\Delta X_L = -3397.0$

$\Delta Y_L = -131.0$
$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 182.208$ (Quadrant III)
True Heading $= 90 - \theta_L = -92.208 =$ **267.792**
Right side:
$R_0 = (8417.0, 5246.0)$
$R_1 = (5676.0, 5148.0)$
$\Delta X_R = -2741.0$
$\Delta Y_R = -98.0$
$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 182.048$ (Quadrant III)
True Heading $= 90 - \theta_R = -92.048 =$ **267.952**
Centerline:
$C_0 = (8057.0, 5151.0)$
$C_1 = (5676.0, 5062.0)$
$\Delta X_C = -2381.0$
$\Delta Y_C = -89.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 182.141$ (Quadrant III)
True Heading $= 90 - \theta_C = -92.141 =$ **267.859**

Segment 2:
Heading from Course Card: **294**
Left side:
$\Delta X_L = -2413.0$
$\Delta Y_L = 1082.0$
$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 155.848$ (Quadrant II)
True Heading $= 90 - \theta_L = -65.848 =$ **294.152**
Right side:
$\Delta X_R = -1675.0$
$\Delta Y_R = 754.0$
$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 155.765$ (Quadrant II)
True Heading $= 90 - \theta_R = -65.765 =$ **294.235**
Centerline:
$\Delta X_C = -1239.0$
$\Delta Y_C = 557.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 155.793$ (Quadrant II)
True Heading $= 90 - \theta_C = -65.793 =$ **294.207**

Segment 3:
Heading from Course Card: **303**
Left side:
$\Delta X_L = -607.0$
$\Delta Y_L = 377.0$
$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 148.156$ (Quadrant II)
True Heading $= 90 - \theta_L = -58.156 =$ **301.844**
Right side:
$\Delta X_R = -689.0$
$\Delta Y_R = 476.0$

$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 145.361$ (Quadrant II)
True Heading $= 90 - \theta_R = -55.361 =$ **304.639**
Centerline:
$\Delta X_C = -272.0$
$\Delta Y_C = 179.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 146.652$ (Quadrant II)
True Heading $= 90 - \theta_C = -56.652 =$ **303.348**

Segment 4:
Heading from Course Card: **334**
Left side:
$\Delta X_L = -328.0$
$\Delta Y_L = 623.0$
$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 117.766$ (Quadrant II)
True Heading $= 90 - \theta_L = -27.766 =$ **332.234**
Right side:
$\Delta X_R = -380.0$
$\Delta Y_R = 835.0$
$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 114.470$ (Quadrant II)
True Heading $= 90 - \theta_R = -24.470 =$ **335.530**
Centerline:
$\Delta X_C = -136.0$
$\Delta Y_C = 279.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 115.987$ (Quadrant II)
True Heading $= 90 - \theta_C = -25.987 =$ **334.013**

Segment 5:
Heading from Course Card: **350**
Left side:
$\Delta X_L = -196.0$
$\Delta Y_L = 1032.0$
$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 100.754$ (Quadrant II)
True Heading $= 90 - \theta_L = -10.754 =$ **349.246**
Right side:
$\Delta X_R = -143.0$
$\Delta Y_R = 936.0$
$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 98.686$ (Quadrant II)
True Heading $= 90 - \theta_R = -8.686 =$ **351.314**
Centerline:
$\Delta X_C = -84.0$
$\Delta Y_C = 492.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 99.689$ (Quadrant II)
True Heading $= 90 - \theta_C = -9.689 =$ **350.311**

Segment 6:

Heading from Course Card: **004**

Left side:

$\Delta X_L = 98.0$

$\Delta Y_L = 1706.0$

$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 86.712$ (Quadrant I)

True Heading $= 90 - \theta_L =$ **3.288**

Right side:

$\Delta X_R = 164.0$

$\Delta Y_R = 2179.0$

$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 85.696$ (Quadrant I)

True Heading $= 90 - \theta_R =$ **4.304**

Centerline:

$\Delta X_C = 113.0$

$\Delta Y_C = 1675.0$

$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 86.141$ (Quadrant I)

True Heading $= 90 - \theta_C =$ **3.859**

Segment 8:

Heading from Course Card: **351**

Left side:

$\Delta X_L = -230.0$

$\Delta Y_L = 1460.0$

$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 98.952$ (Quadrant II)

True Heading $= 90 - \theta_L = -8.952 =$ **351.048**

Right side:

$\Delta X_R = -427.0$

$\Delta Y_R = 2738.0$

$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 98.864$ (Quadrant II)

True Heading $= 90 - \theta_R = -8.864 =$ **351.136**

Centerline:

$\Delta X_C = -306.0$

$\Delta Y_C = 1951.0$

$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 98.914$ (Quadrant II)

True Heading $= 90 - \theta_C = -8.914 =$ **351.086**

Segment 10:

Heading from Course Card: **332**

Left side:

$\Delta X_L = -541.0$

$\Delta Y_L = 1049.0$

$\theta_R = \arctan \frac{\Delta Y_L}{\Delta X_L} = 117.281$ (Quadrant II)

True Heading $= 90 - \theta_L = -27.281 =$ **332.719**

Right side:

$\Delta X_R = -967.0$

$\Delta Y_R = 1836.0$

$\theta_R = \arctan \frac{\Delta Y_R}{\Delta X_R} = 117.775$ (Quadrant II)
True Heading $= 90 - \theta_R = -27.775 =$ **332.225**
Centerline:
$\Delta X_C = -477.0$
$\Delta Y_C = 912.0$
$\theta_C = \arctan \frac{\Delta Y_C}{\Delta X_C} = 117.611$ (Quadrant II)
True Heading $= 90 - \theta_C = -27.611 =$ **332.389**

H.1.2 Verification of Centerline Placement

This section checks that each centerline is placed along the lengthwise geometric bisection of its corresponding segment. The X-coordinates of each pair of centerline endpoints in the database should be substituted into the equation of the bisector of the appropriate segment; if the result does not match the Y-coordinate of the endpoint in the database within 5 yds., a new endpoint must be recomputed by moving the original endpoint to the closest point on the bisector. All centerline endpoints successfully meet this criterion, as shown below.

Segment 0:

$M_0 = \frac{L_0+R_0}{2} = \frac{(8417.0+8417.0,5082.0+5046.0)}{2} = (8417.0, 5164.0)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(5020.0+5676.0,4951.0+5148.0)}{2} = (5348.0, 5049.5)$

Centerline Equation: $y = 0.0373(x - 8417.0) + 5164.0$

Test $C_0 = (8057.0, 5151.0)$ in equation:
$y = 0.0373(8057.0 - 8417.0) + 5164.0$
$y = 0.0373(-360.0) + 5164.0$
$y = 5150.6$

Test $C_1 = (5676.0, 5062.0)$ in equation:
$y = 0.0373(5676.0 - 8417.0) + 5164.0$
$y = 0.0373(-2741.0) + 5164.0$
$y = 5061.8$

Segment 2:

$M_0 = \frac{L_0+R_0}{2} = \frac{(4987.0+4495.0,4968.0+5361.0)}{2} = (4741.0, 5164.5)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(2574.0+2820.0,6050.0+6115.0)}{2} = (2697.0, 6082.5)$

Centerline Equation: $y = -0.4489(x - 4741.0) + 5164.5$

Test $C_0 = (4118.0, 5444.0)$ in equation:
$y = -0.4489(4118.0 - 4741.0) + 5164.5$
$y = -0.4489(-623.0) + 5164.5$
$y = 5444.2$

Test $C_1 = (2879.0, 6001.0)$ in equation:
$y = -0.4489(2879.0 - 4741.0) + 5164.5$
$y = -0.4489(-1862.0) + 5164.5$
$y = 6000.3$

Segment 3:

$M_0 = \frac{L_0+R_0}{2} = \frac{(2574.0+2820.0,6050.0+6115.0)}{2} = (2697.0, 6082.5)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(1967.0+2131.0,6427.0+6591.0)}{2} = (2049.0, 6509.0)$

Centerline Equation: $y = -0.6582(x - 2697.0) + 6082.5$

Test $C_0 = (2530.0, 6193.0)$ in equation:
$y = -0.6582(2530.0 - 2697.0) + 6082.5$
$y = -0.6582(-167.0) + 6082.5$
$y = 6192.4$

Test $C_1 = (2258.0, 6372.0)$ in equation:
$y = -0.6582(2258.0 - 2697.0) + 6082.5$
$y = -0.6582(-439.0) + 6082.5$
$y = 6371.4$

Segment 4:

$M_0 = \frac{L_0+R_0}{2} = \frac{(1967.0+2131.0,6427.0+6591.0)}{2} = (2049.0, 6509.0)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(1639.0+1751.0,7050.0+7426.0)}{2} = (1695.0, 7238.0)$

Centerline Equation: $y = -2.0593(x - 2049.0) + 6509.0$

Test $C_0 = (1940.0, 6734.0)$ in equation:
$y = -2.0593(1940.0 - 2049.0) + 6509.0$
$y = -2.0593(-109.0) + 6509.0$
$y = 6733.4$

Test $C_1 = (1804.0, 7013.0)$ in equation:
$y = -2.0593(1804.0 - 2049.0) + 6509.0$
$y = -2.0593(-245.0) + 6509.0$
$y = 7013.5$

Segment 5:

$$M_0 = \frac{L_0+R_0}{2} = \frac{(1639.0+1751.0,7050.0+7426.0)}{2} = (1695.0,\ 7238.0)$$
$$M_1 = \frac{L_1+R_1}{2} = \frac{(1443.0+1607.0,8082.0+8362.0)}{2} = (1525.0,\ 8222.0)$$

Centerline Equation: $y = -5.7882(x - 1695.0) + 7238.0$

Test $C_0 = (1652.0,\ 7484.0)$ in equation:
$y = -5.7882(1652.0 - 1695.0) + 7238.0$
$y = -5.7882(-43.0) + 7238.0$
$y = 7486.9$

Test $C_1 = (1568.0,\ 7976.0)$ in equation:
$y = -5.7882(1568.0 - 1695.0) + 7238.0$
$y = -5.7882(-127.0) + 7238.0$
$y = 7973.1$

Segment 6:

$$M_0 = \frac{L_0+R_0}{2} = \frac{(1443.0+1607.0,8082.0+8362.0)}{2} = (1525.0,\ 8222.0)$$
$$M_1 = \frac{L_1+R_1}{2} = \frac{(1541.0+1771.0,9788.0+10541.0)}{2} = (1656.0,\ 10164.5)$$

Centerline Equation: $y = 14.828(x - 1525.0) + 8222.0$

Test $C_0 = (1551.0,\ 8606.0)$ in equation:
$y = 14.828(1551.0 - 1525.0) + 8222.0$
$y = 14.828(26.0) + 8222.0$
$y = 8607.5$

Test $C_1 = (1664.0,\ 10281.0)$ in equation:
$y = 14.828(1664.0 - 1525.0) + 8222.0$
$y = 14.828(139.0) + 8222.0$
$y = 10283.1$

Segment 8:

$$M_0 = \frac{L_0+R_0}{2} = \frac{(1492.0+1771.0,11263.0+10558.0)}{2} = (1631.5,\ 10910.5)$$
$$M_1 = \frac{L_1+R_1}{2} = \frac{(1262.0+1344.0,12723.0+13296.0)}{2} = (1303.0,\ 13009.5)$$

Centerline Equation: $y = -6.3896(x - 1631.5) + 10910.5$

Test $C_0 = (1620.0,\ 10986.0)$ in equation:
$y = -6.3896(1620.0 - 1631.5) + 10910.5$
$y = -6.3896(-11.5) + 10910.5$
$y = 10984.0$

Test C_1 = (1314.0, 12937.0) in equation:
$y = -6.3896(1314.0 - 1631.5) + 10910.5$
$y = -6.3896(-2741.0) + 10910.5$
$y = 12939.2$

Segment 10:

$M_0 = \frac{L_0+R_0}{2} = \frac{(951.0+1328.0,13641.0+13313.0)}{2} = (1139.5, 13477.0)$
$M_1 = \frac{L_1+R_1}{2} = \frac{(410.0+361.0,14690.0+15149.0)}{2} = (385.5, 14919.5)$

Centerline Equation: $y = -1.9131(x - 1139.5) + 13477.0$

Test C_0 = (1108.0, 13538.0) in equation:
$y = -1.9131(1108.0 - 1139.5) + 13477.0$
$y = -1.9131(-360.0) + 13477.0$
$y = 13537.3$

Test C_1 = (631.0, 14450.0) in equation:
$y = -1.9131(631.0 - 1139.5) + 13477.0$
$y = -1.9131(-508.5) + 13477.0$
$y = 14449.8$

H.1.3 Centerline Distances

The following calculations verify that the endpoints of each centerline are at least 200 yds. from the ends of the segment. The endpoints that do not meet this criterion must be next to a turn segment (1,7, or 9) so that the trainee will have ample time to complete a turn before centerline deviation is measured again. These cases occur at the end of Segment 6, both ends of Segment 8, and the start of Segment 10. In fact, these points are the only ones below that do not meet the 200 yard distance minimum.

Segment 0:

$|M_0 - C_0| = \sqrt{(8417.0 - 8057.0)^2 + (5064.0 - 5151.0)^2} = 360$
$|M_1 - C_1| = \sqrt{(5348.0 - 5676.0)^2 + (5049.5 - 5062.0)^2} = 328.23$

Segment 2:

$|M_0 - C_0| = \sqrt{(4741.0 - 4118.0)^2 + (5164.5 - 5444.0)^2} = 682.824$
$|M_1 - C_1| = \sqrt{(2697.0 - 2879.0)^2 + (6082.5 - 6001.0)^2} = 199.41$

Segment 3:

$$|M_0 - C_0| = \sqrt{(2697.0 - 2530.0)^2 + (6082.5 - 6193.0)^2} = 200.25$$

$$|M_1 - C_1| = \sqrt{(2049.0 - 2258.0)^2 + (6509.0 - 6372.0)^2} = 249.90$$

Segment 4:

$$|M_0 - C_0| = \sqrt{(2049.0 - 1940.0)^2 + (6509.0 - 6734.0)^2} = 250.01$$

$$|M_1 - C_1| = \sqrt{(1695.0 - 1804.0)^2 + (7238.0 - 7013.0)^2} = 250.01$$

Segment 5:

$$|M_0 - C_0| = \sqrt{(1695.0 - 1652.0)^2 + (7238.0 - 7484.0)^2} = 249.73$$

$$|M_1 - C_1| = \sqrt{(1525.0 - 1568.0)^2 + (8222.0 - 7976.0)^2} = 249.73$$

Segment 6:

$$|M_0 - C_0| = \sqrt{(1525.0 - 1551.0)^2 + (8222.0 - 8606.0)^2} = 384.88$$

$$|M_1 - C_1| = \sqrt{(1656.0 - 1664.0)^2 + (10164.5 - 10281.0)^2} = 116.77$$

Segment 8:

$$|M_0 - C_0| = \sqrt{(1631.5 - 1620.0)^2 + (10910.5 - 10986.0)^2} = 76.37$$

$$|M_1 - C_1| = \sqrt{(1303.0 - 1314.0)^2 + (13009.5 - 12937.0)^2} = 73.33$$

Segment 10:

$$|M_0 - C_0| = \sqrt{(1139.5 - 1108.0)^2 + (13477.0 - 13538.0)^2} = 68.65$$

$$|M_1 - C_1| = \sqrt{(385.5 - 631.0)^2 + (14919.5 - 14450.0)^2} = 529.81$$

H.2 Channel Buoys

Since the main function of the buoys in the VE is to mark the boundaries of the channel segments, each buoy location read from the DMA charts must be checked to be within 20 yds. of the nearest boundary or channel vertex. Any buoy not within this threshold must be moved perpendicularly toward the boundary until the distance is exactly 20 yds. (allowing for some original deviation suggested by the chart). If a buoy is meant to be on a vertex than the simple Euclidean distance between the buoy and the vertex must be less than 20 yds. Table H.1 calculates this distance to the channel boundary for the buoys on the left going inbound, while Table H.2 gives the distances for buoys on the right. The columns of the table are based on Figure 4.1, with the rightmost column, $|BR_1| \sin\theta$, giving the perpendicular buoy-to-channel distance. Table H.3 is for buoys located on channel vertices,

checking the simple Euclidean distance between the buoy and the channel vertex. The buoy locations were taken from the navaid database in Appendix C. The yellow buoys marking the turning basin are not included in the verification.

Table H.1: Verification of green buoys.

Buoy #	$\overline{BR_1}$	$\overline{R_0R_1}$	$\lvert BR_1\rvert$	$\lvert R_0R_1\rvert$	θ	$\lvert BR_1\rvert \sin\theta$
25	(-2347.0, -99.0)	(-3397.0, -131.0)	2349.1	3399.5	0.175	7.2
29	(-1659.0, 755.0)	(-2413.0, 1082.0)	1822.7	2644.5	0.269	8.6
31	(-970.0, 443.0)	(-2413.0, 1082.0)	1066.4	2644.5	0.608	11.3
39	(33.0, 705.0)	(98.0, 1706.0)	705.8	1708.8	0.763	9.40
45	(-131.0, 837.0)	(-230.0, 1460.0)	847.2	1478.0	0.247	3.7
51	(-360.0, 672.0)	(-541.0, 1049.0)	762.4	1180.3	1.12	14.9

Table H.2: Verification of red buoys.

Buoy #	$\overline{BR_1}$	$\overline{R_0R_1}$	$\lvert BR_1\rvert$	$\lvert R_0R_1\rvert$	θ	$\lvert BR_1\rvert \sin\theta$
24	(-1691.0, -66.0	(-2741.0, -98.0)	1692.3	2742.8	0.447	13.2
28	(-197.0, 33.0)	(-1181.0, 213.0)	199.74	1200.05	0.573	2.0
30	(-806.0, 360.0)	(-1675.0, 754.0)	882.74	1836.88	0.157	2.4
32	(-574.0, 427.0)	(-689.0, 476.0)	715.41	837.43	2.008	25.1
34	(-131.0, 148.0)	(-689.0, 476.0)	197.65	837.43	13.848	47.3
36	(-152.0, 334.0)	(-380.0, 835.0)	366.96	917.40	0.005	0.0
40	(99.0, 1458.0)	(164.0, 2179.0)	1461.36	2185.16	0.424	2.72
44	(-213.0, -1394.0)	(-427.0, 2738.0)	1410.18	2771.10	0.212	5.22

Table H.3: Verification of buoys on channel vertices.

Buoy #	Color	Buoy Location	Vertex Location	Distance from Vertex
33	green	(2508.0, 6099.0)	(2574.0, 6050.0)	82.2
35	green	(1623.0, 7050.0)	(1639.0, 7050.0)	16.0
37	green	(1443.0, 8116.0)	(1443.0, 8082.0)	34.0
41	green	(1541.0, 9771.0)	(1541.0, 9788.0)	17.0
43	green	(1475.0, 11263.0)	(1492.0, 11263.0)	17.0
47	green	(1262.0, 12755.0)	(1262.0, 12723.0)	32.0
49	green	(967.0, 13657.0)	(951.0, 13641.0)	22.6
38	red	(1607.0, 8345.0)	(1607.0, 8362.0)	17.0
42	red	(1754.0, 10591.0)	(1771.0, 10558.0)	37.1

The results show that the red buoys numbered 32 and 34 exceed the 20-yard threshold distance from the channel boundary, and buoys 33, 37, 42, 47, and 49 are more than 20 yds. from their associated channel vertices. However, the DMA chart shows buoys 34 and 36 as noticeably offset toward the inside of the channel, so unless a source more recent or reliable indicates they should be on the boundary, they may be left alone. However, the vertex buoys should be moved to their corresponding channel vertices before further experiments.

H.3 Range Markers

This section shows the calculations for checking the perpendicular distance of each range marker from the extended centerline of its corresponding segment. If this distance exceeds 2 yds., the range marker must be given a new location at the vector projection into the extended centerline of a vector from C_1 to the original marker location. All range markers in the VE meet this 2-yard threshold except for the pair used for Segment 5, at a heading of 350. The updated locations for these two markers are also listed, along with a check that they do indeed meet the 2-yard requirement.

Segment 0:

$C_0 = (8057.0, 5151.0)$
$C_1 = (5676.0, 5062.0)$

Centerline Equation: $y = 0.0374(x - 8057.0) + 5151.0$

Test $Rg_1 = (3659.0, 4986.0)$ in equation:
$y = 0.0374(3659.0 - 8057.0) + 5151.0$
$y = 0.0374(-4398.0) + 5151.0$
$y = 4986.5$
$d = |(4986.5 - 4986.0)| \sin 88° = 0.5$

Test $Rg_2 = (1754.0, 4915.0)$ in equation:
$y = 0.0374(1754.0 - 8057.0) + 5151.0$
$y = 0.0374(-6063.0) + 5151.0$
$y = 4915.3$
$d = |(4915.3 - 4915.3)| \sin 88° = 0.3$

Segment 2:

$C_0 = (4118.0, 5444.0)$
$C_1 = (2879.0, 6001.0)$

Centerline Equation: $y = -0.4496(x - 4118.0) + 5444.0$

Test $Rg_1 = (622.0, 7014.0)$ in equation:
$y = -0.4496(622.0 - 4118.0) + 5444.0$
$y = -0.4496(-3496.0) + 5444.0$
$y = 7014.8$
$d = |(7014.8 - 7014.0)| \sin 66° = 0.7$

Test $Rg_2 = (-168.0, 7369.0)$ in equation:
$y = -0.4496(-168.0 - 4118.0) + 5444.0$
$y = -0.4496(-4286.0) + 5444.0$
$y = 7371.0$
$d = |(7371.0 - 7369.0)| \sin 66° = 1.8$

Segment 5:

$C_0 = (1652.0, 7484.0)$
$C_1 = (1568.0, 7976.0)$

Centerline Equation: $y = -5.857(x - 1652.0) + 7484.0$

Test $Rg_1 = (1401.0, 8939.0)$ in equation:
$y = -5.857(1401.0 - 1652.0) + 7484.0$
$y = -5.857(-251.0) + 7484.0$
$y = 8954.1$
$d = |(8954.1 - 8939.0)| \sin 10° = 2.6$
New Point $Rg_1' = (1403.6, 8939.5)$
Recheck:
$y' = -5.857(1403.6 - 1652.0) + 7484.0$
$y' = -5.857(-248.4) + 7484.0$
$y' = 8938.9$
$d' = |(8938.9 - 8939.0)| \sin 10° = 0.0$

Test $Rg_2 = (1346.0, 9256.0)$ in equation:
$y = -5.857(1346.0 - 1652.0) + 7484.0$
$y = -5.857(-306.0) + 7484.0$
$y = 9276.0$
$d = |(9276.0 - 9256.0)| \sin 10° = 3.5$
New Point $Rg_2' = (1349.4, 9256.6)$
Recheck:
$y' = -5.857(1349.4 - 1652.0) + 7484.0$

$y' = -5.857(-302.6) + 7484.0$
$y' = 9256.3$
$d' = |(9256.3 - 9256.0)| \sin 10° = 0.1$

Segment 6:

$C_0 = (1551.0, 8606.0)$
$C_1 = (1664.0, 10281.0)$

Centerline Equation: $y = 14.8230(x - 1551.0) + 8606.0$

Test $Rg_1 = (1416.0, 6607.0)$ in equation:
$y = 14.8230(1416.0 - 1551.0) + 8606.0$
$y = 14.8230(-135.0) + 8606.0$
$y = 6604.9$
$d = |(6604.9 - 6607.0)| \sin 4° = 0.1$

Test $Rg_2 = (1386.0, 6163.0)$ in equation:
$y = 14.8230(1386.0 - 1551.0) + 8606.0$
$y = 14.8230(-165.0) + 8606.0$
$y = 6160.2$
$d = |(6160.2 - 6163.0)| \sin 4° = 0.2$

Segment 8:

$C_0 = (1620.0, 10986.0)$
$C_1 = (1314.0, 12937.0)$

Centerline Equation: $y = -6.3758(x - 1620.0) + 10986.0$

Test $Rg_1 = (1209.0, 13607.0)$ in equation:
$y = -6.3758(1209.0 - 1620.0) + 10986.0$
$y = -6.3758(-411.0) + 10986.0$
$y = 13606.4$
$d = |(13606.4 - 13607.0)| \sin 9° = 0.1$

Test $Rg_2 = (1002.0, 14934.0)$ in equation:
$y = -6.3758(1002.0 - 1620.0) + 10986.0$
$y = -6.3758(-618.0) + 10986.0$
$y = 14926.2$
$d = |(14926.2 - 14934.0)| \sin 9° = 1.2$

Segment 10:

C_0 = (1108.0, 13538.0)
C_1 = (631.0, 14450.0)

Centerline Equation: $y = -1.9119(x - 1108.0) + 13538.0$

Test Rg_1 = (1451.0, 12882.0) in equation:
$y = -1.9119(1451.0 - 1108.0) + 13538.0$
$y = -1.9119(343.0) + 13538.0$
$y = 12882.2$
$d = |(12882.2 - 12882.0)| \sin 28° = 0.1$

Test Rg_2 = (1793.0, 12226.0) in equation:
$y = -1.9119(1793.0 - 1108.0) + 13538.0$
$y = -1.9119(-4286.0) + 13538.0$
$y = 12228.3$
$d = |(12228.3 - 12226.0)| \sin 28° = 1.1$

H.4 Turning Aids

Table H.4: Results of geometric verification of turn bearings.

Navaid	TB	Point O	Point C	Point Q	d
Beacon N	323	(5344.9, 5049.5)	(5261.7, 5440.8)	(5099.1, 5075.3)	57.2
Light A	16	(2836.3, 6020.3)	(2933.1, 6408.4)	(2715.2, 6072.9)	-1.8
C Rear	255	(2231.3, 6389.5)	(2387.6, 6757.7)	(2028.1, 6582.3)	12.6
C Front	225	(1810.2, 7001.2)	(2133.8, 7936.3)	(1739.9, 7166.8)	39.9
Light 2	223	(1561.8, 8012.2)	(1937.7, 8149.0)	(1538.7, 8176.9)	21.7
Light C	266	(1662.1, 10252.5)	(1274.0, 10349.3)	(1669.1, 10411.9)	-44.3
E Front	130	(1303.1, 13006.3)	(903.2, 13013.3)	(1256.4, 13201.1)	-34.1

Table H.4 above summarizes the test results of geometric modeling of the turns for verification of the locations of turning aids with respect to the turn bearings given in column 5 of the course card. For each turn, the table lists the name of the turning aid, the course card turn bearing TB, the startpoint of the turn O, the calculated center of turn C, the calculated endpoint of the turn Q, and the resulting deviation d of Point Q from the new segment's centerline. Positive d indicates the sub ended up on the right of the

centerline while negative indicates leftward error. Deviations more than 20 yds. (or 10 yds. for Segments 3 and 4) should be fixed by computing a new turn bearing that will start the turn sufficiently earlier or later to place the sub near the centerline at turn completion. See Appendix J for these calculations, based on the deviations obtained from integration tests using an autopilot to turn. Though less accurate, these test results also indicate that all turn bearings save the second need to be changed in the course card to improve the turns. The values for the deviations are fairly close to those of Appendix J.

Appendix I

Perceptual Cue Tuning Verification

This appendix gives results of tests on the perceptual cue enhancements of objects in the VE. These enhancements include the scaling algorithm for distant buoys, enlargement and multiple-face display of numbers on the buoys, and upscaling of the range markers and turning beacons.

I.1 Buoys

The two major perceptual cue enhancements in the visual representations of channel buoys are the piecewise linear scaling algorithm dependent on buoy distance and the enlargement of the white buoy numbers to cover the entire upper face of the buoy. This section gives results of tests aimed at verifying that this visual cue tuning enables sufficient visibility of distant buoys and legibility of buoy numbers to match the real task.

I.1.1 Visibility of Buoys

Section 3.4.7 described the piecewise linear scaling algorithm used to make distant buoys more visible. The scale is maintained at a factor of one for distances less than 1000 yds. ($ODIST$), a value slightly lower than the distance at which unscaled buoys' color becomes indistinguishable in the HMD. Then the scale linearly increases up to a factor of two when the distance reaches 2000 yds. ($ADIST$), the typical maximum distance for recognizing a

buoy and its color in the real task. Section 4.3.6.1 discussed verification techniques using human subjects to test whether this scaling algorithm renders enables buoy visibility in the VE from distances at least as great as in the real task. Five subjects were tested. Use of binoculars was not allowed. The first test featured a buoy receding into the distance, with the same surrounding scenery as the normal VE except that no land was included. The view direction was fixed, with no head tracking. Subjects marked the time they could no longer distinguish the buoy from the surrounding background (eg when its size became less than a pixel in the HMD). The distance of the buoy from the viewpoint at this moment was recorded. Both green and red buoys were tested in this way. Since the overall minimum $ODIST$ distance is slightly greater than 1000 yds., the use of 1000 yds. as the $ODIST$ value in the scaling algorithm is validated. The second test featured approaching buoys; the moment at which a subject could first identify an approaching buoy's color was recorded. Table I.1 gives the distance of the buoy at this moment, $ODIST'$, both for green and red buoys. Since the minimum of each subject's results over the two colors exceeds the goal of 2000 yds., the effectiveness of the algorithm has been verified.

Table I.1 gives the results of the tests on the five subjects to determine the observed green, red, and minimum $ODIST$ values and the green, red, and minimum $ODIST'$ values. The last row gives the averages for the green and red columns, and the overall minimum of the minimum columns.

Table I.1: Test results for buoy visibility.

	$ODIST$			$ODIST'$		
Subject	Red	Green	Min	Red	Green	Min
1	1798.0	1522.0	1522.0	4755.0	4767.0	4755.0
2	1316.0	1116.0	1116.0	4797.0	4269.0	4269.0
3	1768.0	1654.0	1654.0	5532.0	4167.0	4167.0
4	1504.0	1426.0	1426.0	3183.0	2022.0	2022.0
5	1434.0	1196.0	1196.0	2940.0	2541.0	2541.0
Avg/Min	1564.0	1382.0	1116.0	4241.4	3553.2	2022.0

I.1.2 Legibility of Buoy Numbers

A similar test to the above determination of $ODIST'$ was performed to determine the distance at which subjects could first correctly read a buoy's number. Binoculars were turned on for this test, since reading the number without them is only possible at very close range. A red buoy numbered "34" and later a green buoy numbered "51" were moved slowly toward the viewpoint, with the same surroundings as above (normal scene, no land). Subjects could make several attempts at reading the number, but only the first correct guess was recorded. Table I.2 lists the red, green, and minimum distances of first correct reading for four different subjects. The last row again gives the averages of the green and red columns and the overall minimum of the minimum column. Since this minimum is greater than 540 yds., this test verifies that subjects should be able to read the nearest buoy from any point in the channel. Other range marker pairs in the VE may yield quite different results, however, since size and distance of range markers varies considerably.

Table I.2: Test results for buoy number legibility.

	Distance of first correct reading		
Subject	Red	Green	Minimum
1	708.0	848.0	708.0
2	730.0	550.0	550.0
3	674.0	979.0	674.0
4	615.0	813.0	615.0
Avg/Min	681.8	797.5	550.0

I.2 Range Markers

Section 4.3.6.2 discussed visual tests on the range marker scaling to determine the first distance at which range marker separation could be distinguished without binoculars for a 20-yard centerline deviation. The same test also recorded when the subject first could distinguish the orange tip of one of the range markers without binoculars. The range markers used in the test were those in Segment 0, at heights of 15 and 30 yds., and the viewpoint was initially placed 4000 yds. away from the shorter marker. Table I.3 gives these distances as measured for five subjects. The average result of about 2000 yds., or

one nautical mile fails to meet the approximate 3-mile estimate of Lt. Andrew for visibility of dayboards and separation, indicating that the VE still necessitates greater reliability on binoculars than the real task for range markers.

Table I.3: Test results for distinguishing range marker separation and dayboard color.

Subject	Distance saw separation	Distance saw dayboard color
1	2678.0	1996.0
2	1506.0	1506.0
3	2776.0	1986.0
4	2130.0	1860.0
5	2032.0	1840.0
Avg	2224.4	1837.6

Appendix J

Integration Tests

As with the data validation tests in Appendix H.4, the purpose of the tests in this appendix are to verify that the turn bearings given in column 5 of the course card for inbound navigation are consistent with the locations of turning aids. This time experimental data was obtained from the integrated simulation, using extra code for an "autopilot" in the dynamics process. For each turn, the trial began with the submarine on the beginning of the initial segment's centerline (or near the end for longer segments), moving forward at 12 knots along the segment's heading. At the moment the compass bearing to that segment's turning aid reached the prescribed course card value, an automatic turn via a "Steer <HDG>" command was given, where <HDG> is the heading of the next segment. Each test ended when the sub crossed the opposite side of that segment; at this time, mean centerline deviation for both segments were written to a datafile by the experimenter's interface. Mean deviation was used here, rather than RMS deviation, because the error after a turn would always lie on the same side of the centerline and would maintain approximately the same value throughout the travel to the end of the segment, since the course is parallel to the centerline from then on. An allowable mean centerline deviation of 20 yds. was used for turns into segments with centerlines, while a threshold of 10 yds. was used for those without centerline (Segments 3 and 4). All turns except the second turn from Segment 2 to Segment 3 were found to exceed these thresholds. For these cases, an improved point of turn was computed (see Section 4.4.1) and the compass bearing to the turning aid from this point was suggested as an updated turn bearing entry for the course card. These updated turn

bearings were then rechecked using the same methods; all turns easily passed the criteria by exhibiting less than 10 yds. of deviation from the target centerlines. As a final check on the accuracy of the new turn bearings for human pilots, the author used the experimenter's interface to record his performance in the simulator going through the entire channel, from start to finish. No attempts at aligning the range markers were made in between turns, to see whether small errors from each turn would accumulate into unacceptable error levels by the last segment. This in fact was not the case; as with the segment-by-segment autopilot tests, each turn produced well under 10 yds. of deviation. Thus, the new turn bearings have been sufficiently verified and should be installed as soon as possible into the course card view before the next series of pilot experiments begin.

Table J.1 lists the important data inputs and results for each of the seven tested turns. The first through fourth columns give the inputs, namely the starting segment number, the starting point, the location of the turning aid for that segment, and the turn bearing from the fifth column of the course card which was used by the autopilot to time the start of the turn. The remaining columns are outputs of the test, namely mean centerline deviation for the first segment (except for segment 1 which started at the end of the centerline), mean centerline deviation for the second segment, and whether or not the deviation passed or failed the allowable threshold. Positive deviation indicated rightward error while negative indicates leftward error.

Table J.1: Results of integrated tests using an autopilot for turns.

Start Seg	Start Point	Navaid Loc	TB	Dev1	Dev2	Pass/Fail
0	(5676.0, 5062.0)	(4841.0, 5718.0)	323	–	53.55	Fail
2	(4118.0, 5444.0)	(2934.0, 6361.0)	16	-2.89	-4.56	Pass
3	(2530.0, 6193.0)	(1386.0, 6163.0)	255	-1.37	11.00	Fail
4	(1940.0, 6734.0)	(1416.0, 6607.0)	225	-0.34	30.00	Fail
5	(1652.0, 7484.0)	(1016.0, 7427.0)	223	-1.95	21.14	Fail
6	(1551.0, 8606.0)	(754.0, 10189.0)	266	2.24	-42.83	Fail
8	(1620.0, 10986.0)	(1451.0, 12882.0)	130	-2.80	-31.32	Fail

For those turns that failed the threshold, Table J.2 summarizes the computations of new turn bearings. The first column gives the number of the segment from which the turn begins (matching Table J.1). The second column gives the point O on this segment at which the

turn began, which is the intersection of the centerline with a line through the turning aid at an angle whose compass heading matches the prescribed turn bearing. The third column gives the mean centerline deviation d for that segment from Table J.1. The fourth column lists ΔH, the change in heading, equal to $H_{cmd} - H_0$ in degrees. The fifth column gives the computed distance that the turn point should be moved along the centerline, positive indicating forward and negative indicating backward. This is obtained by dividing the mean deviation from the last column by $\sin \Delta H$. The sixth column, gives the coordinates of this new translated turning point, O'. From this, the updated turn bearing TB' to the turning aid was directly derived and is given in column seven, rounded to the nearest whole number.

Table J.2: Calculations of improved turn bearings.

Start Seg	Point O	d	ΔH	$d/\sin \Delta H$	Point O'	TB'
0	(5344.9, 5049.5)	53.57	26	122.2	(5222.8, 5048.3)	330
3	(2231.3, 6389.5)	11.00	9	21.4	(2213.4, 6401.1)	254
4	(1810.2, 7001.2)	30.02	16	108.9	(1762.5, 7099.1)	215
5	(1561.8, 8012.2)	21.18	14	87.5	(1546.6, 8098.4)	218
6	(1662.1, 10252.5)	-42.83	-13	188.4	(1675.2, 10440.4)	255
8	(1303.1, 13006.3)	-31.38	-19	96.4	(1288.0, 13102.5)	144

Table J.3 shows the results of a second series of autopilot tests using the new updated values for course card turn bearings. The format matches that of Table J.1. This time, all turns successfully passed the allowable deviation threshold.

Table J.3: Autopiloted test results for the updated turn bearings.

Start Seg	Start Point	Navaid Loc	TB	Dev1	Dev2	Pass/Fail
0	(5676.0, 4118.0)	(4841.0, 5718.0)	330	–	3.84	Pass
2	(4118.0, 5444.0)	(2934.0, 6361.0)	16	-2.89	-4.56	Pass
3	(2530.0, 6193.0)	(1386.0, 6163.0)	254	-1.37	0.00	Pass
4	(1940.0, 6734.0)	(1416.0, 6607.0)	215	-0.30	0.02	Pass
5	(1652.0, 7484.0)	(1016.0, 7427.0)	218	-1.94	-1.08	Pass
6	(1551.0, 8606.0)	(754.0, 10189.0)	255	2.24	-1.67	Pass
8	(1620.0, 10986.0)	(1451.0, 12882.0)	144	-2.80	3.47	Pass

Table J.4 gives summary performance data of a human-piloted navigation of the entire channel in the integrated simulation, using the updated turn bearings. Paralleling the

autopilot tests, the author issued a "Steer" command for each turn at the moment the compass bearing on the turning aid reached the updated turn bearing value. Here, however, no pause or correction occurred in between segments. Table J.4 simply lists the start point and the mean centerline deviations for all segments with centerlines after Segment 0. As with Table J.3, all turns successfully passed 10-yard thresholds.

Table J.4: Results of human-piloted test of the updated turn bearings over the entire course.

Start Point	Seg 2	Seg 3	Seg 4	Seg 5	Seg 6	Seg 8	Seg 10
(5676.0, 5062.0)	6.37	4.76	-0.32	1.30	-0.53	-5.17	2.60

Appendix K

Domain Expert Questionnaire

The following survey was administered to Lt. Allen Andrew, from the Naval Officers' School in Groton, CT on February 21, 1995. This was Lt. Andrew's second trip to MIT to evaluate the simulator. He had had considerable experience with the real OOD task and had provided helpful verbal feedback on an early prototype of the simulator during the first visit. The majority of the survey asked for numerical ratings on the utility and accuracy of each feature in the VE. The last portion of the survey included two sets of open-ended questions, one about the logical interface and one aimed at obtaining information relevant to the task elements for future iterations of the simulator. The survey is shown on the following pages with Lt. Andrew's answers in bold and italics.

One of the critiques listed below, namely lack of backup advice from the Chief Navigator, is not relevant to Version 1.0, which does not model that participant of the task. All other specific complaints and items with scores below three 3 were either validated as acceptable implementation tradeoffs, or were fixed or improved by the completion of Version 1.0. For instance, a uniform scaling of all range markers greatly improved their size and visibility. The utility and readability of the chart view was improved by adding symbols for turning beacons and range markers and enlarging the text overlays. Adding a felt strip underneath the adjustable bar over the top of the helmet rendered it much more comfortable. Finally, the rudder was thickened for better visibility when amidships and the entire submarine was further submerged to place the propeller under sea level. Overall, Lt. Andrew reported that the simulation had greatly improved since his first visit three months earlier.

OOD Questionnaire

To be administered to OOD from Groton on Tuesday, 2/21/95, as an evaluation of the baseline simulation.

In order to help us verify that our simulation is as accurate as possible, please answer the following questions about the features of the Virtual Harbor Environment and the ease of use of the physical interface, based on your experiences navigating the real King's Bay and other harbors. Use the following six point scale:

5 - Extremely Satisfactory
4 - Moderately Satisfactory
3 - Borderline
2 - Moderately Unsatisfactory
1 - Extremely Unsatisfactory
D - Don't Know / Not Applicable

For answers less than "moderately satisfactory," please suggest how the feature might be improved in the space following the question.

Channel

1. How closely did the shape of the channel match the real King's Bay channel?
   D (5)
2. How accurate was the channel width?
   4

Buoys

1. How accurate were the sizes of buoys?
   5
2. How accurate were the locations of buoys?
   5
3. How sufficient was the visibility for sighting distant buoys?
   5
4. How well could you identify buoy numbers, compared to the real world task?
   4

Range Markers

1. How well did the shape of the range marker supports match those in King's Bay?
   D (5)
2. How accurate was the color of the range marker supports?
   D
3. How accurate was the color of the range marker dayboards?
   4

4. How accurate were the sizes of the range markers?
   2
5. How accurate were the locations of range markers?
   4
6. How accurate was the alignment of the range marker pairs with the centerlines?
   5
7. How sufficient was the visibility for sighting distant range markers?
   2

Turning beacons

1. How accurate was the representation of the daybeacons?
   4
2. How accurate were the locations of all navaids used as turning beacons?
   4

Land and Water appearance

1. Rate the recognizability and accurate placement of landmarks (e.g. Fort Clinch, Drum Island)?
   4
2. How accurate was the shoreline geometry?
   4
3. How realistic was the water appearance?
   4
4. How realistic was the sense of vehicular motion?
   4

Binoculars

1. How accurate was the magnification factor?
   5
2. How closely did the field of view match that of real binoculars?
   5
3. How easily were you able to distinguish binocular view from normal view?
   5
4. How well were you able to see sufficient detail with the binoculars?
   5

Compass

1. How legible was the compass?
   5
2. How easy to use was the compass?
   5
3. How well did your use of the virtual compass correspond to its use in a real task situation?
   4

Course Card

1. How sufficient was the information in the course card for successful task performance?
   5
2. How well did the format of the information match that of a real course card?
   3 – Course Card will not have O/S Cse, Speed, or last order
3. How legible was the course card?
   3 – O/S Cse Speed & Last order hard to read.
4. How easy to use was the virtual course card?
   4

Virtual Charts

1. How useful was the virtual chart?
   3
2. How accurate was the representation of the chart?
   5
3. How easy was identifying a particular buoy or channel segment on the chart using the various chart viewing commands?
   4
4. How easy was determining the location of the submarine using the various chart viewing commands?
   4 – Chart should have turn markers.

Head Mounted Display

1. The field of view of the head mounted display (HMD) is restricted both vertically and horizontally. How would you rate using this HMD for the OOD task?
   4
2. How accurate were the colors of objects you saw in the HMD?
   5
3. How accurate was the brightness of the image for a midday harbor scenario?
   5
4. How well did the view correspond to where you felt you were looking?
   5
5. Was the HMD comfortable to wear throughout the duration of the session?
   3 – Started hurting top and back of head after about 45 minutes

Speech recognition

1. How accurate was the syntax and phrasing of the simulation's allowed commands?
   4
2. Out of the commands allowed in this simulation, how well did the system understand your commands?
   3 – ("2" gave a repeat back of "7")
3. How accurate was the delay in the system's verbal response?
   5
4. How accurate was the phrasing of system's response?
   4
5. How accurate was the response for unrecognized commands (e.g. "Say again, sir.")? *4*

Submarine

1. How realistic was the appearance of the submarine?
   4 – Rudder looks small when amidships.
2. How accurate was the sub's level of submergence?
   4 – Propeller should not show above water.
3. How accurate was the placement of viewpoint on the conning tower?
   5
4. How easily were you able to look back at the rudder, and other parts of sub?
   5
5. How accurate was the delay in rudder movement in response to steering commands?
   4 – Rudder seemed to move too much for small course changes.
6. How well did the apparent motion of the sub match the displayed speed?
   5
7. How accurate was the acceleration rate of the sub?
   5
8. How accurate were the turn rates?
   5

Please answer briefly the following questions about your experience with the simulation:

1. Were you able to perform the task with the given information in the simulation? ***Yes***

If so, were you able to perform the task as well as you could in a real world situation? If not, why?

No – no backup from Navigator.

2. Did the information in the Course Card accurately reflect what you saw in the "subview" mode?

Yes

3. When switching back to the submarine view after looking at the course card, did you feel any unusual disorientation?

No

4. Did you have any difficulty in resighting an object after switching back from the course card?

No

5. Were there instances where an unrecognized voice command noticeably hampered your ability to perform the task? If so, please explain briefly.

Yes – "2" vs. "7"

Please answer briefly the following questions based on your experiences as an OOD to aid us with future development.

1. In clear weather, what is the maximum distance an OOD typically recognizes a buoy and its color without binoculars?

Size depends, but normally can see 2-3 sets

With binoculars?

Size depends, but normally can see 4-5 sets

2. What is the maximum distance an OOD would recognize a range marker without binoculars? ***about 3 mi.***

With binoculars?

about 6–8 mi.

3. What is the maximum distance an OOD would recognize a daybeacon without binoculars? ***about 1.5 mi.***

With binoculars?

about 3–4 mi.

4. What is the maximum distance at which an OOD can typically read the number on a buoy without binoculars? ***.5 mi.***

With binoculars?

1 mi.

6. Does a submarine typically bank left/right or up/down during turns?

Not at slow speeds.

7. Is there a recommended speed for making turns when the navaid listed in the course card is at the given turn bearing value? If so, what is it?

The transit speed.

8. Will a faster speed produce a larger turn radius (therefore requiring an earlier turn)?

Yes

Appendix L

Pilot Experiment Results

The tables on the next two pages list the summarized scores for six of the seven subjects used in the pilot experiments for Version 1.0 (one subject was started too far into Segment 2 due to a problem with the simulation pause mode). All ten trials are included for each subject. For segment-by-segment scores, see files with extension .apr on beamish.mit.edu in vett/ood/EI/pilotdata. The performance categories in the tables are as follows:

A Percentage of time spent more than 20 yards from the centerline (not including turn areas or segments without centerlines)

B Percentage of time spent out of the channel

C Mean deviation from the centerline (yards). Negative values indicate left of centerline.

D Standard deviation of centerline deviation (yards)

E Root Mean Square of centerline deviation (yards)

F Total time of trial (seconds)

All subjects navigated the portion of the inbound channel from Segment 2 to the end of Segment 6. Subjects 1 and 2 ran using the channel shown in Appendix B, except with no centerlines in Segments 3 and 4. Subjects 3-6 ran with centerlines in all five segments, but with Segment 6 shortened to about half its normal length. Graphs of trial-by-trial scores in Category A are given at the end of this appendix.

Subject 1

Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	37.8	2.7	10.6	19.9	22.5	2312
2	28.2	0.0	-3.2	18.2	18.4	2535
3	29.8	0.0	11.5	17.8	29.8	1267
4	12.6	0.0	-2.9	10.9	11.3	1202
5	19.3	2.1	-12.2	29.9	32.3	1188
6	26.1	0.0	12.9	10.3	16.5	1272
7	12.5	0.0	2.6	13.7	13.9	1470
8	7.8	0.0	2.9	11.8	12.1	1222
9	8.9	0.0	4.3	10.9	11.7	1208
10	0.3	0.0	3.1	6.0	6.8	1211

Subject 2

Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	2.8	0.0	10.0	7.3	12.4	1441
2	25.2	0.0	-8.8	15.0	17.4	1130
3	13.2	6.1	7.5	17.4	19.0	1766
4	13.5	3.5	4.1	20.4	20.8	1685
5	1.0	0.0	-2.0	6.6	6.9	1077
6	8.1	0.0	9.7	10.3	14.1	1208
7	1.3	0.0	0.3	9.2	9.2	1315
8	0.0	0.0	3.3	4.1	5.3	1107
9	0.0	0.0	6.6	2.9	7.2	1254
10	1.5	0.0	1.5	6.5	6.7	1272

Subject 3

Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	44.3	1.5	-10.0	33.6	35.1	1546
2	2.0	0.0	-0.5	8.8	8.8	1075
3	0.0	0.0	3.5	4.6	5.8	909
4	4.5	0.0	-2.0	8.0	8.2	817
5	1.6	0.0	-3.4	6.7	7.5	967
6	2.0	0.0	7.3	5.6	9.1	906
7	1.0	0.0	-2.2	5.2	5.7	903
8	0.0	0.0	3.1	3.8	4.9	898
9	2.7	0.0	2.9	5.8	6.5	911
10	0.0	0.0	-2.6	3.1	4.0	911

Subject 4

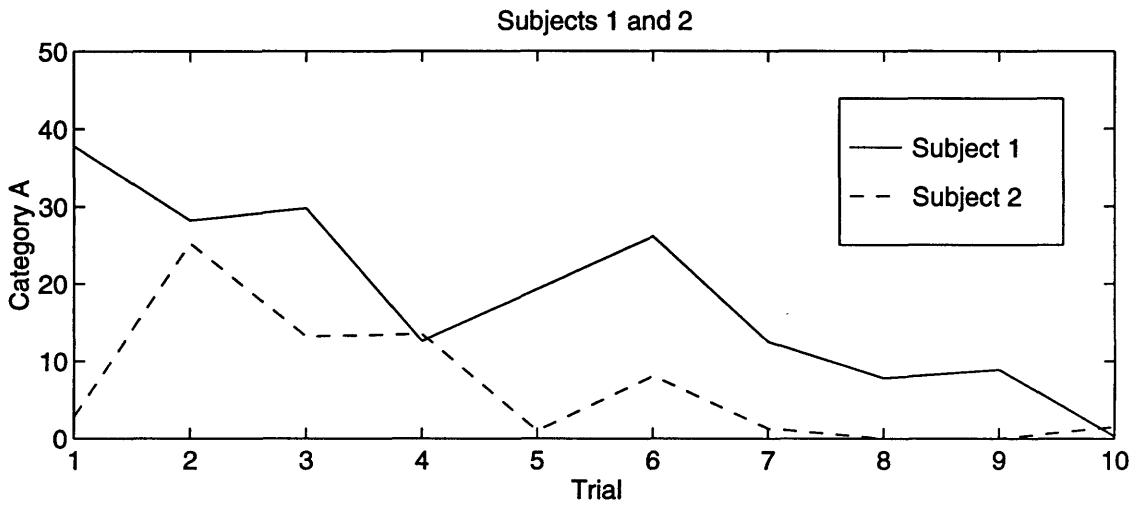
Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	25.8	0.0	10.7	15.7	19.0	1672
2	0.0	0.0	-5.0	8.4	9.8	1474
3	0.0	0.0	6.7	3.4	7.6	975
4	0.0	0.0	-2.2	8.8	9.1	895
5	0.0	0.0	-5.1	6.3	8.2	911
6	1.4	0.0	-0.7	5.7	5.8	892
7	9.9	0.0	0.8	10.9	10.9	903
8	0.0	0.0	-0.2	4.7	4.7	991
9	11.0	0.0	-2.2	10.0	10.2	916
10	9.7	0.0	-4.5	8.1	9.3	1009

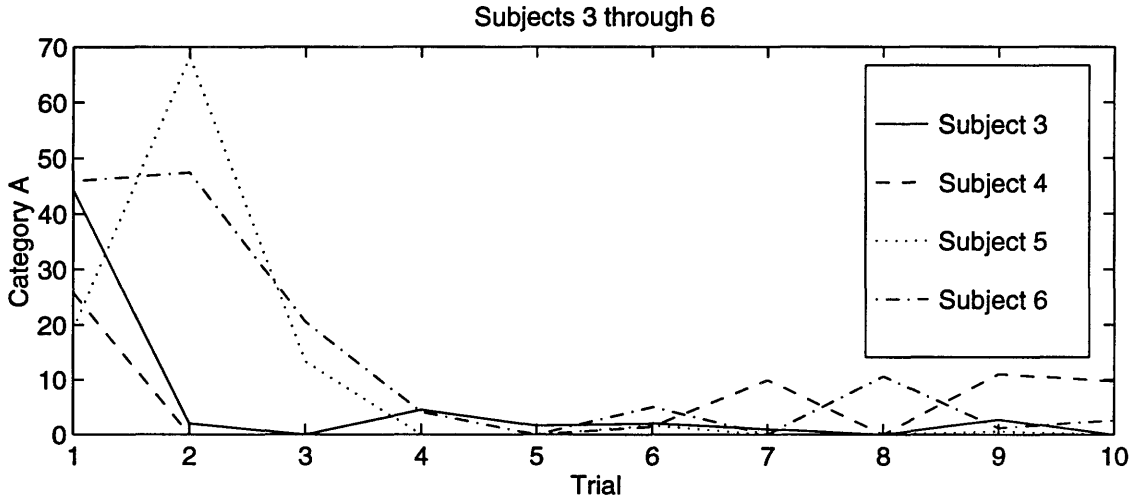
Subject 5

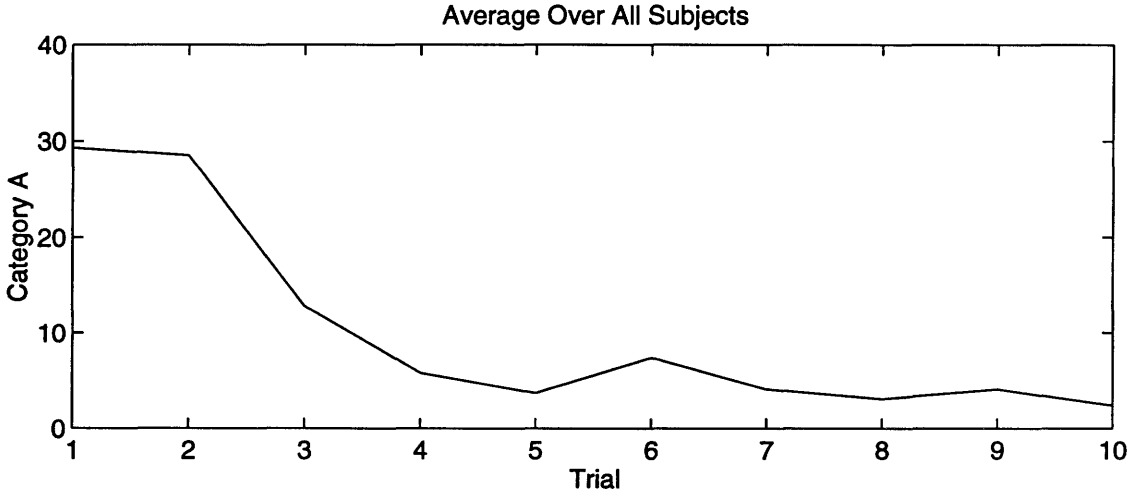
Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	19.2	14.2	-16.4	40.8	44.0	1794
2	68.2	20.1	19.8	38.3	43.1	1417
3	13.3	0.0	8.3	7.1	10.9	1354
4	0.0	0.0	3.6	4.2	5.5	1325
5	0.0	0.0	-3.4	6.3	7.2	1298
6	1.6	0.0	4.2	5.5	6.9	903
7	0.0	0.0	-3.9	3.7	5.3	904
8	0.0	0.0	0.7	3.4	3.5	898
9	0.6	0.0	0.0	5.7	5.7	904
10	0.0	0.0	-1.5	3.4	3.7	916

Subject 6

Trial	Category A	Category B	Category C	Category D	Category E	Category F
1	45.9	2.8	-18.8	33.2	38.1	1284
2	47.4	8.4	6.2	25.2	25.9	1486
3	20.5	0.0	1.4	17.2	17.2	1769
4	4.1	0.0	-4.9	11.0	12.0	1244
5	0.0	0.0	-3.5	6.6	7.5	1487
6	5.0	0.0	8.1	9.3	12.4	1345
7	0.0	0.0	-3.1	10.0	10.4	1271
8	10.6	0.0	6.8	6.1	9.1	1403
9	1.2	0.0	5.3	5.6	7.7	1294
10	2.6	0.0	-4.3	10.0	10.9	1257


Subjects 1 and 2
Category A
Trial
Subject 1
Subject 2


Subjects 3 through 6
Category A
Trial
Subject 3
Subject 4
Subject 5
Subject 6


Average Over All Subjects
Category A
Trial

References

[Asc94] Ascension Technology Corporation, Burlington, VT. *The Flock of Birds Position and Orientation Measurement System Installation and Operation Guide*, October 1994.

[Ban95] James Bandy. Measurement of end-to-end system delays in the vett core testbed. Summary Report of Research Assistant activities, May 1995.

[BD87] B. Bigelow and D. Day. Digital typography. In Ronald M. Baecker and William S. Buxton, editors, *Readings in HCI*, chapter 3, pages 1–40. Morgan Kaufman Publishers, Inc., Los Altos, CA, 1987.

[Boe84] Barry W. Boehm. Verifying and validating software requirements and design specifications. *IEEE Software*, 1(1):75–88, January 1984.

[Car90] Stuart K. Card. Modeling scenarios for action. In Julian Hochberg Jerome I. Elkind, Stuart K. Card and Beverly M. Huey, editors, *Human Performance Models for Computer-Aided Engineering*, pages 233–237. Academic Press, San Diego, CA, 1990.

[CZ92] David Chen and David Zeltzer. *The 3d Virtual Environment Dynamic Simulation Language.* Computer Graphics and Animation Group, MIT Media Lab, Cambridge, MA, August 1992.

[Dav93] Alan M. Davis. *Software Requirements: Objects, Functions and States.* PTR Prentice Hall, Englewood Cliffs, NJ, 1993.

[DJ92] William K. Durfee and Richard M. Willis Jr. Simple depth cues for representing 3-D workspaces on 2-D computer displays. *IEEE Journal of Robotics and Automation*, 1992.

[DPC+87] Colin G. Drury, Barbara Paramore, Harold P. Van Cott, Susan M. Grey, and E. Nigel Corbett. Task analysis. In G. Salvendy, editor, *Handbook of Human Factors*, pages 370–401. Wiley, New York, 1987.

[ERJD85] Robert T. Hennessy Edward R. Jones and Stanley Deutsch. *Human Factors Aspects of Simulation.* National Academy Press, Washington, D.C., 1985.

[GKL94] Brad Goodman, Alfred Kobsa, and Diane Litman, editors. *Proc. Fourth International Conference on User Modeling*, Bedford, MA, August 1994. The MITRE Corporation.

[Gom90] Hassan Gomaa. The impact of prototyping on software system engineering. In Richard Thayer and Merlin Dorfman, editors, *System and Software Requirements Engineering*, pages 543–552. IEEE Computer Society Press, Washington, D.C., 1990.

[Gua94] U.S. Coast Guard. Light list: Atlantic and gulf coasts, 1994.

[Har87] D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987.

[Har92] Kelly Harwood. Defining human-centered system issues for verifying and validating air traffic control systems. In John A. Wise, V. David Hopkin, and Paul Stager, editors, *Verification and Validation of Complex Systems: Human Factors Issues*, pages 115–129. Springer-Verlag, Berlin Heidelberg, Germany, 1992.

[Hue68] Edmund B. Huey. *The Psychology and Pedagogy of Reading.* MIT Press, Cambridge, MA, 1968.

[Ins93] Institute for Simulation and Training, Orlando, FL. *Standard for Information Technology – Protocols for Distributed Interactive Simulation Applications*, version 2.0 third draft edition, May 1993.

[Joh87] Richard S. Johnston. The simnet visual system. In *Proc. Ninth ITEC Conference*, pages 264–273, Washington, D.C., December 1987.

[KA93] Peter L. Knepell and Deborah C. Arangno. *Simulation Validation.* IEEE Computer Society Press, Los Alamitos, CA, 1993.

[LS+93] Margaret Loper, Steve Seidensteiker, et al. The dis vision: A map to the future of distributed simulation. Technical report, Institute for Simulation and Training, October 1993. Comment Draft prepared by DIS Steering Committee.

[Mag93] Lochlan Magee. Mars virtual reality simulator. Fact Sheet G-07, Defence and Civil Institute of Environmental Medicine, North York, Ontario, Canada M3M 3B9, November 1993.

[Mal77] E. S. Maloney. *Chapman Piloting: Seamanship and Small Boat Handling.* American Book-Stratford Press, Inc., New York, 1977.

[Nag91] Shojiro Nagata. How to reinforce perception of depth in single two-dimensional pictures. In Stephen R. Ellis, editor, *Pictorial Communication in Virtual and Real Environments*, pages 527–545. Taylor & Francis Ltd., London, 1991.

[Nyg94] Erik Nygren. Vettnet design and use. Manual for using VETTnet blackboard software, May 1994.

[Oma91] Charles M. Oman. Sensory conflict in motion sickness: An observer theory approach. In Stephen R. Ellis, editor, *Pictorial Communication in Virtual and Real Environments*, pages 362–376. Taylor & Francis Ltd., London, 1991.

[Ous94] John K. Ousterhout. *TCL and the TK Toolkit.* Addison-Wesley, Reading, MA, 1994.

[Pfa94] Jonathan Pfautz. Fitts' law in a 3-dimensional simulated environment. Cognitive Science research paper, May 1994.

[PL94a] Richard W. Pew and William H. Levison. Presentation to virtual environment training technical advisory group by bbn inc. Transcript of slide presentation, October 1994.

[PL94b] Richard W. Pew and William H. Levison. Visit of new london oods. Transcript of slide presentation, November 1994.

[Pre93] Jenny Preece. *A Guide to Usability: Human Factors in Computing.* Addison-Wesley, 1993.

[Rei94] Brett S. Reid. Creating a multimodal virtual environment and an interface to it. Bachelor's thesis, Massachusetts Institute of Technology, Computer Science and Engineering, December 1994.

[RS86] J. M. Rolfe and K. J. Staples. *Flight Simulation.* Cambridge University Press, 1986.

[She92] Thomas B. Sheridan. *Telerobotics, Automation, and Human Supervisory Control.* MIT Press, Cambridge, MA, 1992.

[SZ93] Thomas B. Sheridan and David Zeltzer. Virtual reality. *Technology Review,* 10(4), October 1993.

[Vir93] Virtual Research Systems, Inc., Santa Clara, CA. *Eyegen3 User's Guide,* August 1993.

[Vir94] Virtual Research Systems, Inc., Santa Clara, CA. *VR4 User's Guide,* December 1994.

[War82] John Ware. Comments on simplified simulation equations for attack class submarines. Reference used by BBN for encoding submarine dynamics, January 1982.

[Wel89] Pierre D. Wellner. Statemaster: A uims based on statecharts for prototyping and target implementation. In Ken Bice and Clayton Lewis, editors, *CHI '89 Proceedings,* pages 177–182, Austin, TX, May 1989.

[WHLG94] R. W. Pew W. H. Levison and D. J. Getty. Application of virtual environments to the training of naval personnel. BBN Report 7988, BBN Systems and Technologies, 10 Moulton St., Cambridge, MA 02138, June 1994.

[ZDA+94] David Zeltzer, Nathaniel I. Durlach, Walter A. Aviles, Rakesh Gupta, Nicholas Pioch, and Jeng-Feng Lee. Officer of the deck: Implementation and evaluation of a virtual environment training system. Submitted to 1995 Symposium on 3D Interactive Graphics, October 1994.

[Zel92] David Zeltzer. Autonomy, interaction and presence. *Presence*, 1(1), 1992.