Virtual Environments for Training
Annual Productivity Report
October 1 1996 to September 30, 1997

Submitted by R. Stiles
Lockheed Martin Advanced Technology Center

Prepared for
Office of Naval Research

Contract N00014-95-C-0179
(CDRL Report A001)

August 1997

Virtual Environments for Training
Overview of Scientific Progress
Lockheed Martin Contract N00014-95-C-0179

The Training Studio, which allows the application of virtual environments to training, integrates three significant components: Vista Viewer, the virtual environment display system; VIVIDS, the object simulation system; and Steve, the pedagogical agent system. For the Training Studio, the second year of research and development on the Virtual Environments for Training (VET) contract has resulted in improvements and advancements in four technical areas that apply to training: authoring VE simulations, virtual environment interaction, pedagogical agents, and team training. To assess some of the new development, evaluation efforts began during the latter part of the contract year. A study focusing on the effects of 3D vs 2D learning on retention was designed and initiated by the Human Resources Cognitive Training group (HRCT) of U.S. Air Force's Armstrong Laboratory at Brooks Air Force Base in Texas.

The authoring of VE simulations for training has been improved on several fronts affecting authoring 3D models, training simulations, and instruction. The quality of virtual environment interaction is very important for training in equipment operations and maintenance. By adapting VRML 2.0 for immersive use, we have realized a unifying framework for interaction that allows storing knowledge of how to manipulate objects and controls. The pedagogical agents we have developed are a radically new approach to intelligent tutoring. They are able to help students in ways that conventional intelligent tutors cannot. Agents like Steve that can share the environment with the students can demonstrate to them how to perform tasks if appropriate. Team training in a virtual environment is significant because many of the tasks suited to instruction in a virtual environment, such as those involving dangerous situations or expensive equipment, also involve several people. Our development of participant (team-member) specific capabilities in Vista, VIVIDS, and Steve allows us to deliver team training in a virtual environment.

Research and development for the Vista Viewer concerned two major areas: supporting common model formats to facilitate 3D authoring with best-of-breed commercial software, and providing a uniform framework for immersed interaction using standard VRML sensors. Vista Viewer extensions for presentation and interaction were designed to provide a richer and more meaningful experience for both instructors and students. To accomplish this, Vista software was continually assessed and modified to retain optimal performance levels, novel techniques for immersed manipulation and navigation were developed, multiple devices were supported, and significant VRML 2.0 capabilities, such as Java and JavaScript script nodes, were integrated.

VIVIDS development focused on supporting instruction in the virtual environment, developing and testing large simulations, and supporting team training. A much larger interactive 3D simulation than the one created during the first year of this contract has been developed, demonstrating the robustness of the chosen approach. The instructional authoring and delivery system of VIVIDS has been radically modified to support a much more productive integration with the Vista 3D presentation system. Several new facilities in VIVIDS have been developed in support of team training in virtual environments, including participant-directed instructional primitives and opportunistic instruction.

The primary focus of the USC/ISI team during the past year has been on extending the pedagogical agent (Steve) to allow more natural interactions with trainees. This task included providing a more natural and responsive appearance for Steve. Another key area of research and development involves support of team training capabilities. With respect to enhancements toward the Training Studio in general, USC/ISI provided support for speech input and output, and use of audio to give trainees an enhanced sense of presence in the virtual environment.

Virtual Environments for Training
Annual Productivity Report
October 1, 1996 to September 30, 1997

Submitted by R. J. Stiles
Lockheed Martin Advanced Technology Center
http://vet.parl.com/~vet/

Prepared for
Office of Naval Research
Contract N00014-95-C-0179
August 1997
(CDRL Report A001)

Abstract: This report constitutes the Annual Productivity report for the Virtual Environments for Training contract, funded by the Office of Naval Research. Technical progress achieved in the contract year is described, as well as its relevance to the task of applying virtual environment technology to training. Achievements in opportunistic instruction, event sequencing, and team task modeling for team training are described, as well as advances in a unifying framework for virtual environment interaction, training simulation authoring, and pedagogical agents. We also relate steps in the development of the overall software system, known as the Training Studio, and the individual major components: VIVIDS, Steve, and Vista. This report includes significant contributions from the USC/BTL and USC/ISI sub-contracts, by Dr. Allen Munro and Dr. Lewis Johnson, respectively.

1. Summary

The Training Studio, which allows the application of virtual environments to training, integrates three significant components: Vista Viewer, the virtual environment display system; VIVIDS, the object simulation system; and Steve, the pedagogical agent system. For the Training Studio, the second year of research and development on the Virtual Environments for Training (VET) contract has resulted in improvements and advancements in four technical areas that apply to training: authoring VE simulations, virtual environment interaction, pedagogical agents, and team training. To assess some of the new development, evaluation efforts began during the latter part of the contract year. A study focusing on the effects of 3D vs 2D learning on retention was designed and initiated by the Human Resources Cognitive Training group (HRCT) of U.S. Air Force's Armstrong Laboratory at Brooks Air Force Base in Texas.

1.1 Description of Development

As contract prime, Lockheed Martin Missiles & Space has structured the system architecture and approach to allow independent development of capabilities in each software component, and yet at the same time, sharing of interdependent data across components, through the Communications Bus abstraction, and the TScript message protocol.

The research emphasis for the Vista component was improving the interface to accommodate more natural behaviors for immersed user interaction with the environment. Extensions were made to increase the variety of virtual environment devices that could be used; for example, the pinch gloves. Extensive and ground breaking work in VRML provides for natural interactions with objects, finally delivering a unifying framework for virtual environment interaction. Another focus of research was facilitating the acquisition, use and development of models for domain authors. Vista's ability to display common model formats allows the reuse of existing models as well as formats such as VRML 2.0 for which powerful commercial authoring tools are readily available.

Vista acts as the focal point for interaction between the immersed student, the training simulation software, the pedagogical agents, and the physical 3D representations of the world. As such, Vista development has a large support role for the other software components, providing answers to queries about the spatial properties of the world, interaction with complex objects, and answers to queries about the immersed student. The work on categorizing primitive capabilities that are to be used by the other software components is very important, suggesting a list of services that are common across virtual environment display systems for training. Indeed, at Brooks AFB, they have reproduced variants of visual display software along the lines originating with Vista for internal projects.

The primary focus of University of Southern California (USC) Information Sciences Institute (ISI) in the Virtual Environments for Training project has been on the development of intelligent training technology suited to use in virtual environments. Our approach has centered on building training capabilities into *pedagogical agents* that can share the virtual environment with trainees and provide individualized instruction in response to trainee needs. An agent architecture called Steve (Soar Training Expert for Virtual Environments) was developed in the first year of the effort, and an initial version of the architecture was demonstrated. In the second year we extended Steve's capabilities so that he can interact more naturally with trainees, has a more natural and responsive appearance, and is able to support training of team tasks. Continuing progress was made on using advanced demonstration-based techniques to author agent behavior.

In addition, USC/ISI provided other kinds of support for development of the VET training system, including speech input and output, and use of audio to give trainees an enhanced sense of presence in the virtual environment.

At USC Behavioral Technology Laboratories (BTL), there has been a continued effort to extend the RIDES authoring environment to support the development of interactive simulations and simulation-centered tutorials in virtual environments. The prototype authoring system for building simulation behaviors and structured tutorials for virtual environments is called VIVIDS (Virtual Interactive Intelligent Tutoring System Development Shell). This work tests the hypothesis that the 2D behavior authoring interface of RIDES can be adapted and extended to provide a effective and natural way to specify simulations for virtual training.

1.2 Technological Significance

The technological significance of our development for the past year lies in four areas that apply to training: authoring VE simulations, virtual environment interaction, pedagogical agents, and team training.

The authoring of VE simulations for training has been improved on several fronts. We have improved simulation authoring by more closely tying VIVIDS to the virtual environment, adapting instructional primitives for VE, providing an instructional control interface, and testing VIVIDS on large simulations. Authoring by example for Steve was also improved, to re-use previous tasks already authored, and to reset VIVIDS state when generating examples. Support for Immersed use of VRML 2.0 was dramatically improved, allowing authoring of 3D models using commercial, high quality VRML authoring tools. By improving the authoring capabilities of the three main Training Studio components, it is more practical to consider developing a 3D, immersed training simulation for a given domain.

The VIVIDS authoring system constitutes the first system for authoring (as opposed to programming) robust complex interactive simulations for virtual environments. Furthermore, these authored simulations have features that support the automatic construction of certain types of structured tutorials. The combination of productive simulation authoring with efficient tutorial development is designed to make feasible the application of virtual environment technologies to a very wide range of technical training requirements.

The quality of virtual environment interaction is very important for training in equipment operations and maintenance. Tasks to operate equipment always involve manipulating controls, and maintenance almost always involves assembling, disassembling, and replacing equipment. Researchers in the computer graphics field, such as Mark Mine, Fred Brooks and Carlo Sequin, state that *virtual environments lack a unifying framework for interaction*, and elaborate *that knowledge on how to manipulate objects or controls cannot be "stored in the world"*. [Mine 97] By adapting VRML 2.0 for immersive use, we have realized a unifying framework for interaction, that allows storing knowledge of how to manipulate objects and controls. To accomplish training, we had to go further and provide support for sensing and querying the state of the world by other training software components such as VIVIDS and Steve. This is significant because it takes us beyond selecting objects in the world, and allows authoring the manipulation and use of complex objects for training.

Pedagogical agents are a radically new approach to intelligent tutoring. They are able to help students in ways that conventional intelligent tutors cannot. Agents like Steve that can share the environment with the students, can demonstrate to them how to perform tasks if appropriate. Conventional intelligent tutors have a limited ability to demonstrate and mainly provide commentary on actions that the students make. Pedagogical agents can be easily used to support team training, since they can participate with the students in team activities. They can make use of non-verbal cues and gestures to provide subtle feedback to students, and to interact with students in a more natural, human-like fashion. There is evidence that giving agents an interactive human-like appearance improves student motivation. Finally, agents can be used to more closely emulate the way training is performed in the real world with

human tutors. In particular, we are using agents to simulate the way the U.S. Navy currently trains crews in surface ships, where teams of trainers interact one-on-one with members of trainee crews in team exercises.

Pedagogical agents also constitute a technical advance in autonomous agent technology itself. Agent research has yielded advances in the ability of agents to perform tasks autonomously, but agents that are able to perform tasks cannot necessarily explain what they are doing, or teach the tasks to others. Pedagogical agents extend agent capabilities in these important directions. "Believable agents" (as agents with a lifelike appearance are called) often have limited intelligence – the focus is on giving an impression of intelligence rather than intelligence itself. In contrast, pedagogical agents require a significant level of intelligence; a superficial impression of intelligence is not enough for an agent that can assume a variety of instructional roles.

Pedagogical agent technology relies heavily on, and naturally supports, virtual environment (VE) technology. It makes use of the position and orientation data that VE's collect to track students more closely and provide a wider range of feedback. It permits natural interaction with the intelligent tutoring system, just as VE technology permits natural interaction with the simulated environment.

Team training in a virtual environment is significant because many of the tasks suited to instruction in a virtual environment, such as those involving dangerous situations or expensive equipment, also involve several people. Damage control onboard a ship and casualty control to maintain equipment performance while underway are examples where virtual environments that support team training are useful. To accomplish team training in a virtual environment, we have structured our shared graphics environment (Vista) to support participant-specific changes to the world state, and participant-specific queries for world state, as well as maintenance of shared state. We have modified VIVIDS to allow presenting event sequences, and participant-specific opportunistic instruction that is sensitive to the team task progress. We have greatly augmented our pedagogical agent's task representations to support team tasks, and team roles for agents, whether monitoring a live student's action, or acting as missing team member. We have also explored representing the hand-off and task expectations between team members. Our development of participant (team-member) specific capabilities in Vista, VIVIDS, and Steve allows us to deliver team training in a virtual environment.

2. Introduction

This report describes the research and development efforts with respect to the Training Studio, an authoring and presentation system for training in a virtual environment. Enhancements were made to the three main components of the system to provide better authoring tools, enhanced training capabilities, and more natural and useful interactions for the immersed user. An emphasis on team training functionality was placed on the development efforts during this second year. An evaluation study was designed and initiated during this first year. Human Resources Cognitive Training group (HRCT) of the U.S. Air Force's Armstrong Laboratory at Brooks Air Force Base in Texas, is using the Training Studio to study the effectiveness of 3D vs 2D learning with respect to retention.

One purpose of our work is to explore, develop, and evaluate novel techniques for incorporating automated instruction in virtual environments. USC/ISI's focus has been on incorporating pedagogical capabilities in an intelligent agent architecture called Steve. We are investigating the following hypotheses: 1) that an agent architecture and knowledge representation can be developed that permits autonomous agents to act as guides, mentors, and team members, 2) that machine learning and high level languages can be employed to assist instruction developers in creating agent-based instruction, and 3) virtual environment technology enables new types of interactions between trainees and

instructional systems, which improve the quality of instruction provided by the instructional systems.

In the current year we have performed a number of investigations addressing these hypotheses, and have in the process have greatly improved the capabilities of the Steve pedagogical agent. Steve can switch instruction and interaction styles more rapidly and support more natural interaction between trainees and Steve agents. Steve can participate in team activities, and provide instruction in team settings. Steve now has improved dialog capabilities, and tracks the student's field of view when demonstrating tasks. Work has begun on speech input, team training capabilities, and providing Steve with emotional sensitivity. Steve now makes effective use of gaze to signal his focus of attention; this has proved to have a dramatic effect on Steve's believability as a pedagogical agent. The current contract year saw Steve tested in immersive environments for the first time.

At Behavioral Technology, research and development during the second year of the VET contract have centered on three major issues.

- How can VIVIDS be made a more effective and productive tool for developing and delivering lessons for students in the immersed Vista environment?
- Will the successful development of the High Pressure Air Compressor (HPAC) simulation scale up successfully to the much larger Gas Turbine Engine (GTE) control system?
- How must VIVIDS be modified to provide support for team training?

Vista Viewer extensions for presentation and interaction were designed to provide a richer and more meaningful experience for both instructors and students. To accomplish this, Vista software was continually assessed and modified to retain optimal performance levels, novel techniques for immersed manipulation and navigation were developed, multiple devices were supported, and significant VRML 2.0 capabilities, such as Java and JavaScript script nodes, were integrated.

During the second year of the VET contract, the Lockheed Martin VET team included: Randy Stiles (VET Program Manager), Sandeep Tewari, Mihir Mehta, and Laurie McCarthy. The USC/BTL team for the VET project consisted of the following individuals: Dr. Allen Munro (Principal Investigator), and Dr. Quentin Pizzini, assisted by Mark Johnson, David Surmon, James Wogulis, Josh Walker, and David Feldon. The USC/ISI team consisted of the following individuals: Dr. Lewis Johnson (Principal Investigator), Dr. Jeff Rickel (Research Scientist), Erin Shaw (programmer), Marcus Thiebaux (programmer), and Richard Angros (Graduate Student). Mr. Angros was supported full time on the project during the end of May; afterwards his effort was transferred to the new AASERT grant from ONR. Ms. Shaw contributed some effort to the project for a short time to assist in developing Steve's graphical realization. Mr. Thiebaux was hired onto the project during the summer, and is currently working on the graphical appearance of Steve. Ben Moore, an undergraduate student supported by the AASERT grant, has been contributing to the effort in the areas of speech recognition and use of sound. Two additional graduate students, Behnam Salemi and Art Kroetz, have also been working with the project team, in preparation for future research investigations. Evaluation support and design was performed by Dr. Craig Hall, USAF AL/TRAIN, and Ms. Carol Horwitz, USAF AL/HRTC.

3. Methods, Assumptions & Procedures

Our research approach is to apply virtual environments to training, by taking steps to improve the authoring of virtual environments simulations for single students or teams. Our development approach is component based. Messages are passed between software components to accomplish training for each participant in a virtual environment. Each participant has a display process (Vista) that allows them to interact

with a shared world, which is simulated by VIVIDS. Instruction takes place using VIVIDS and the pedagogical agent Steve.

Team training, where more than one person must be trained to carry out a team task, was proposed as part of the VET effort. We are concerned with instruction for teams in an immersive virtual environment. Our hypothesis is that with the ability to capture more student interactions with a virtual environment, a successful team training system can be realized. The steps in testing this hypothesis are to: identify a domain that exercises team training, analyze the team tasks to determine issues and scope, determine representations and approach, then build tools and capabilities to realize team training for that domain. This is the first pass, another pass should be taken with a different domain, to see if tools and representations are general enough for authoring use. Then, when these aspects are in place, the efficacy of the team training tools and approach should be tested on a statistically significant body of subjects.

Our team training development and approach primarily addresses the first part, deferring designing and conducting experiments on efficacy of the team training approach and tools to the period after lessons have been learned through building the prototype testbed. Of course, from the start we have been guided by previous research into what approaches work or don't work for team training.

To discuss team training in a meaningful way, it's important to be able to classify aspects or qualities of team training. Are the teams large teams with a control hierarchy in place, or are they small teams with at most one level of control hierarchy? Are the roles of team members fixed during for the tasks being trained, or are they fluid and overlapping, where any one of the team can be doing the same type of task as the other, or helping the other team member do theirs? Are the teams asynchronous in time, where a team member may carry out a task and leave the results for other team members to use at a later date, or are they roughly synchronous, where most of the team members must be present or in communication at the same time in order to conduct the team task?

We refer to these qualities as small/large teams, fixed/fluid teams, and synchronous/asynchronous teams. For the scope of our effort, we are addressing small teams with at most one level of control hierarchy, with fixed or fluid roles, and synchronous time coordination.

Lockheed Martin's approach for Vista focuses on providing those capabilities that facilitate communications and scene display and manipulation for Steve and VIVIDS, as well as optimizing interactions within the virtual environment. New capabilities are developed, tested, and released to collaborating VET organizations for further evaluation and critique. Other capabilities are developed in response to a direct request by one of the other team members; or provided as a solution to a problem encountered by one of the collaborators. The Lockheed Martin team keeps abreast of developing technologies in VE-related fields. By beta testing software such as SGI's Cosmo Worlds and Optimizer, and actively participating in VRML standards efforts, Lockheed Martin is ensuring that new standards and tools address VET requirements.

The USC/BTL methodology has been to progressively adapt RIDES and VIVIDS functionalities to provide appropriate simulation and instruction services for a virtual environment delivered by Vista and to provide services to the Steve autonomous agent. These new capabilities are tested by developing simulation and instruction materials using the revised authoring tools. Two levels of formative evaluation are pursued: both the usability of the revised authoring system and the functionality of the tutorials it produces must be examined. Based on, first, in-house evaluations, and, after initial revisions, the evaluations of our research partners, further changes are made, and the authoring and testing cycle resumes.

The USC/ISI research methodology is as follows. We identify a new capability that, if incorporated into Steve, would contribute to

validating one of our research hypotheses. We then design a set of extensions to the Steve system that implements the capability. We develop a prototype implementation of the capability, and conduct a series of demonstrations and in-house tests. We then make arrangements for further evaluation of the capabilities by our partner organizations.

4. Results and Discussion

Lockheed Martin, USC/BTL, and USC/ISI each accomplished major milestones regarding development of their respective components: Vista Viewer, VIVIDS, and Steve. The groups also worked together to ensure continuing coherency of the Training Studio. An evaluation study, led by HRCT of the U.S. Air Force's Armstrong Laboratory at Brooks Air Force Base in Texas, uses the Training Studio to study the effectiveness of 3D vs. 2D learning with respect to retention.

4.1 Simulation-Based Training

VIVIDS development during the second VET contract year focused on three major areas: supporting instruction in the virtual environment, developing and testing large simulations, and supporting team training. The USC/BTL team completed a successful year of development:

- A much larger interactive 3D simulation than the one created during the first year of this contract has been developed, demonstrating the robustness of the chosen approach.
- The instructional authoring and delivery system of VIVIDS has been radically modified to support a much more productive integration with the Vista 3D presentation system.
- Several new facilities in VIVIDS have been developed in support of team training in virtual environments, including participant-directed instructional primitives and opportunistic instruction.

4.1.1 Modifying VIVIDS to support instruction in a virtual environment.

Several modifications have been made to VIVIDS to make it collaborate more effectively with Vista, Steve, and TrishTalk, in order to provide more effective instruction and to make instructional development more productive.

Support for Speech in Instruction. A mechanism was provided to permit tutorial authors to specify that all instructional text output should be directed to speech. The TrishTalk speech system developed at USC/ISI is used to deliver this speech. The author can optionally also specify that the text be directed to an instructional text window in the 2D VIVIDS environment.

Initialization Objects. VIVIDS library objects in support of session initialization have been developed and are in use in the GTE simulation and our recently revised HPAC simulation.

VET-integrated Instructional Primitives. The following instructional item types of VIVIDS have been modified so that they work appropriately with VIVIDS.

Keypad Question
Menu Question
Text Presentation
Topic Presentation
Wait for Student
Free Play
Highlight Object
Set Control
Read Indicator
Find Object

At this point, all the primitive instructional actions of RIDES are supported by VIVIDS in the Vista environment, except for keyboard items and URL presentation items.

The Student's Instructional Control Interface. A new advanced approach to providing a user interface for student control over VIVIDS-based instruction was developed. These user interface elements are tightly integrated with the VIVIDS presentation code, so that

authors are not required to carry out special steps to make instruction work in 3D environments.

As an aid to the use of the VET system for exploratory free-play learning, the user interface has been extended to provide the functionality of moving and orienting the student to a "Home" position and orientation when the student wants. In this mode the student also has access to content-based help. This is the information that authors have entered into VIVIDS knowledge units that have been associated with simulation objects. Students can access menus of authored topics and hear TrishTalk read the content of these topics.

4.1.2 Developing and testing a very large VET simulation

The High Pressure Air Compressor (HPAC) simulation developed during the first year of this project was a simulation of moderate size. It's success has encouraged us to develop a much larger simulation of the Gas Turbine Engine (GTE) system on Arleigh Burke class ships. The GTE simulation is now the largest RIDES or VIVIDS simulation ever built at Behavioral Technology Laboratories. The 2D representation of the simulation has 44 graphic scenes with thousands of objects. Of these, 825 have thus far been associated with 3D model objects in the Vista world. As additional Vista model objects are developed, they will be linked to the corresponding simulation objects that have already been created in VIVIDS, and the simulation will be refined to best support immersed interactions.

4.1.3 Supporting team training in VIVIDS

Team training support in VIVIDS has two major aspects: simulation features and instruction features.

4.1.3.1 Behavioral Simulation Support for Team Training

Two approaches to team training simulations are supported in VIVIDS: several students interacting with a single simulation and multiple students interacting with networked simulations.

Several students working with a single simulation. To this date, this has been our chosen mode of simulation support for team training in the VET project. Several Vistas (one for each student participant) all receive broadcast requests from VIVIDS to modify the appearance of the 3D world when the simulation engine modifies an attribute that controls the visual world. When any student takes an action that could affect the simulation, that student's Vista informs VIVIDS of the student action, and the single VIVIDS simulation computes all the appropriate consequences of that action.

Multiple students interacting with networked VIVIDS simulations. This mode of interaction was originally intended to support interactions between loosely coupled training simulations, where only a few attribute values need to be exchanged. It may prove to be an appropriate approach to follow in the VET project, if we find that simulation computation requirements are so high that the simulation needs to be divided into two or more different communicating simulations running on different processors. There are three distinct methods that could be employed to link collaborating VIVIDS simulations. The first is a long-standing RIDES method that was explicitly designed for this purpose. The second is to make use of TScriptSend messages. The third is to employ a new high-speed socket system designed to support open communications in VIVIDS. The first approach has the advantage of being well-tested, but it uses a closed protocol that supports only RIDES-to-RIDES (or VIVIDS-to-VIVIDS) communications. The second approach, burdened by ToolTalk's inefficiencies, may prove to be a bottleneck if the collaborating simulations turn out to be closely coupled. The third approach seems to offer the best opportunity for supporting good performance while offering the potential for communications with other application components in the future.

4.1.3.2 Instruction Support for Team Training

There are four major approaches to supporting team training in VIVIDS:

- Content-based help on request for team members
- Structured lessons with particular instructional items directed to particular team members
- Opportunistic instruction directed to relevant team members
- Scenario delivery to teams

Content-based help for team members. The RIDES 2D tutoring system provides an optional facility for authors to make content-based help available to students. Much of this information is authored in *knowledge units* that can be associated with simulation objects, but some is generated from the current state data for the object. The help information includes the name of a selected object, its control or indication state, and a list of relevant topics for which textual discussions were available. If the object is of a type that can exhibit failure behavior, the menu also includes an option to replace it. In order to make similar capabilities available in the VET team training environment, several hurdles had to be surmounted. First, a user interface was required to let students choose the mode that supports selections for content-based help. Second, a menu of the information items that are available for a particular object had to be provided. Rather than display a popup menu on or near the object, as in RIDES, VIVIDS asks Vista to display this object-specific menu on the palette that is on the student's left hand. Third, if a student selects one of the textual discussions from this menu, then VIVIDS requests that TrishTalk speak the selected discussion. At present, these features are available to students using VIVIDS in its exploratory, free-play mode.

Instructional steps directed to particular team members. VIVIDS can now deliver structured lessons that direct different parts of the lesson to different team members. For example, during a team training structured procedure lesson, a student playing one role would be directed to carry out certain actions, while a student playing another role would carry out a different set of actions. Implementation of this feature required modifying the nature of ten of the twenty-five VIVIDS primitive instruction types (Set Control, Read Indicator, Find Object, Keypad Question, Menu Question, Text Presentation, Topic Presentation, Goal, Highlight Object, and Wait for Student) to support their assignment to particular student participants. The special participant name "All" is supported. This directs text/speech to all student participants and allows any of them to carry out a requested action or to provide a requested answer.

A special case of instruction items directed to team members is instructional items directed to Steve. It will be possible for an author to specify, for example, that certain Set Control instructional items should be performed in demo mode by asking Steve to carry out those steps. This will make it possible for VIVIDS and Steve to collaborate during structured lessons, by having Steve help deliver VIVIDS remediations by demonstrating the required operations.

Opportunistic instruction. Opportunistic instruction presents a (usually brief) instructional item or group when a student does something that has been marked as a trigger for that instruction. To support team training, VIVIDS will have to be able to present opportunistic instruction only to the student that triggered it, or to the entire team.

Opportunistic instruction can now be authored using the RIDES/VIVIDS event mechanism. Authors can create events with simple or complex conditions that determine when a particular piece of opportunistic instruction is to be presented. The author can specify that the opportunistic instruction will be activated only when certain lessons are being played. In the body of the event, the author specifies what instruction will be played and in what mode. A new event language element, *PlayInstruction* has been added to VIVIDS. *PlayInstruction* has three required parameters: the name of the instructional node to present, the mode in which the instruction will be presented (Demonstrate, Practice, or Test), and a specification of what to do when the opportunistic instruction ends. Two choices are available for this last parameter: "Stop" ends the current lesson and supplants it with this new opportunistic

instruction; "Resume" specifies the intrusion of the opportunistic instruction, but then allows the original lesson to resume at the point where the intrusion took place. An optional fourth parameter for *PlayInstruction* specifies a TimeOut value, after which the intrusive instruction is to end. The instruction node that is presented need not be the top node of a whole lesson. A single leaf node (such as a warning message in response to a safety violation) can be presented, or a small group (such as highlighting a particular control, requiring its manipulation, and then unhighlighting it again). In order for *PlayInstruction* to work, the control structure of VIVIDS will be modified to allow instructional interruptions and to support the suspension and resumption of lessons.

Scenario delivery. A scenario is an authored set of simulation (and sometimes instruction) events that are to occur under particular conditions and/or at particular times. Many team training exercises call for the delivery of scenarios that provide the team members with a sequence of events that they must deal with together. Scenarios are supported through the use of VIVIDS events. An example of scenario delivery will be developed for the GTE training environment later this year.

4.1.4 VIVIDS Future Development

Several additional enhancements are expected to be completed in the current contract year.

The GTE simulation and tutorials. The GTE simulation will be completed. As the Vista model for GTE grows, new model objects will be linked to the corresponding VIVIDS simulation objects so as to provide the correct interactive behavior in the 3D world. In addition, simple VIVIDS tutorials will be created for the GTE domain. We plan to visit the Philadelphia Naval Base in support of our GTE knowledge acquisition process.

Better simulation support for autonomous agents. Two new VIVIDS features in support of autonomous agents are planned.

- Object name export in support of Steve.
- Configuration definition and loading under the control of autonomous agents, in support of tutor development by agent exploration.

Improved team training features. Although certain team training features have been very successfully implemented in VIVIDS, including the essential participant-directed instructional items, more remains to be done in the current contract year. There are bugs in our implementation of opportunistic instruction; the feature works correctly only for certain simple opportunistic interactions. These problems are to be remedied before the end of the second year. In addition, a demonstration of an instructional scenario for team training is to be developed for the GTE domain.

4.2 Human Task Modeling

The primary focus of the USC/ISI team during the past year has been on extending the pedagogical agent (Steve) to allow more natural interactions with trainees. This task included providing a more natural and responsive appearance for Steve. Another key area of research and development involves support of team training capabilities. With respect to enhancements toward the Training Studio in general, USC/ISI provided support for speech input and output, and use of audio to give trainees an enhanced sense of presence in the virtual environment.

4.2.1 Graphically Representing Steve Agents in the Virtual Environment

Unlike most intelligent tutoring systems, Steve inhabits the virtual world along with students (and perhaps other agents). For teaching physical tasks, like operation and repair of equipment, this allows valuable interactions between Steve and students. For example, Steve can physically demonstrate task steps; this could be particularly helpful for students when those steps involve spatial motor skills. He can also draw students' attention to objects by pointing at them. Steve's embodiment will be especially useful for team training (discussed in Section 4.2); when Steve is used to fill the role of a missing team member, it will be important for other team members to keep track of Steve's

activities. These types of interaction between Steve and students would be impossible with a traditional disembodied tutor. Moreover, this approach is more flexible and interactive than video, because Steve can adapt domain procedures to the student's current situation.

While the motivation for embodiment of Steve is clear, the type of embodiment is not. Since Steve is teaching physical tasks, some variant of a human form seems most appropriate. The question is how much detail is needed. During the first contract year of the VET project, we explored two ends of the spectrum. On the simple end, we represented Steve simply as a hand. This allows Steve to manipulate and point at objects, which may be sufficient for many applications. At the complex end, we represented Steve as a full human figure, using the Jack software developed at the University of Pennsylvania [Badler et al. 93]. While a full human figure is more difficult to control and more visually obtrusive (a serious concern given the limited field of view offered by current head-mounted displays), it provides the highest fidelity for physical demonstrations. Although our experience with Jack suggests that the technology for full human figures is still not sufficiently mature for our purposes, and is still difficult to integrate with other technologies, we are continuing to track this area because of its potential.

In this second contract year, we have explored an interesting intermediate point on the spectrum: representing Steve as a head and hand. This representation provides the benefits of a hand, such as the ability to manipulate and point at objects, yet the use of a head opens up many new possibilities. First, we use the head to help the student track Steve's location as he moves through the virtual environment from object to object. Second, the head allows the student to track Steve's gaze. This lends coherence to Steve's demonstrations. Steve looks at people when he talks to them, he looks at objects when he manipulates them, he looks where he is going when moving, he looks at objects that the student is manipulating, and he looks at the student (or a teammate in team training tasks) when he is waiting for them to do something. Gaze plays many important roles in human interactions [Cassell et al. 94, Pelachaud et al. 96, Thorisson and Cassell 97], and we hope to leverage more and more of these in Steve to provide more natural human-agent interactions. In addition to gaze, we are also beginning to experiment with other important pedagogical uses of a head. For example, Steve could use a nod of approval to show agreement with the student's actions, and a nod of disapproval or look of puzzlement to make the student think twice. Such nonverbal feedback is less obtrusive than the verbal interjections used by previous tutoring systems, thus giving Steve a wider range of types of feedback that he can provide. We hope to have some initial use of such feedback by the end of this contract year, and we will continue to explore this area in the third contract year.

Besides the addition of a head, we made other improvements in controlling Steve's graphical representation this past year. First, we extended the range of object manipulations for which he is capable. Steve is now capable of pressing objects (e.g., buttons) and grasping objects in order to turn, pull, push, and move them. This provides a useful set of building blocks for defining new motor actions. Second, we added the ability for the course author to define the orientation of objects, to facilitate Steve's manipulation of them. We plan to continue such development; our design philosophy is that Steve should have default motor actions for manipulating objects, while providing hooks to allow the course author to provide additional knowledge to customize his motor actions for particular objects.

We plan to continue evaluating different graphical representations for Steve. From the beginning of this project, we have designed Steve so as to facilitate such experimentation. The cognitive component of Steve, implemented in Soar [Laird et al. 87, Newell 90], sends out a variety of abstract motor commands, such as gazing at an object or pressing a button. A separate module controls Steve's motor actions; it receives the motor commands and converts them into appropriate operations on Steve's graphical representation. Thus, we can implement a

new graphical representation for Steve simply by plugging several replacement procedures into this module. During this second contract year, we further structured this module and its interfaces to make it still easier to plug in different graphical representations for Steve. This not only facilitates our own experimentation and evaluation, but will also make it easier for course authors to select an appropriate graphical representation for their domain.

4.2.2 Team Training

We have been exploring two roles for Steve in team training. First, as in individual instruction, Steve can serve as a coach for a single student. Each human team member, therefore, can have their own Steve agent helping them with their role in the team task. This mirrors the approach taken at Great Lakes to train Navy personnel to perform team tasks; each trainee has an instructor that follows them, providing assessment and assistance. However, these instructors, and their agent counterparts, cannot function simply as tutors for individual tasks. In addition to understanding their trainee's task, they must understand the dependencies among team members, and teach these dependencies to their trainee. To our knowledge, there has been very little work on this problem in the intelligent tutoring systems community. The second role Steve can play in team training is as a substitute for a missing team member. If different Steve agents can play the role of various team members, a single student can practice a team task even when his human teammates are unavailable. A third possible role for Steve in team tasks is as a coach for the entire team; this is a potentially interesting area, but we have not explored it yet.

Our work in the first contract year provided important support for team training. Most importantly, Steve does not execute tasks by rote, nor does he require this of students. As described in last year's report, and in subsequent papers [Johnson et al. forthcoming, Rickel and Johnson 97a, Rickel and Johnson 97b], Steve continually monitors the state of the virtual world and determines how to complete the current task. Thus, Steve can adapt standard procedures to handle unexpected events. This is an important requirement for team tasks, where multiple students and agents are all acting in the same world.

During this second contract year, we made two main extensions to handle team training. First, Steve's task representation was extended to include the different roles and responsibilities in team tasks. As developed in the first contract year, Steve models tasks as hierarchical, partially ordered plans [Russell and Norvig 95]. A task description includes a set of steps to be performed (each one either a primitive action, such as pressing a button, or a subtask), a set of ordering constraints, and a set of causal links. The causal links [McAllester and Rosenblitt 91] describe the causal relationships among the steps, i.e., how one step achieves a goal that is a precondition for another step or for completion of the task. For team tasks, we added a representation of the different roles within each task (or subtask), and the steps in that task for which the team member playing that role is responsible. We also added the ability to represent speech acts as steps of a plan; specifically, a course author can specify the point in the task at which one team member should inform another of some information. Such steps are commonly specified in the Naval procedures we studied (see Section 4.7).

The second main extension was to generalize Steve's abilities to demonstrate tasks, as well as monitor students performing tasks, to handle team tasks. Given an assignment of team members (both humans and agents) to roles in a task, Steve uses his task model to determine who is responsible for the different task steps, and he uses this understanding of responsibilities when selecting his own actions and when assisting students. When performing as a missing team member, Steve performs those steps for which he is responsible, waiting for teammates where appropriate. When demonstrating a role to a student, Steve both performs the steps for which he is responsible and explains what he is doing. Finally, when monitoring a student performing a role in a team task,

Steve's assistance is now sensitive to his understanding of responsibilities. That is, when the student asks "What should I do next?", Steve's answer is based on the student's responsibilities within the team.

We are still exploring the types of knowledge that students must learn about team tasks, and how Steve can provide such knowledge. Steve's assistance sometimes helps a student understand the relationship between their own steps and their teammates' steps, but we plan to do more in this area by the end of the contract year. For example, if a student asks "What should I do next?", we would like Steve to be able to give answers like "You don't need to do anything yet, but as soon as Joe gathers the oil sample, you should analyze it." Steve's task model provides such knowledge; our challenge is to identify situations in which such answers are useful.

4.2.3 Improving Steve's Discourse Capabilities

Steve's interactions with students can be viewed as tutorial, task-oriented discourse. There has been a long line of research in the computational linguistics community on both tutorial discourse and task-oriented discourse. Our objective in the VET project has been to complement such research; we have focused on the novel opportunities provided by virtual environments, such as the possibility of an embodied tutor that can demonstrate tasks, rather than the natural language aspects of tutorial discourse. However, where appropriate, we have drawn on that extensive literature. In this second contract year, we extended Steve to leverage two valuable ideas from that literature: cue phrases and a discourse focus stack.

Cueing phrases, such as "Next, ..." and "In order to ...", have been studied extensively. Their primary role is to help a listener understand the rhetorical relation among different utterances. Our initial experience with Steve showed a clear need for cue phrases; his demonstrations sounded like an unstructured explanation of one step after another, with no clear relation among them. To help students follow the structure of tasks, we extended Steve to use cue phrases. As Steve gives a demonstration, he now keeps track of the last task step he described. Then, when explaining a task step, Steve uses a cue phrase to show the relationship of the step to the previous step. For example, if the current step is the first step of a subtask that was just introduced, Steve will say "First, ...". Or, if the current step was enabled by the previous step, Steve will say "Now we can ...". Steve currently has six different cue phrases that he selects dynamically, lending more coherence to his demonstrations. We also added a rhetorical pause in between Steve's demonstration of one step and the next one. This not only helps to structure the discourse, but also gives the student an opportunity to interrupt Steve. We intend to extend Steve's range of cue phrases as his development progresses, but his current use of cue phrases provides a marked improvement over his earlier demonstrations.

Like cue phrases, discourse focus has been studied extensively. Discourse focus concerns the orderly progression of one utterance (or topic) from another. In their seminal paper on the structure of discourse, Grosz and Sidner [1986] proposed that discourse focus is governed by a stack-like mechanism. That is, new topics can be pushed onto the stack when they are a subtopic of the topic currently in focus (at the top of the stack). When a topic is complete, it is popped off the stack, and focus returns to the next item on the stack. To improve Steve's discourse, we added such a focus stack to Steve. The focus stack keeps track of topics (e.g., subtasks) Steve has started (but not completed) discussing. He uses the focus stack to guide his behavior during a demonstration. The focus mechanism provides two main improvements to Steve. First, if subtasks can be done in any order, Steve would sometimes interleave them, jumping back and forth from one to the other; now, Steve uses the focus stack to make his demonstrations as coherent as possible, interrupting the current subtask only for good reason (e.g., an unexpected event). Second, the focus stack allows Steve to recognize discourse situations that he could not previously recognize, such as a return from a digression or interruption, and choose appropriate cue phrases (e.g., "Now, where

were we?"). We have not yet implemented all these new cue phrases, but the discourse focus stack was a necessary prerequisite.

4.2.4 Tracking the Student's Location and Field of View

Virtual environments make it easy and natural for students to control their own view of the virtual world. In the VET system, students wear position and orientation sensors on their head and hands. The Vista software uses the sensor data to continuously update each student's location and field of view, and it makes this information available to Steve. By controlling their own field of view, students learn to navigate around their work environment, and they can view objects from different angles. In contrast, most tutoring systems, and even multimedia presentation systems [Maybury 93], assume they can design and control the student's view. The Vista software allows Steve to control the student's field of view when necessary. However, to avoid losing the benefits of having students control their own view, we have ignored that option. Instead, Steve dynamically adapts his presentation to the student's location and orientation.

Steve uses his knowledge of a student's location in several ways. In the first contract year, we made Steve's pointing ability sensitive to the student's location. When pointing at objects, Steve orients his hand perpendicular to the student's line of sight. This simple technique ensures that the student can clearly see the hand and the referenced object. During this second contract year, with the addition of Steve's head, we made his gaze sensitive to the student's (and other agents') location. Specifically, when talking to someone or waiting for them to do something, Steve looks at them.

During this second contract year, we have also extended Steve to keep track of the student's field of view. Specifically, based on messages from Vista, Steve keeps track of which objects are in the student's field of view. He makes use of this information in two important ways. First, he uses it to help determine what the student is doing. For example, Steve knows that the student is not checking an indicator light if the light is not in the student's field of view. Second, Steve's demonstrations are now sensitive to where the student is looking. Previously, Steve would plow through a demonstration whether the student was looking or not. Now, Steve will delay demonstrating an action until the student is looking. In addition, Steve exclaims "Look over here!" to inform the student that he is waiting.

4.2.5 Spoken Communication Among Humans and Agents

Team tasks require communication among team members. Communication among agents can be accomplished with artificial languages, while communication among people can be accomplished with natural language. However, the VET system supports teams composed partly of agents and partly of people, and this greatly complicates communication. Moreover, communication among team members is typically via speech, so, to avoid unnaturally constraining team communication, the VET system should support this modality. For these reasons, we have been integrating both text-to-speech and speech recognition into the VET system.

To provide Steve agents with a text-to-speech capability, we have designed and implemented a program called TrishTalk. Each human participant in the virtual world has their own TrishTalk process, which acts as their speech server. Steve agents send text messages to human participants, and these messages are received by that participant's TrishTalk process. In the first contract year, we implemented the core functionality of TrishTalk; when it receives a message, it uses Entropic's TrueTalk software to generate the corresponding speech. During this second contract year, we extended TrishTalk to handle team (multi-agent) environments. First, we added a mechanism that allows TrishTalk to queue up messages if it is still processing a previous speech message; this way, if two agents try to speak to the same human participant at once, TrishTalk will process the messages one after the other. Second, we modified TrishTalk to inform a speaker when the speech channel is not

available (i.e., someone else is talking to the same listener) and the speech request is being queued. This allows the speaker to reason about whether to leave the speech on the queue or cancel it (e.g., if it only makes sense to say it immediately). Third, we added a message by which a speaker can cancel speech requests, either for the reason just mentioned or to simply abort a speech message that is no longer appropriate. Finally, we implemented the ability for agents such as Steve to define their voice so that TrishTalk will use this voice whenever generating speech for them. The ability to define different voices for different agents is crucial in handling team tasks, where multiple agents must work together yet be distinguishable to human team members.

To allow Steve to understand human speech, whether that speech is directed at him or at another human team member, we are integrating Entropic's GrapHVite speech recognition software into the VET system. Our first objective has been to allow users to use speech in place of the immersive graphical user interface. The speech recognizer needs to be able to work with a fair degree of reliability, or notify Steve if the user's utterance is not recognizable, so that Steve can request clarification. To this end we have created a grammar that includes a wide range of variants on the commands that appear in the GUI. GrapHVite uses Markov chains for recognition, which match regular expressions; since regular expressions are not powerful enough in general to parse human language, we find that the grammars we create also admit some ungrammatical sentences. This is actually a desirable feature, since it raises confidence in the parse when it is grammatical. GrapHVite's program interface does not provide numeric estimates of the confidence of its parse, so the best way that we have found to assign confidence is to create Markov networks that permit both grammatical and ungrammatical utterances, and make sure that the recognized utterance is grammatical. Once the speech recognition routines complete a parse and returns a text string representing the utterance, the text is parsed a second time, this time to make sure that the text is an expected grammatical utterance. If so, a message is sent to Steve with the utterance; if not, a message is sent indicating that the utterance was not understood. Steve can than request that the student repeat the utterance, if necessary.

The recognizer for GUI commands is complete, and is currently being integrated with Steve. Once this is complete, we will explore other uses of speech as time permits. The item that is of next highest priority is to be able to the recognize the communications that are required between team members in the Navy tasks that we are developing. Although that per se is not difficult, the challenge is to set up the parser and recognizer in such a way that it is easy for an instructional designer to modify and extend the recognition capabilities without having to understand speech recognition technology in detail or program in Java. We use GrapHVite's graphical user interface to define the Markov network, and list the texts to be parsed in a text file that is read in at configuration time. This simple approach appears to be both powerful enough and easy for nonspecialists to modify.

4.2.6 Authoring by Demonstration

Last October we gave an initial demonstration of our authoring by demonstration capability, called Diligent. During the past year we have been refining Diligent, to extend the range of procedures that can be acquired and to make it easier to interact with the system.

One of the most important extensions is the ability to learn complex procedures – procedures which themselves contain procedures as components. At present Diligent is able to learn complex procedures when those procedures are introduced in the context of demonstrating a larger procedure. The instructor must indicate to Diligent when the subprocedure begins and when it ends; Diligent focuses on the subprocedure until it is learned, and then continues with the main procedure.

Procedures can now include observations as well as actions. If in the course of the demonstration the instructor points at an

object but does not manipulate it, this is taken to indicate that some property of the object is being observed and noted. For example, in demonstrating a procedure that involves turning on and checking a light, the instructor indicates that he is checking the light by pointing at it. Subsequent interaction between Diligent and the user is used to clarify what properties of the light are being observed.

As we have been studying more complex procedures, we have encountered an increasing need for conditional branches in procedures. Such conditionals were not previously supported in Steve, so we are adding them, again following the plan representation described in [Russell and Norvig 1995].

Diligent currently tries to construct a procedure description from a single instructor example, plus additional experiments that it performs on its own. We expect soon to remove this limitation, so that the instructor can present a series of examples, perhaps each appearing in a different context, in order to convey to Diligent the different ways in which the procedure is performed.

Given this basic framework, we are now extending Diligent's acquisition mechanism to support more robust and natural interaction. We wish to support a more flexible approach to complex procedures, so that instructors can choose whether to describe the subprocedures first or work on the entire procedure. We also need to support different roles that subprocedures can perform within the overall procedures. Some subprocedures are procedures in their own right, that can be applied in a wider range of contexts than the one illustrated in the main procedure. Other subprocedures are simply segments of the main procedure. Diligent needs to learn different things about each type of procedure, but both are important in order to manage the complexity of procedures containing many steps.

Currently Diligent controls the interaction between itself and the instructor, requesting information and clarification whenever it needs it. We need to change the interaction style so that Diligent does not force the instructor to provide information until he or she is prepared to provide it, in the manner that the instructor considers to be appropriate. Currently, for example, as soon as Diligent is shown a demonstration of a procedure Diligent makes a guess as to what the goal of the procedure is, and asks the instructor to verify it. In the new approach, the instructor would have the option of verifying the procedure right away or waiting until he or she has given further demonstrations.

4.2.7 Evaluation

During the current contract year Steve's capabilities have begun to reach the point where it makes sense to perform evaluations of effectiveness. We have a series of evaluations planned, some of which we will perform at USC/ISI and some of which will be performed in collaboration with our colleagues at Armstrong Laboratory.

One question of interest to us is how the believability of Steve as an interactive character , with personality, contributes to learning effectiveness. A student in the USC School of Education named Art Kroetz plans to investigate this issue. First, we need Steve's face to be more expressive, to convey intentions and emotion. We have recently received permission from Ken Perlin's group at New York University to obtain a copy of their Improv agent construction tool, which is able to express emotions and attitudes. Once we have applied Improv techniques to Steve, we can then conduct lesion study evaluations. We are interested in finding out what subjective impressions subjects have of Steve, as well as how the presence of Steve improves learning efficiency and retention.

We are also ready to study the effectiveness and perceived appropriateness of Steve's pedagogical styles, both from students' perspectives and from the perspective of instructors. We are interested in finding out whether Steve provides the kind of instructional support that students desire, and that instructors consider to be appropriate. We expect that Armstrong Laboratory will provide us guidance and assistance in this area.

In the mean time, our primary evaluation focus will be on the problem of determining the benefits of 3D environments in supporting learning, as opposed to 2D environments, as directed by ONR. Following Rickel's suggestion, Armstrong's evaluation is focusing on retention effects, the hypothesis that learning a task in an immersive environment will help students better to retain it.

4.2.8 Impact on the Research and Development Communities

The idea of autonomous agents in general, and pedagogical agents in particular, is still relatively new, but it is having a surprising impact on the research community. Attendance at the First International Conference on Autonomous Agents, chaired by Lewis Johnson, far exceeded all expectations. The Workshop on Pedagogical Agents, chaired by Jeff Rickel, which is to be held in conjunction with the World Conference on AI in Education, received more submissions than any other workshop at that conference except the one on World Wide Web applications. An upcoming IJCAI workshop on Animated Interface Agents, whose program committee includes Jeff Rickel, has two sessions devoted to pedagogical agents. Through these conferences and workshops, our concept of a pedagogical agent, which has developed primarily through the VET contract, seems to be spreading rapidly.

The reactions from senior researchers when they see Steve is equally encouraging. Dr. William Clancey, a long-time leader in the field of intelligent tutoring systems, was impressed by a demo of Steve, and he asked for a video that he could show to Dr. John Seely Brown and Dr. Daniel Bobrow at Xerox PARC. The Air Force Armstrong Labs has been giving a number of demos of Steve, and Dr. Wes Regian described people's reactions by saying that Steve is "big news." Jim Fleming related that Prof. Ed Feigenbaum, Chief Scientist for the USAF, when he saw one such demo, said "This is very significant." Steve is still an early prototype, and many important research questions remain, but we are very encouraged by the potential people see in this research direction.

Our work on Steve is also already producing spin-off development projects. The Air Force Armstrong Laboratory has provided funding to support the commercialization of Steve's capabilities. The USC Medical School is collaborating with ISI to apply ideas from Steve in medical training systems; we are making plans to develop commercial products from this work as well. These efforts provide an important complement to the VET project; while VET allows us to address the important research issues in the area of pedagogical agents, these spin-off projects allow us to focus on transferring the technology we have already developed to practical applications.

4.2.9 ISI Future Development

We wish to show in the VET project that pedagogical agents can advise, guide, and collaborate with students, and that such agents can take special advantage of the capabilities of virtual environments to support rich interaction. To advise and guide effectively, agents should adapt their instruction to the students' knowledge and goals, just as is the case in other intelligent tutoring systems. However, since conventional ITS's do not support team interactions and are not designed for instruction in rich virtual environments, we should not expect simply to apply student modeling and student-adapted interaction techniques from other ITS's to Steve. In what follows we will describe how we believe pedagogical agents should adapt instruction to individual students in virtual environments, and our plans for providing such adaptation capabilities.

Virtual environment interfaces make it possible to continually track what students are doing: where they are, what they are looking at, what they are reaching for with their hands, etc. It is therefore possible for a VE-based agent to anticipate and form expectations about a student's actions before he or she performs those actions. In contrast, a conventional ITS can only react to actions that the student actually performs, or

perhaps statements of intended actions. We therefore believe that a VE-based agent should adapt instruction both according to what is known about the student and to the current situation. More important research remains to be done in finding ways of adapting instruction to the situation, and this should continue to be a priority for our work. At the same time, we need to determine what sort of student model is appropriate to the virtual environment context, and how it can complement the agent's knowledge of the situation in adapting instruction.

If a student is having difficulty completing a task, it could be for a number of reasons: the student may not know which object to manipulate, where the object to be manipulated is located, or may not know how to manipulate it. It should be possible for Steve to make inferences about the nature of the student's difficulty by observing where the student is relative to the object to be manipulated and what is in the student's field of view. For example, it is useful to know whether particular objects are within view and within reach. If objects to be manipulated are not visible, then the student may require help with navigation and orientation. If the objects are visible and reachable, then the student may have confusion about the task. Steve already receives and processes information about what objects are in the student's field of view; student position information is readily available as well. Thus we are in a position to experiment with using such information in providing feedback to the student, and indeed experimentation will be necessary to determine how valuable this information is. Such spatial reasoning can also help improve Steve's model of how well the student understands the situation. Currently Steve assumes that the student is able to see whenever an important event occurs in the environment; we can now avoid that assumption when we know that the objects involved are outside of the student's field of view.

In a similar vein, Steve could take advantage of the spatial environment to provide subtle guidance to the student. Steve already does this, for example, by looking at objects when they are moved. If Steve were also to glance at objects that require the student's attention, he could guide the student in the right direction without saying explicitly what to do. Such capabilities would also contribute to the sense that the agent is lifelike and shares the environment with the student.

A model of the student would be useful for interpreting student difficulties, but the model is likely to be different from what is typical for ITS's. For example, it would be useful for Steve to know ahead of time how familiar the students are with the working environment, and whether they can navigate through it. Most ITS student models do not address such concerns.

We hope to start addressing some of these issues in the current contract year. However, it is likely to take some time to give them a thorough analysis, so further investigation during the third contract year is essential.

4.2.9.1 Spatialized Sounds

Audio, particularly spatialized audio, is known to be very helpful in conveying a sense of realism and presence in virtual environments. We plan in the remainder of the current contract year to incorporate sound into the VET virtual environment.

The following types of sound processing are important to support in domains such as the gas turbine engine domain:

- the ability to generate ambient noise on a continual basis,
- selectively blocking sounds that are outside of a student's range of hearing, perhaps because they are in a different compartment aboard ship,
- attenuating the volume of sounds based on their distance from the hearer, and
- filtering sounds to give the impression of emanating from various locations in space.

We hope to provide each of these capabilities in the current contract year, time permitting.

Tools are available that help provide the above capabilities: multimedia players like XAnim can be used to provide continuous background sounds, and convolution software such as Audioworks can give sound an impression of three dimensions. However, some of these tools, particularly the convolution tools, require costly software licenses, and therefore are not generally available. We are planning to develop an audio client, similar to Trishtalk, that manages audio generation on a given student workstation, and provides differing degrees of audio support depending upon what audio generation software is available on the client machine.

4.2.9.2 *Affective Reasoning*

Motivation is a key ingredient in learning, and emotions play an important role in motivation. A pedagogical agent that appears to care about a student, and is sensitive to that student's emotions, will be more likely to motivate that student. Despite the apparent importance of such affective reasoning in tutors, this issue has received no attention in the intelligent tutoring systems community.

We have begun collaborating with Prof. Clark Elliott of DePaul University on this topic. Prof. Elliott did his dissertation work on affective reasoning, and he is interested in using a pedagogical agent such as Steve as a vehicle for exploring the role of affective reasoning in tutoring systems. Our work in this area is still preliminary, but we have laid out our basic objectives, along with some initial ideas, in a recent workshop paper [Elliott et al. 97].

4.3 Virtual Environment Interaction

Research and development for the Vista Viewer concerned two major areas: supporting common model formats to facilitate 3D authoring with best-of-breed commercial software, and providing a uniform framework for immersed interaction using standard VRML sensors. According to Mark Mine, Fred Brooks, and Carlo Sequin in their SIGGRAPH 97 paper *Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction* several less obvious factors have hampered development of "real-world" virtual environment applications:

1) *The precise manipulation of virtual objects is hard.* Although immersion, head-tracked view specification, and six degree-of-freedom (DoF) hand tracking facilitate the coarse manipulation of virtual objects, the precise manipulation of virtual objects is complicated by:

Lack of haptic feedback: Humans depend on haptic feedback and physical constraints for precise interaction in the real world; the lack of physical work-surfaces to align against and rest on limits precision and exacerbates fatigue. Though there is considerable ongoing research in the area of active haptic feedback [Durlach 95], general-purpose haptic feedback devices that do not restrict the mobility of the user are not yet practical or available.

Limited input information: Most virtual-environment systems accept position and orientation (pose) data on the user's head and (if lucky) two hands. One also typically has a button or glove to provide signal/event information. This suffices for specifying simple 6 DoF motion and placement. In the real world, we do this and much more:
a) Object modification, usually with tools. b) Directing the cooperation of helping hands, by spoken commands ("Put that there"). c) Measuring. d) Annotating objects with text.
Today in most VR systems: a) Tool selection is difficult. b) Voice command technology is marginal. c) Measuring tools are rarely available. d) Alphanumeric input is difficult.

Limited precision: The lack of haptic and acoustic feed-back, inaccurate tracking systems, and whole-hand input typical of current VR systems restricts users to the coarse manipulation of virtual objects. Fine-grained manipulations are extremely difficult using this "boxing glove" style interface. Shumin Zhai of the University of Toronto, for example, has demonstrated that users' task completion times were slower in a 3D docking task when using a 3D input device which excluded the use of the fingers (vs. a similar device that utilized the fingers) .

2) *Virtual environments lack a unifying framework for interaction,* such as the desktop metaphor used in conventional through-the-window computer applications. Without haptics, neither real-world nor desktop computer interaction metaphors are adequate in a virtual environment. Knowledge on how to manipulate objects or controls can no longer be "stored in the world", with the physical constraints of the devices giving the user clues as to their use (e.g. a dial can only be rotated about its axis).

The desktop metaphor further breaks down when the user is inside the user interface. Interface controls and displays must move with the user as he moves through the environment and be made easy to locate. The differences between working in a conventional computer environment and working immersed are analogous to the differences between a craftsman at a work-bench and one moving about a worksite wearing a toolbelt. His toolbelt had better be large and filled with powerful tools.

With our immersed adaptation of VRML 2.0, **there is a unifying framework for immersed interaction.** In VRML 2.0, knowledge of how to manipulate objects or controls is stored in the world. Physical constraints for devices give the user clues about their use - numerous doors, dials, throttle controls, etc. in our VRML 2.0 worlds can be manipulated as they normally are. VRML 2.0 models for training can be authored using commercial tools, such as Cosmo Worlds, to set up animations, constraints on motion of the objects, and scripts for object behavior. The authoring can be built and tested on the desktop, then used immersively.

4.3.1 VRML 2.0

The foundation for accomplishing operations and maintenance instruction in a virtual environment is interaction with objects in the virtual environment. Vista support for VRML 2.0 was developed because of VRML's potential for training applications. It is becoming a widespread 3D exchange format, good-quality scene authoring software is available, and the method for dynamic update of 3D scenes over the Internet is built into the standard.

Using standard VRML 2.0, we have taken steps to realize object interaction in an immersed setting. Many types of object interaction can be represented by authoring scenes using VRML 2.0 sensors, and then routing this sensor output to relevant nodes in the 3D scene graph. In this way, VRML 2.0 can be used to accomplish human-computer interaction for training in a fairly flexible and domain independent manner.

During the first year of the VET contract, a file loader for VRML 1.0, and an initial VRML 2.0 file loader were developed. Work during the second year of development concentrated on support for a greatly expanded set of VRML nodes as well as techniques for object manipulation and user interaction consistent with the VRML 2.0 standard. Vista supports most of the node types specified in the VRML 2.0 standard. Texture support was integrated into the Vista Viewer, as well as support for coordinate and scalar interpolators. The VRML capabilities were moved into a stand-alone library with the goal of making the library available for public use by the VRML researchers.

A key feature of VRML models is the ability to embed sensors that will trigger preauthored behavior when activated by a user. To use this capability for immersed equipment manipulations, Vista Viewer was first extended to support script nodes, which are used for programming behavior in a VRML scene. Script nodes receive events, contain a program module that performs some computation, and effect change somewhere else in the scene graph. Vista Viewer supports VRMLscript, a subset of JavaScript. Development is underway to also support Java as a scripting language in Vista Viewer, by using the actual Java Virtual Machine and just-in-time compilation. Implementation of Vista Viewer script node support allowed Lockheed Martin to research novel techniques for manipulating VRML sensors by both direct and projected interaction. This work was presented at the VRML 1997 conference in Monterey, CA and appears to be the first reported successful effort for immersed VRML 2.0 interactions.

Other work in VRML relevant to authoring training includes a converter from VRML 2.0 to Open Inventor, which is widely supported by different modelers. The conversion involves stripping out the VRML 2.0 behavior and correctly storing the geometry in the Open Inventor format. To our knowledge this is the only available VRML 2.0 to Open Inventor converter.

4.3.2 HCI Research

User acceptance is key to the success of any application. Much effort, therefore, was spent in identifying and addressing user interaction issues. Research covered both interaction device support and immersive interactions techniques.

4.3.2.1 Device Support

During the first year of development, support was provided for a 3D mouse. This allowed users to manipulate within the environment, but the required point and click operation did not naturally map to real world gestures. Midway through the second year of development, support was added to the Vista Viewer for Pinch Gloves and techniques developed to facilitate navigation. Object manipulation using the index finger sensor of the glove provides a more natural extension of real world selection. Other fingertip sensors are used to provide a means of flying and picking. Immersed representations include hand and wand.

4.3.2.2 Immersed Interactions

Immersed interactions present issues not associated with flat screen interactions. Using a head-mount display to view a scene allows rotation of the user's head for viewing different parts of the scene with concurrent movement of the user's hands for object interaction, regardless of whether the hands and/or object are in the users view. This flexibility can create problems; for example the user may select objects not in view or perhaps trigger an event that cannot be viewed. Because objects in flat screen mode must be in view to be selected with a mouse, these problems do not occur. To deal with immersed out-of-sight selection problems, our VRML 2.0 implementation does not generate any events if the intersection point with VRML geometry is outside the field of view. Also, manipulations can be restricted to a direct mode; i.e., the user would be required to navigate with arms reach of the object to allow direct handling. An advantage of this mode for equipment maintenance and operations is the mirroring of real world behaviors, as opposed to requiring users to make a cognitive mapping form a moving mouse to the actual motions of a 3D object.

An immersive environment provides greater freedom to explore, and the user expects to be able to look around. To determine what the user is viewing, Vista makes use of the VRML 2.0 VisibilitySensor. This allows other Training Studio components to query whether the student can view a lesson sequence or event. This technique can prevent events or instruction from occurring when the student is unable to view the activity.

There are times when the author needs to control the student's viewpoint to ensure they see a particular piece of equipment or area. Both taking and releasing control causes a sudden snap between the user's view and the controlled view. In a flat screen mode, this

sudden change in viewpoint is okay because the user is not immersed. In the immersive environment, however, a sudden change can adversely affect the user; they can become disoriented. Without considering the current orientation, when control is released to or taken from the viewer, the viewpoint could suddenly snap from the end of the controlled path. An innovative technique was developed to provide a slow transition between the controlled view to the actual viewer orientation. This not only removes the disconcerting snap to a different view, but also provides context of the previous view in relation to the actual orientation.

4.3.3 Vista Viewer Optimization

Much effort was spent to optimize the Vista Viewer performance. As new and complex capabilities (such as large application domain scenes, immersed interaction, VRML scripting, etc.) were developed, Vista's performance was affected. The system was examined for performance bottlenecks and resulting issues were resolved. Other actions were taken to increase speed and reduce memory requirements, such as creating geometry nodes in a VRML scene directly through Performer, instead of indirectly using Open Inventor, which came at a significantly higher overhead. The Vista implementation of VRML 2.0 allowed the use of ProximitySensors for reducing the complexity of the immersed scene, by switching off the display of rooms when participants leave them. In VRML 2.0, in addition to the geometry being drawn, events are pressed in response to user action. Sensors and script nodes which add to the behavior to the scene can be switched off, depending on where the user is in the scene. Events for behavior not in the user's proximity are not processed.

Optimized versions of the most commonly used nodes are also now used, determined by whether the node is named and therefore can receive events. Another change implemented to improve performance was the sharing of similar materials across models. The Vista technique of handling inline nodes was developed to reduce hits on performance by adapting the common convention of delayed loading when the Inline nodes appear as the children of LOD nodes. This allows for patching worlds together, and reducing the wait time involved before one can view and interact with VRML scenes.

During the second development year, a software port of the Vista Viewer from SGI platform to the PC platform was examined. Our group beta-tested Optimizer (Silicon Graphics Inc.) software, and development of an Optimizer version of the Vista Viewer was initiated. Optimizer not only provides greater graphics capabilities, such as occlusion culling, for efficiently rendering large CAD scenes such as ships, but will also be supported on the PC Windows platform. The approach is to first develop and modify an Optimizer based Vista Viewer to run satisfactorily on the SGI platform, while simultaneously transitioning to the PC after each step.

4.3.4 TScript extensions

Communications between the components of the Training Studio is accomplished via a communications bus over which messages using the Lockheed Martin developed TScript protocol is used. The TScript protocol consists of groups of messages which can change visual scenes and convey changes in the distributed state of the simulation. A core of TScript messages was developed and implemented during the first year of the contract. During the second year, extensions were made as new requirements were identified to facilitate the component functional developments.

4.4 Team Training and Task Domain

Virtual reality opens up exciting possibilities for team training. Multiple students, possibly at different locations, can cohabit the same virtual world, practicing team tasks (or even adversarial tasks) together. During the first contract year of the VET project, we focused on training individual students. During this second year, our focus has shifted to supporting team training.

The team training effort required a different emphasis from each of the three Training Studio components. In the Vista Viewer, research and development issues involved

representations and models necessary to allow authoring and presentation of instruction in the virtual environment for a team. Issues surrounding Steve development addressed various roles of the Agent as mentor, guide, and team member. At the end of the first year, multiple Staves could inhabit the environment. During the second year of the contract, interactions between the agents cohabiting the virtual environment were addressed.

A formalized approach for developing team training capabilities into the Training Studio is found in "Team Training Development and Approach", delivered to ONR with the 6th Quarterly Report in April 1997. The report, which resulted from a January 1997 VET team meeting, lists the required capabilities for team training as well as a plan for development these capabilities within each component. It describes the representational changes for group tasks needed in Steve, the representations need to support consistent visual update in several Vista Viewer displays, and changes to simulation and instruction representation needed in VIVIDS. A series of integration meetings were held at Palo Alto at monthly intervals to ensure progress towards the team training goals.

Throughout the VET project, our work has been guided by Navy procedures whose training could benefit from virtual reality. Although the Training Studio is intended as a set of domain independent tools that can be easily authored to support training in various domains, Navy procedures provide a valuable testbed for our research and development. As outlined to the Society of Naval Engineers during the Intelligent Ships Symposium, shipboard training, specifically for the Arleigh Burke class ship, has been addressed in detail for demonstration of the Training Studio [Stiles et al. 96]. During the first contract year, we focused on operation and maintenance of high pressure air compressors, which are part of the gas turbine engines that propel surface ships. During this second contract year, we have broadened our focus to include other components of gas turbine engines, and we have particularly focused on team tasks.

Early in this second contract year, we traveled to the Naval Training Facility in Great Lakes. During this trip, the area of casualty control tasks was identified as a potential team training domain. Besides gaining overall knowledge of the GTE domain overall, each of the three development teams focused on obtaining specific information with respect to the individual system components. From ISI's perspective, the main benefits of the trip were gathering sound clips for possible use as 3D audio, learning about equipment and procedures (especially team tasks) for the gas turbine engine, and learning about instructional methods and types of interaction between instructors and students. The Lockheed Martin group took photographs and videos of the control console and engine room layout, as well as specific pieces of equipment, and gathered diagrams and technical information for modeling purposes. The Central Control Station activity during a Casualty Control drill was also captured on videotape. For VIVIDS development, the BTL team took photographs and gathered reference materials of control panels instrumentation and behavior. To both drive and validate the developing team training capabilities, several tasks from the Gas Turbine Engine domain were selected.

Building on the information gathered during that trip, ISI formalized some of the procedural knowledge required to operate and maintain shipboard gas turbine engines. We have focused primarily on casualty control procedures, especially those involving teamwork. Lockheed Martin developed Vista 3D models of the EPCC and PACC Control Consoles in VRML 2.0 format, as well as filling out the existing engine room CAD model with pertinent equipment representations. The BTL team members integrated their knowledge into a VIVIDS simulation model of the EPCC and PACC consoles. While the EPCC and PACC consoles were being developed by Lockheed Martin and BTL, ISI added procedural knowledge to Steve for operating these consoles. Although only a few short procedures have been implemented so far, the graphical and simulation models are reaching a state where many more procedures can be implemented.

We expect this to be an important part of our work during the remainder of this second contract year.

Two other site visits were made during the year. A tour of Naval Surface Warfare Center (NSWC), Carderock Division's Land Based Engineering Station in Philadelphia provided a view of the engine space and actual equipment onboard the DDG-51 as well as access to technical experts. A second trip to Great Lakes captured video footage in the engine space during a Casualty Control drill in the CG-47. To supplement information and resources gathered during the three site visits, Lockheed Martin investigated other sources of technical material. Our ongoing relationship with GLNTC has provided continuing access to technical manuals and information as the need is identified. A CD ROM containing the U.S. Navy Engineering Operational Sequencing System (EOSS), which includes the standard and mandatory ship procedures, was obtained from NAVSES with assistance from the ONR. MicroGraphx Designer files of the gas turbine engine were requested and received from the GLNTC Multi Media Support Group. Other agencies contacted and still under investigation are: General Electric (LM2500 CAD model), Lockheed Martin AEGIS program (VRML models of the DDG-51), and Lockheed Martin in Orlando (controller system models).

4.5 VE Training Evaluation

In keeping with requirements outlined in the our proposal, the Lockheed Martin VET team initiated evaluation efforts during the second year of development. This evaluation study examines the differential effects of conventional computer-based instruction versus virtual environment-based instruction on the retention of procedural knowledge. The focus and general parameters of the evaluation was established as a joint effort between Lockheed Martin, USC/BTL, USC/ISI and USAF Armstrong Lab partners. The evaluation plans were designed by the HRCT of the U.S. Air Force's Armstrong Laboratory at Brooks Air Force Base in Texas. Lockheed Martin and the two USC groups provided technical support to the evaluation team; however, implementation is under the AL-HRCT group. A pilot study was conducted in July 1997 to validate instructional designs, media, experimental procedures and instruments for use in a main experiment, scheduled for October 1997. At the time of this report, the USAF group is in process of running the pilot study. Results of this initial study and plans for conducting the main experiment will be presented to the Office of Naval Research in September 1997.

5. Conclusions

The technological significance of our development for the past year lies in four areas that apply to training: authoring VE simulations, virtual environment interaction, pedagogical agents, and team training.

The authoring of VE simulations for training has been improved on several fronts affecting authoring 3D models, training simulations, and instruction. The quality of virtual environment interaction is very important for training in equipment operations and maintenance. By adapting VRML 2.0 for immersive use, we have realized a unifying framework for interaction that allows storing knowledge of how to manipulate objects and controls. The pedagogical agents we have developed are a radically new approach to intelligent tutoring. They are able to help students in ways that conventional intelligent tutors cannot. Agents like Steve that can share the environment with the students can demonstrate to them how to perform tasks if appropriate. Team training in a virtual environment is significant because many of the tasks suited to instruction in a virtual environment, such as those involving dangerous situations or expensive equipment, also involve several people. Our development of participant (team-member) specific capabilities in Vista, VIVIDS, and Steve allows us to deliver team training in a virtual environment.

For future work, much more remains to be done to support authoring team training (as opposed to authoring individual training) as well as supporting team interactions, both physical and in terms of expectations.

Authoring in general can be improved by a serious productization effort of each of the software components, as well as further research into authoring complex tasks by example. Evaluation of different methods of instruction, such as instruction in a 3D setting with or without Steve, should be carried out in the future.

6. References

[Badler et al. 93] Badler, N., Phillips, C., and Webber, B., 1993. *Simulated Agents and Virtual Humans*, Oxford University Press.

[Cassell et al. 94] Cassell, J., Steedman, M., Badler, N., Pelachaud, C., Stone, M., Douville, B., Prevost, S., and Achorn, B., 1994. Modeling the interaction between speech and gesture, *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, Lawrence Erlbaum Associates, Hillsdale, NJ.

[Durlach 95] Durlach, N. I. and A. S. Mavor, ed., 1995.*Virtual Reality: Scientific and Technological Challenges*, pp. 261-267, National Academy Press.

[Elliot et al. 97] Elliott, C., Rickel, J., and Lester, J.C., 1997. Integrating Affective Computing into Animated Tutoring Agents. *In IJCAI Workshop on Animated Interface Agents*, Nagoya, Japan.

[Grosz and Sidner 86] Grosz, B.J. and Sidner, C.L., 1986. Attention, Intentions, and the Structure of Discourse, *Computational Linguistics* 12 (3), pp. 175—204.

[Johnson et al forthcoming] Johnson, W.L., Rickel, J., Stiles R., and Munro, A. Integrating Pedagogical Agents into Virtual Environments. *Presence*, forthcoming.

[Thorisson and Cassell 97] Thorisson, K.R. and Cassell, J., 1997. Communicative Feedback in Human-Humanoid Dialogue, *IJCAI Workshop on Animated Interface Agents*

[Laird et al. 87] Laird, J.E., Newell, A., and Rosenbloom, P.S., 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33(1), pp. 1-64.

[Maybury 93] Maybury, M.T., 1993. *Intelligent Multimedia Interfaces*, AAAI Press, Menlo Park, CA.

[McAllester and Rosenblitt 91] McAllester, D. and Rosenblitt, D., 1991. Systematic Nonlinear Planning, Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91), pp. 634-639, AAAI Press, Menlo Park, CA.

[Mine 97] Mine, Mark R., Brooks, Frederick P., Sequin, Carlo H., *Moving Objects In Space: Exploiting Proprioception In Virtual Environment Interaction*, Proc. ACM Press, SIGGRAPH 97, Los Angeles, CA, August 1997.

[Newell 90] Newell, A., 1990. *Unified Theories of Cognition*, Harvard University Press, Cambridge, MA.

[Pelachaud et al. 96] Pelachaud, C., Badler, N.I., and Steedman, M., 1996. Generating Facial Expressions for Speech, *Cognitive Science* 20 (1).

[Rickel and Johnson 97a] Rickel, J. and Johnson, W.L., 1997. Integrating Pedagogical Capabilities in a Virtual Environment Agent, *Proceedings of the First International Conference on Autonomous Agents*, ACM Press.

[Rickel and Johnson 97b] Rickel, J. and Johnson, W.L., 1997. Intelligent Tutoring in Virtual Reality: A Preliminary Report, Proc. Eighth World Conference on AI in Education.

[Russel and Norvig, 1995] Russell, S., and Norvig, P., 1995. *Artificial Intelligence: A Modern Approach.* Prentice Hall, Englewood Cliffs, NJ.

[Stiles et al. 97] Stiles, R. Tewari, S., Mehta, M. *Adapting VRML 2.0 for Immersive Use.*, ACM Press, Proc. VRML97, Second Symposium on the Virtual Reality Modeling Language, Monterey CA, Feb 1997.

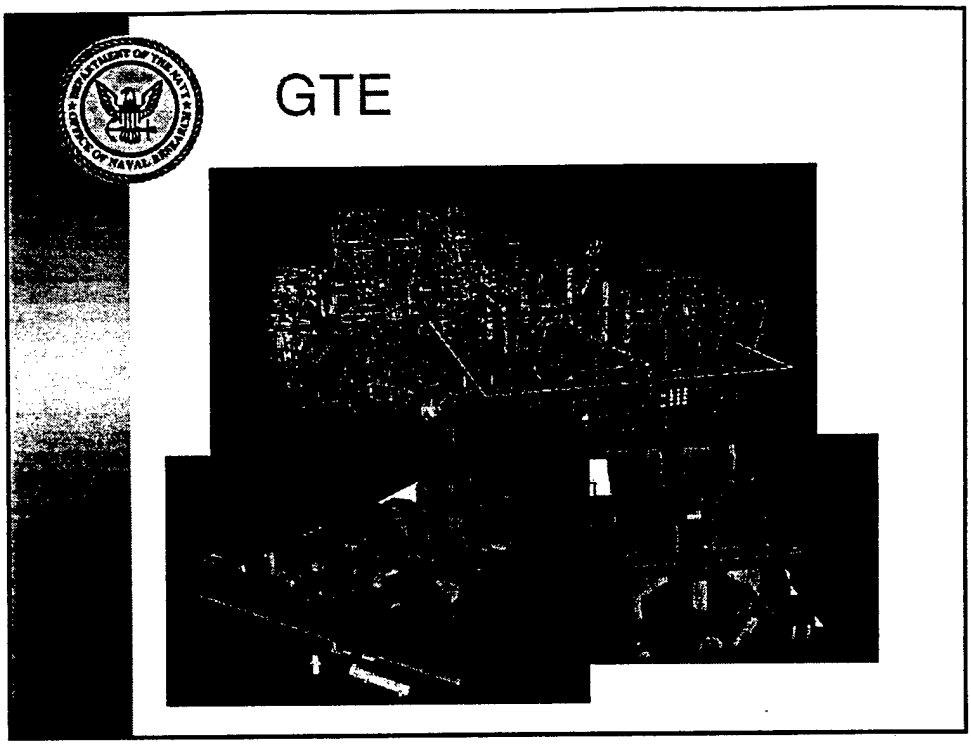
[Stiles et al. 96] Stiles, R., McCarthy, L., Johnson, L., Rickel J., Munro A., and Pizzini, Q. *Virtual Environments for Shipboard Training.* In Proc., Intelligent Ships Symposium II, The American Society of Naval Engineers, Philadelphia, PA, November 1996.

7. Symbols, Abbreviations, and Acronyms

AAAI	American Association for Artificial Intelligence
BTL	Behavioral Technologies Laboratories, located in Redondo Beach, CA, a performing organization in the Lockheed Martin VET effort, a laboratory of the University of Southern California.
DARPA	Defense Advanced Research Projects Agency
DIS	Distributed Interactive Simulation, a real-time distributed message protocol used in training and operational simulations developed by ARPA and now an International Standards Organization standard.
FBM	Fleet Ballistic Missiles program, a Lockheed Martin program funded by the U.S. Navy for the production of submarine-launched ballistic nuclear missiles
GTE	Gas Turbine Engine - similar to jet engine, which drives propulsion of a Navy Ship. In our case we are usually referring to the LM2500 Gas Turbine Engine on USS Arleigh Burke (DDG-51) ships.
HPAC	High Pressure Air Compressor, an oil-free air compressing system prevalent on many navy vessels, which prepares compressed air for gas turbine engines.
ICAI	Intelligent Computer Aided Instruction, a method of instruction whereby an intelligent model of a student's understanding is used to guide a student during instruction using a computer.
IPEM	Integrated Planning, Execution and Monitoring architecture for coordinating different planning strategies as required for SOAR activities.
IRL	Institute for Research on Learning
ISI	Information Sciences Institute in Marina del Rey, CA, a performing organization in the Lockheed Martin VET effort, affiliated with the University of Southern California in Los Angeles, CA.
JPL	Jet Propulsion Laboratories, a National Laboratory affiliated with the National Aeronautics and Space Administration.
MCO	Multi-Channel Option for Silicon Graphics Onyx Workstations, a necessary option to provide separate video channels used in immersive virtual environment displays.
ONR	Office of Naval Research, the funding agency for the VET effort.
RIDES	Rapid Instructional Development for Educational Simulation
SIGART	Special Interest Group on Artificial Intelligence
SGI	Silicon Graphics Incorporated, a workstation company whose whole culture centers around fast 3D graphics.
SOAR	A platform independent, cognitive architecture based on a production system which seeks to address those capabilities required of a general intelligent agent.
STEVE	SOAR Training Expert for Virtual Environments
Tcl/Tk	A windowing interface toolkit assembled around a UNIX-shell like interpreter originally developed at UC Berkeley.
TScript	Training Script message protocol for virtual environments
URL	Uniform resource locator, a tag that indicates a media format and location on the Internet as part of the World Wide Web.
USC	The University of Southern California.
VE	Virtual Environment, a 3D visual display and accompanying simulation which represent some aspect of an environment. Expanded forms of VE also address other senses such as audio, touch, etc.
VET	Virtual Environments for Training, a Defense Department focused research initiative concerned with applying virtual environment technology to training
VR	Virtual Reality *see Virtual Environment*
VRIDES	Virtual Rapid Instructional Development for Educational Simulation. A special version of the RIDES program for use in developing simulations and tutorials that collaborate with Vista Viewer and Soar to deliver training in virtual environments.
VIVIDS	*See VRIDES above*
VRML	Virtual Reality Modeling Language, an analog to HTML used for documents, but focused on 3D objects and scenes for the World Wide Web.
WWW	World-Wide Web, a system incorporating the HTTP message protocol and the HTML document description language that allows global hypertext over the Internet.

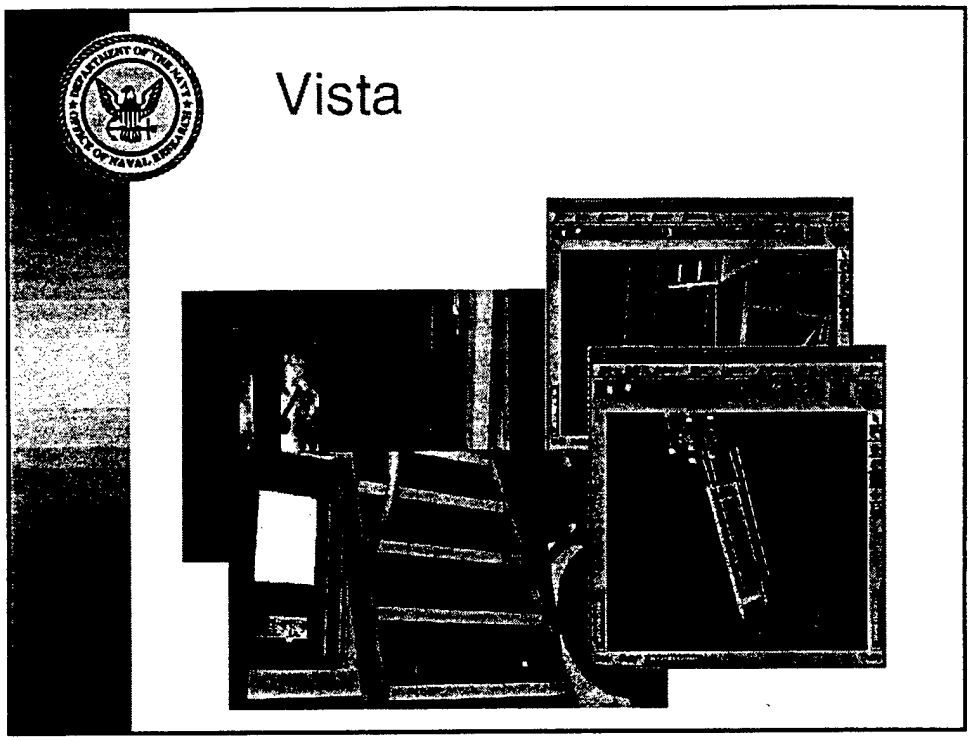
APPENDIX

Viewgraph Annotations



Early in this second contract year, we traveled to the Naval Training Center in Great Lakes. During this trip, the area of casualty control tasks was identified as a potential team training domain. Besides gaining knowledge of the GTE domain overall, each of the three development teams focused on obtaining specific information with respect to the individual system components. From ISI's perspective, the main benefits of the trip were gathering sound clips for possible use as 3D audio, learning about equipment and procedures (especially team tasks) for the gas turbine engine, and learning about instructional methods and types of interaction between instructors and students. The Lockheed Martin group took photographs and videos of the control console and engine room layout, as well as specific pieces of equipment, and gathered diagrams and technical information for modeling purposes. The Central Control Station activity during a Casualty Control drill was also captured on videotape. For VIVIDS development, the BTL team took photographs and gathered reference materials of control panels instrumentation and behavior. To both drive and validate the developing team training capabilities, several tasks from the Gas Turbine Engine domain were selected.

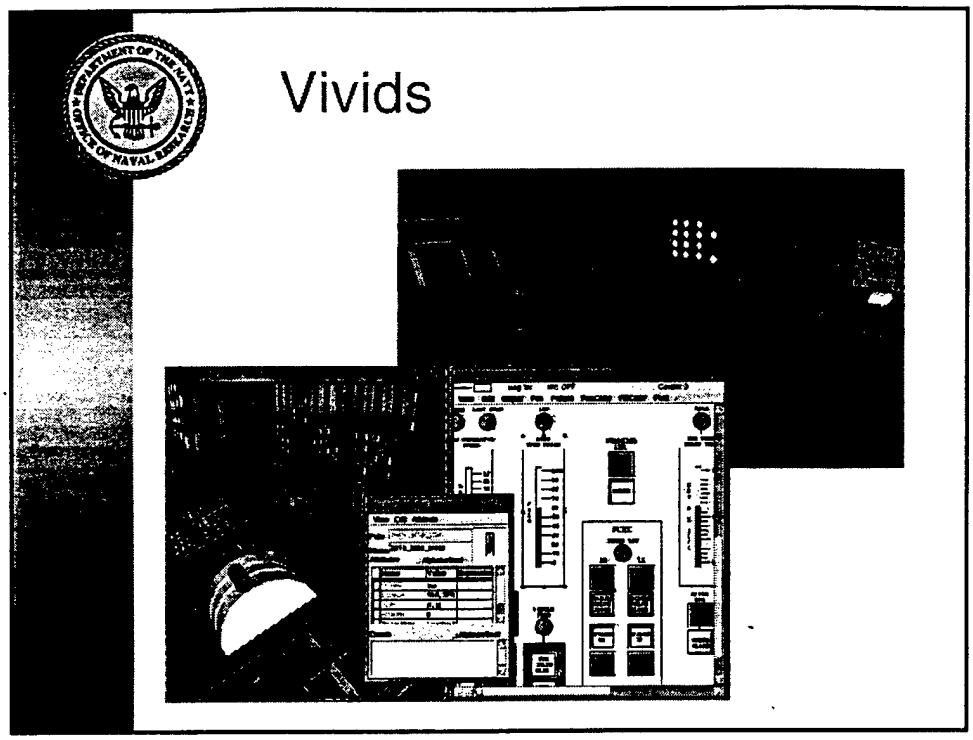
Team training in a virtual environment is significant because many of the tasks suited to instruction in a virtual environment, such as those involving dangerous situations or expensive equipment, also involve several people. Damage control onboard a ship and casualty control to maintain equipment performance while underway, are examples where virtual environments which support team training are useful. To accomplish team training in a virtual environment, we have structured our shared graphics environment (Vista) to support participant-specific changes to the world state, and participant-specific queries for world state, as well as maintenance of shared state. We have modified VIVIDS to allow presenting event sequences, and participant-specific opportunistic instruction that is sensitive to the team task progress. We have greatly augmented our pedagogical agent's task representations to support team tasks, and team roles for agents, whether monitoring a live student's action, or acting as missing team member. We have also explored representing the hand-off and task expectations between team members. Our development of participant (team-member) specific capabilities in Vista, VIVIDS, and Steve allows us to deliver team training in a virtual environment.



Shown here, SGI Cosmo Worlds Authoring tool is used to set up object behavior in VRML 2.0. This and other commercial authoring tools can be used with Vista to realize virtual environments for training.

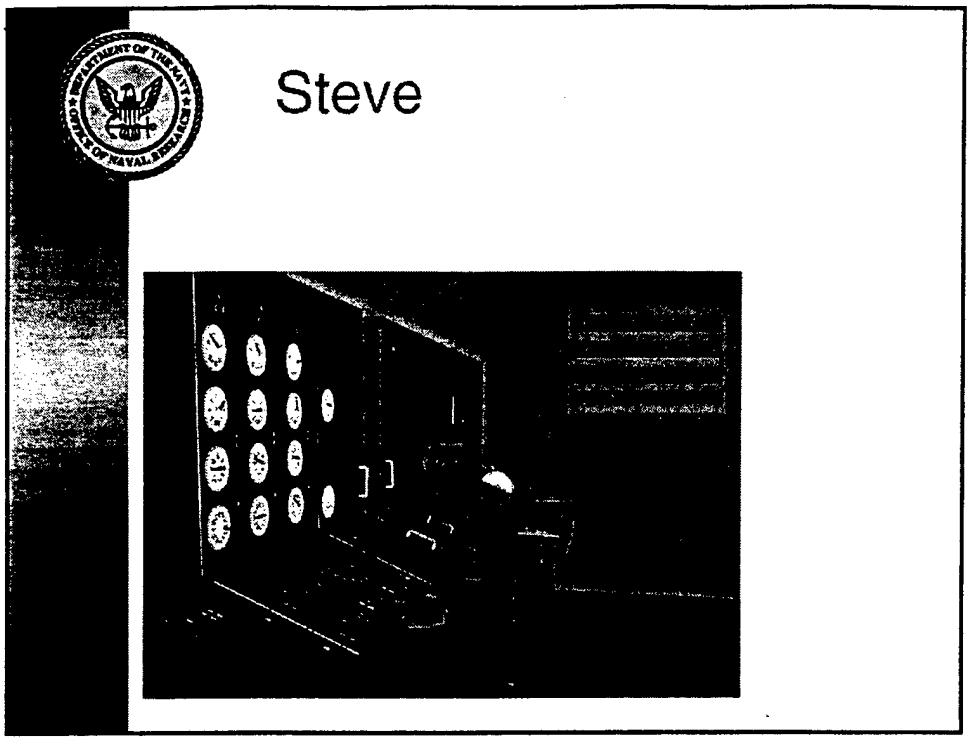
The foundation for accomplishing operations and maintenance instruction in a virtual environment is interaction with objects in the virtual environment. Vista support for VRML 2.0 was developed because of VRML's potential for training applications. It is becoming a widespread 3D exchange format, good-quality scene authoring software is available, and the method for dynamic update of 3D scenes over the Internet is built into the standard.

Using standard VRML 2.0, we have taken steps to realize object interaction in an immersed setting. Many types of object interaction can be represented by authoring scenes using VRML 2.0 sensors, and then routing this sensor output to relevant nodes in the 3D scene graph. In this way, VRML 2.0 can be used to accomplish human-computer interaction for training in a fairly flexible and domain independent manner.



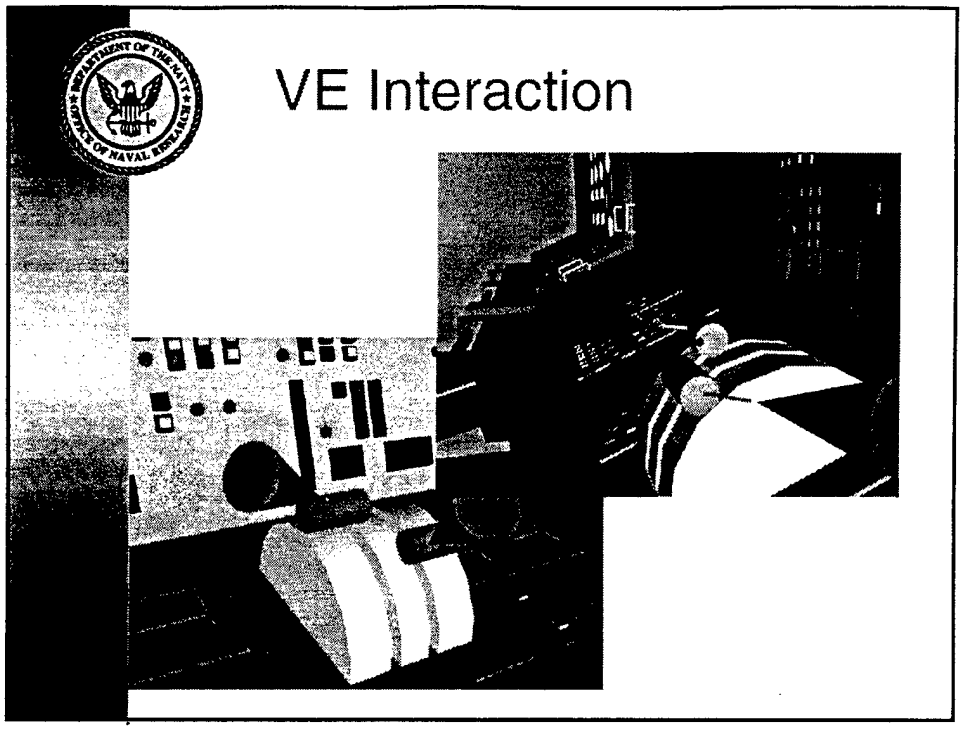
The authoring of VE simulations for training has been improved on several fronts. We have improved simulation authoring by more closely tying VIVIDS to the virtual environment, adapting instructional primitives for VE, providing an instructional control interface, and testing VIVIDS on large simulations. Authoring by example for Steve was also improved, to re-use previous tasks already authored, and to reset VIVIDS state when generating examples. Support for immersed use of VRML 2.0 was dramatically improved, allowing authoring of 3D models using commercial, high quality VRML authoring tools. By improving the authoring capabilities of the three main Training Studio components, it is more practical to consider developing a 3D, immersed training simulation for a given domain.

The VIVIDS authoring system constitutes the first system for authoring (as opposed to programming) robust complex interactive simulations for virtual environments. Furthermore, these authored simulations have features that support the automatic construction of certain types of structured tutorials. The combination of productive simulation authoring with efficient tutorial development is designed to make feasible the application of virtual environment technologies to a very wide range of technical training requirements.



Pedagogical agents are a radically new approach to intelligent tutoring. They are able to help students in ways that conventional intelligent tutors cannot. Agents like Steve that can share the environment with the students, can demonstrate to them how to perform tasks if appropriate. Conventional intelligent tutors have a limited ability to demonstrate and mainly provide commentary on actions that the students make. Pedagogical agents can be easily used to support team training, since they can participate with the students in team activities. They can make use of non-verbal cues and gestures to provide subtle feedback to students, and to interact with students in a more natural, human-like fashion. There is evidence that giving agents an interactive human-like appearance improves student motivation. Finally, agents can be used to more closely emulate the way training is performed in the real world with human tutors. In particular, we are using agents to simulate how the Navy currently trains crews in surface ships, where teams of trainers interact one-on-one with members of trainee crews in team exercises.

Pedagogical agent technology relies heavily on, and naturally supports, virtual environment (VE) technology. It makes use of the position and orientation data that VE's collect to track students more closely and provide a wider range of feedback. It permits natural interaction with the intelligent tutoring system, just as VE technology permits natural interaction with the simulated environment.



The quality of virtual environment interaction is very important for training in equipment operations and maintenance. Tasks to operate equipment always involve manipulating controls, and maintenance almost always involves assembling, disassembling, and replacing equipment. Researchers in the computer graphics field, such as Mark Mine, Fred Brooks and Carlo Sequin, state that virtual environments lack a unifying framework for interaction, and elaborate that knowledge on how to manipulate objects or controls cannot be "stored in the world". [Mine 97] By adapting VRML 2.0 for immersive use, we have realized a unifying framework for interaction, that allows storing knowledge of how to manipulate objects and controls. To accomplish training, we had to go further and provide support for sensing querying the state of the world by other training software components such as VIVIDS and Steve. This is significant because it takes us beyond selecting objects in the world, and allows authoring the manipulation and use of complex objects for training.

ONR ANNUAL PRODUCTIVITY REPORT,
01 OCT 1996 TO 30 SEP 1997
(Revision 1)

1. **Principal Investigator Name:** Randy Stiles
(with Dr. Allen Munro, USC/BTL and Dr. Lewis Johnson, USC/ISI)

2. **Institution:** Lockheed Martin Advanced Technology Center

3. **Full Address:**
3251 Hanover Street
O/H1-42, B/255
Palo Alto, CA 94304-1191

4. **Current Phone:** (650)354-5256

5. **Current Fax:** (650)424-3425

6. **E-mail:** stiles@ict.atc.lmco.com

7. **Project Title:** Virtual Environments for Training (AFOSR Focused Research Initiative)

8. **Do you post web pages that provide reports of descriptions of your research?**
Yes XX **No**_____ **URL:** http://vet.parl.com/~vet/
http://btl.usc.edu/VET/
http://www.isi.edu/isd/VET/vet.html

9. **For the current fiscal year, please provide:**

a. **Number of ONR supported:**

i. **Papers published in refereed journals:** ____

ii. **Papers accepted for public. in refereed journals:** 1

iii. **Papers or reports in non-refereed journals:** ____

iv. **Books or book chapters published:** ____

v. **Books or book chapters in press:** ____

b. **Number of ONR supported patents/inventions filed** NA **or granted** NA, **with patent numbers:**

c. **Trainee Data:**

	Total	Female	Male	Minority	Non US Citizen
i. No.Grad Students	3	0	3	0	1
ii. No.Postdocs					
iii. No.Undergrads	1	0	1	0	0

d. Number, cost and description of equipment items costing more than $1000 that were purchased on your ONR grant.

USC/BTL Equipment

$14,796 Silicon Graphics O2 Webforce Workstation

USC/ISI Equipment/Software

$ 2,000 FakeSpace Pinch Glove System

$ 4,382 Paradigm Simulation AudioWorks2 3D sound software for SGI Indigo Impact

e. Awards/Honors to PI and/or members of PI's research group (please describe).

f. Brief description of all transitions (or intended transitions) of your ideas or techniques to industry, to military laboratories or to military application. Include the name of the organization to which the work was transitioned, and the name of a point of contact. (Note: Transitions are of increasing importance in the current applications-oriented climate of DoD research.

(1) The VET project is working with the US Navy ONR to use NAVSEA CAD data for the Arleigh Burke in developing Gas Turbine Engine System operations and maintenance and training in a virtual environment. POC is Capt. Tim Steele.

(2) The VET project is working with the US Navy Strategic Systems Program to use their virtual prototype data for immersed virtual environment training for D5 ballistic missile operations and maintenance. POC is Mr. Ben Wosoogh.

(3) The VET project is collaborating for testing and practical evaluation of the VET software for use in Air Force domains with USAF Armstrong Labs Cognitive Training group (HRCT), Brooks AFB, San Antonio TX. POC Mr. James Fleming.

(4) Lockheed Martin has been transitioning lessons learned and insights from immersive virtual environment training systems to the Virtual Reality Modeling Language (VRML) standardization efforts for VRML 1.0, VRML 2.x, and VRML 3.0. VRML is an analog to HTML for the world-wide-web, but covers 3D model exchange and updating using the world-wide-web and is enjoying wide industry acceptance. No formal POC, archive of participation in www-vrml standards mailing list with companies such as Sony, SGI, MicroSoft, Apple, etc is at http://vag.vrml.org/

(5) BTL is transitioning many of the ideas resolved and derived in the VET project to the tutor-authoring system it is building under the VIVIDS project with funding from USAF Armstrong Labs Human Resources Cognitive Training group (HRCT), Brooks AFB, San Antonio TX. POC Mr. James Fleming. Two other systems were transitioned from BTL during this year: AVATAR - Advanced Virtual Adaptive Tutor for Air Traffic Control Readiness, sponsored by Armstrong Laboratory, USAF, POC: Mr. James Fleming, and a system for Quaker State Corporation, POC: Joseph Carlisle

(6) ISI is working with the DARPA Computer Assisted Training and Education Initiative (CAETI) on the PROBES project to transition dynamic virtual world technology to military training exercise management No Formal POC, URL with contact info is http://www.isi.edu/isd/PROBES/probes.html

(7) The Air Force Armstrong Laboratory has provided funding to support the commercialization of Steve's capabilities. POC is James Fleming.

(8) The USC Medical School is collaborating with ISI to apply ideas from Steve in medical training systems. Plans to develop commercial products from this work are underway.

g. Attach list of papers and other publications with full citation.

Journal papers:

W.L. Johnson, J. Rickel, R. Stiles, and A. Munro. Integrating Pedagogical Agents into Virtual Environments. To appear in the journal Presence.

Conference papers:

J. Rickel and W.L. Johnson. *Intelligent Tutoring in Virtual Reality: A Preliminary Report*. In Proc. Eighth World Conference on AI in Education, August 1997. (Also appears in the working notes of the Workshop on Pedagogical Agents, held in conjunction with the main conference.)

R. Stiles, S. Tewari, and M. Mehta. *Adapting VRML 2.0 for Immersive Use*. In Proc. VRML 97 Second Symposium on the Virtaul Reality Modeling language, Monetery, CA February 1997.

J. Rickel and W.L. Johnson. *Integrating Pedagogical Capabilities in a Virtual Environment Agent*. In Proc. First International Conference on Autonomous Agents, February 1997.

R. Stiles et al. *Virtual Environments for Shipboard Training.* In Proc., Intelligent Ships Symposium II, The American Society of Naval Engineers, Philadelphia, PA, November 1996.

Workshop papers:

C. Elliott, J. Rickel, and J.C. Lester. *Integrating Affective Computing into Animated Tutoring Agents*. In IJCAI Workshop on Animated Interface Agents, Nagoya, Japan, August 1997.

A. Munro and D. S. Surmon. *Primitive Simulation-Centered Tutor Services.* In AI-ED 97 Workshop on Architectures for Intelligent Simulation-Based Learning Environments, Kobe, Japan, August 1997.

A. Munro, Q.A. Pizzini, D. S. Surmon, and M.C. Johnson, *Cost Effective Learning Environments Through Authoring: Lessons Learned.* In AI-ED 97 Workshop on Issues in Achieving Cost-Effective and Reusable Intelligent Learning Environments, Kobe, Japan, August 1997.

J. Rickel and W.L. Johnson. *Steve: An Animated Pedagogical Agent for Procedural Training in Virtual Environments.* In IJCAI Workshop on Animated Interface Agents, Nagoya, Japan, August 1997.

J. Rickel and W.L. Johnson. *Mixed-Initiative Interaction between Pedagogical Agents and Students in Virtual Environments.* In AAAI Spring Symposium on Computational Models for Mixed Initiative Interaction, Stanford University, March 1997.

h. Attach title and brief description of patents/inventions, if any.

N/A