VOLUME II

Proceedings of the Contributed Sessions

1993 Conference on

Intelligent Computer-Aided Training and Virtual Environment Technology

Sponsored by

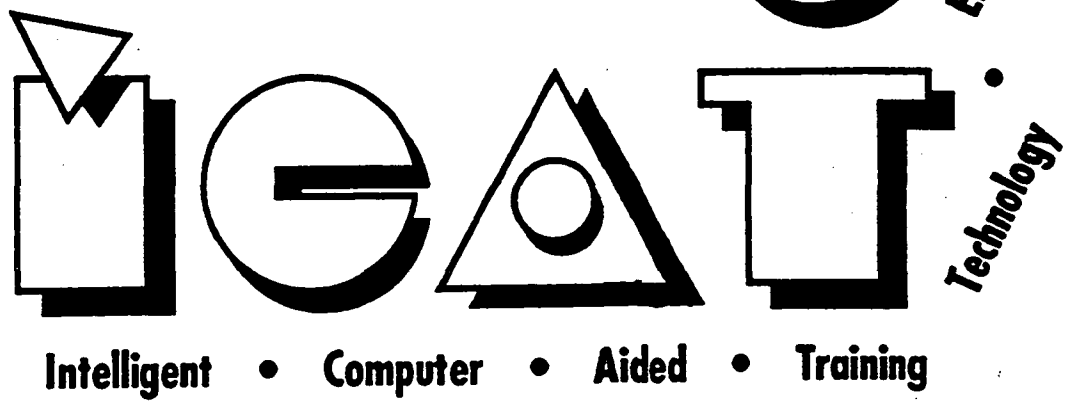
Software Technology Branch
NASA/Johnson Space Center
and
U.S. Army Training and Doctrine Command
and
RICIS/University of Houston-Clear Lake

May 5-7, 1993

Gilruth Recreation Center
NASA/Johnson Space Center

Edited by

Patricia R. Hyde and
R. Bowen Loftin
University of Houston-Downtown



Proceedings of the

1993 Conference on Intelligent Computer-Aided Training and Virtual Environment Technology

edited by
Patricia R. Hyde
and
R. Bowen Loftin
University of Houston-Downtown

A1: VE Training for Space Flight

Patrick J. Kenney, David B. Palumbo — Instructional and Technological Development Activities of a Synthetic Solar System Exploration Environment 1

Michael Goldsby, Abhilash Pahdya, Ann Aldridge, James Maida — A Virtual Reality Browser for Space Station Models 2

Steve Tanner, Kellie Miller — The Use of High Fidelity CAD Models as the Basis for Training on Complex Systems 10

A2: Virtual Environment Hardware

Arden Strasser — Improved Visualization of Virtual Environments 14

John W. Worthington, K. Duncan, W.G. Crosier — Network and User Interface for PAT Dome Virtual Motion Environment System 15

Brian Vandellyn Park — The Personal Motion Platform 20

A3: Knowledge Acquisition for ICAT & VE

Tim Saito, R. Bowen Loftin — Issues in Capturing and Representing Domain Knowledge and Polygonal Assignment for Virtual Environments 26

Pamela K. Fink, L. Tandy Herren — A Knowledge Engineering Taxonomy for Intelligent Tutoring System Development 27

David C. Wilkins — Generic Expert System Shells and Apprenticeship Tutoring: The Representation and Use of Explicit Control Knowledge 39

David B. Palumbo, David Germain, Will Lidwell — The Impact of Cognitive Task Analysis as a Knowledge Acquisition Method in Intelligent Computer Assisted Training Systems 40

A4: Multimedia in ICAT Systems

Charles P. Bloom, Frank Linton, Brigham Bell, Edwin Norton — The Learn, Explore and Practice (LEAP) Intelligent Tutoring Systems Platform: Multimedia Integration 41

Peter T. Bullemer, Rose W. Chu, Nagi Kodali, Mike Villano — The Home and Building Control Fundamental Tutor: A Design Framework for the Multimedia Instruction of Declarative Concepts 58

Stephen Desrosiers, Debra Bettis — SMART—Situated Multimodal Advanced Real-time Trainer 59

B1: VE in Training & Education I

Sharon A. Stansfield — A Computer-based Training System Combining Virtual Reality and Multimedia 60

Brett Achorn, Norman Badler — A Virtual Training Environment with Simulated Agents 65

Andrew F. Payer, J. Mark Voss, Laurie Spargue — Development of Virtual Environments For Medical Educaiton and Training ... 69

B2: Virtual Environment Software I

Dr. Gerald Pitts, Mark Robinson, Steve Strange — A Microbased Shared Virtual World Prototype 70

Georges G. Grinstein, David A. Southard, J. P. Lee — Virtual Environment Architecture for Rapid Application Development 75

C.F. Codella, R. Jalili, L. Koved, J.B. Lewis — A Development System for Multi-User, Distributed Virtual Environments 83

Chuck Kosta
Dr. Patrick D. Krolak — Rapid Prototyping 3D Virtual World Interfaces within a Virtual Factory Environment 84

B3: Models in ICAT Systems

Rose W. Chu
Mike Villano — Application of Knowledge Space Theory for Student Modeling 92

Randall W. Hill, Jr.
W. Lewis Johnson — Impasse-Driven Tutoring for Reactive Skill Acquisition 93

John D. Farquhar
J. Wesley Regian — The Effects of a Dynamic Graphical Model during Simulation-based Training of Console Operation Skills 111

Bradley J. Wiederholt
T. Kiki Widjaja
Joseph Y. Yasutake
Hachiro Isoda — Advanced Technology Training System on Motor Operated Valves 117

B4: ICAT Commercial Applications

Charles P. Bloom
Brigham Bell
Frank Linton
Edwin Norton — The Learn Explore and Practice (LEAP) Intelligent Tutoring Systems Platform 128

Minoru Kiyama
Yoshimi Fukuhara — An Authoring System for Practice Environment in the Network Service Field 145

Clifford M. McKeitham, Jr. — Integrated Intelligent Training and Job Aiding for Combustion Turbine Engines 153

Amir Hekmatpour
Gary Brown
Randy Brault
Greg Bowen
Larry Grant — FTDD973: A Multimedia Knowledge-Based System & Methodology for Operator Training & Diagnostics 161

C1: VE in Training & Education II

Christian-Arved Bohn
Wolfgang Krueger — Embedding Speech into Virtual Realities 171

Major Donald Pryor
James Larsen — Virtual Instrument Technology 179

Laurie McCarthy
Michael Pontecorvo
Frances Grant
Randy Stiles
Spatial Considerations for Instructional Development in a Virtual Environment..... 180

R. Bowen Loftin
Mark Engelberg
Robin Benedetti
Virtual Environments for Science Education: A Virtual Physics Laboratory... 190

C2: Virtual Environment Software II

Peter Astheimer
Sounds of Silence–How to Animate Visual Worlds with Sound..... 191

Long V. Truong
Living Color Frame System: PC Graphics Tool for Data Visualization Applications..... 203

Lac Nguyen
Patrick J. Kenney
Design Strategies and Functionality of the "Visual Interface for Virtual Interaction Development" (VIVID) Tool..... 212

C3: ICAT Architectures & Authoring Systems

Brigham Bell
Charles Bloom
Frank Linton
Edwin Norton
The LEAP Intelligent Tutoring Architecture..... 218

Michael Orey
Ann Trent
James Young
Michael Sanders
Streamling ICAT Development Through Off-the-Shelf Hypermedia Systems..... 219

P. Ahmed
A. Al-Dhelan
A. Shah
On the Construction of a Knowledge-Based Intelligent Tutoring System (KB-ITS)..... 234

Kimberly C. Warren
Bradley A. Goodman
Engineering Intelligent Tutoring Systems..... 235

C4: ICAT Education & Medical Applications

Rodger Marion
Bruce R. Niebuhr
B. Chris Renten
John Bernstein
An Idealized Computer-Based Patient Record for Teaching Medical Diagnosis and Care Planning..... 244

K. Macura
R. Macura
V. Toro
E. Binet
J. Trueblood
Case-Based Teaching System for Radiology..... 252

Madeline Kovarik — Intelligent Computer Aided Training Systems in the Real World: Making the Technology Reachable for Everyday Educators 256

Noah Rifkin, Hans ten Cate, Alison Watkins — ICAT and the NASA Technology Transfer Process 260

D1: Assessing VE for Training

Mark S. Schlager, Randall J. Mumaw, Douglas G. Hoecker — Analytical Tools For Designing Virtual Environment Training Systems: Nuclear Power Plant Maintenance Applications 261

Donald R. Lampton, James P. Bliss, Bruce W. Knerr — Assessing Human Performance in Virtual Environments 270

Joseph P. Hale II — Marshall Space Flight Center's Virtual Reality Applications Program 271

D2: VE & Human Systems I

Joseph Psotka — Virtual Egocenters as a Function of Display Geometric Field of View and Eye Station Point 277

Joseph Psotka, Sharon Davison — Cognitive Factors Associated with Immersion in Virtual Environments 285

D.E. Parker, D.L. Harm, F.L. Florer, K. Davino, N.C. Skinner, J.T. Kuennen — Attitude Awareness Training: An Aid to Effective Performance in the Virtual Environment? 298

D3: ICAT Theory & Natural Language

Frank Linton, Brigham Bell, Charles Bloom, Edwin Norton — Tutoring Methods and Strategies in LEAP 305

Michelle R. Sams — An Intelligent Foreign Language Tutor Incorporating Natural Language Processing 315

Wayne A. Nelson — *Knowledge Acquisition and Interface Design for Learning-on-Demand Systems* 321

Bruce Porter, Art Souther — Improving the Explanation Abilities of Tutoring Systems 328

D4: ICAT Applications in the Military

Tucker Maney, Henry Hamburger — VIS/ACT: The Next Episode 338

Sylvia Acchione-Noel, Joseph Psotka — Mach III: Past and Future Approaches to Intelligent Tutoring 344

Walter G. Butler, Richard R. Laferriere, Philipp A. Djang — Training Mix Model 352

E1: VE Applications in Engineering

Cory W. Logan, Henk Prenger, Jim Clark, Liew Wu, Michael E. Goldsby, Jim Maida — Virtual Environment and Computer-Aided Technologies Used for System Prototyping and Requirements Development 361

Steve Bryson — The Virtual Windtunnel: Visualizing Modern CFD Datasets with a Virtual Environments 390

Mark Bolas — Practical VR--Five Years of Lesson's Learned 396

E2: VE & Human Systems II

K. M. Duncan, D. L. Harm, W. G. Crosier, J. W. Worthington — Using Virtual Environment Technology for Preadapting Astronauts to the Novel Sensory Conditions of Microgravity 397

D.L. Harm, F.E. Guedry, D.E. Parker, M. F. Reschke — Development and Implementation of Inflight Neurosensory Training for Adaptation/Readaptation (INSTAR) 402

Robert S. Kennedy, Kevin S. Berbaum, Lawrence Hettinger, Michael Lilienthal — Ataxia and Other Posteffects of Flight Simulators: Should Virtual Reality Systems Have Warning Labels? 407

E3: Knowledge Acquisition for ICAT

Vincent J. Kovarik Jr. — Autononmously Acquiring Declarative and Procedural Domain Knowledge for ICAT Systems 408

Will Lidwell
David Palumbo
Automating Knowledge Acquisition Processes: An Analysis of the Psychological and Technological Issues Surrounding the Automated Elicitation of Expert Knowledge 415

David Germain
Bob McNenny
David Palumbo
Beyond Knowledge Acquisition - A Blueprint for Knowledge Construction 416

Debra Bettis
Stephen Desrosiers
David Mulcihy
Ken Ruta
Robo-Cat–An Intelligent Robotics Trainer 417

E4: ICAT Applications in Aerospace

Mark Rolincik
Michael Lauriente
Harry C. Koons
David Gorney
An Intelligent Computer-Aided Tutoring System For Diagnosing Anomalies Of Spacecraft In Operation 418

Capt. Keith B. Shoates
Patrick Moran
Training Augmentation Device (TAD) for the Air Force Satellite Control Network (AFSCN) 419

The LEAP Intelligent Tutoring Architecture

Brigham Bell, Charles Bloom, Frank Linton, and Edwin Norton
U S WEST Anvanced Technologies
4001 Discovery Drive, Suite #270
Boulder, CO 80303
bbell@uswest.com

The LEAP (Learn, Explore, And Practice) Intelligent Tutoring System is a general platform for teaching procedural knowledge through simulation. Taking a LEAP course involves working through many different simulations and, when desired, perusing related declarative material. LEAP employs several strategies to ensure that students are continually but not overly challenged. One strategy picks simulations on the fringes of a student's abilities, skipping over simulations considered too easy for the student, and presenting simulations relating to recently viewed declarative material. Another strategy skims easy parts of a simulation so that the student is quickly brought to challenging parts without losing context.

The LEAP Architecture accomplishes these strategies independent of the domain through the use of a task grammar, which is an abstract representation of the procedural knowledge. When a student answers questions about a specific simulation, the performance is actually recorded on the task grammar. Since all simulations are linked together through this grammar, skimming over questions and whole simulations is a straightforward matter of referring to the performance in the underlying grammar. The task grammar is also linked to topics of the declarative material, enabling a student to jump between practicing procedural knowledge and viewing related declarative information.

Streamlining ICAT Development Through Off-the-Shelf Hypermedia Systems

Michael Orey
The University of Georgia
Ann Trent, James Young, and Michael Sanders
Army Research Institute, Fort Gordon

This project examined the efficacy of building intelligent computer assisted training using an off-the-shelf hypermedia package. In addition, we compared this package to an architecture that had been developed in a previous contract which was based in the C programming language. One person developed a tutor in LinkWay (an off-the-shelf hypermedia system) and another developed the same tutor using the ALM C-based architecture. Development times, ease of use, learner preferences, learner opinions, and learning effectiveness were compared. In all cases, the off-the-shelf package was shown to be superior to the C-based system.

It seems as though the field of Instructional Systems Development (ISD) has abandoned any desire to work on the design and development of Intelligent Tutoring Systems (ITS, Rosenberg, 1987; Schank, Personal Communication, April 5, 1991). There are many reasons for this purposeful disinterest. One of the primary ones has to do with the amount of time that it takes to build a substantial ITS. This paper reports on the examination of using off-the-shelf development packages in order to streamline the ITS development process.

There have been many people who have discussed the complexity of ITS development (Jonassen & Wang, 1991; Merrill, Li, & Jones, 1991; Orey & Nelson, in press; Towne, 1991). Merrill and his colleagues have suggested that traditional computer-based training takes about 100 hours of design and development for every one hour of instruction. Further, they have estimated that the design and development time for ITS is approximately 500 hours per hour of instruction. At least one of the current authors has bragged about the time it took to design and develop an ITS. One of these tutors, a whole number subtraction tutor, took 800 hours of programming in Lisp to develop. Children who have used this tutor, average about 5 hours to achieve mastery. Therefore, it took about 160 hours of time to develop an hour of ITS instruction. This seems prohibitively long and is at the root of the prohibitiveness of ITS.

An ITS must include an expert component. Jonassen and Wang (1991) presented an expert system for science at the annual meeting of the Association for Educational Communications and Technology. They intimated that the development time using an expert system shell for this expert system (not ITS) was very lengthy. The response by the audience was clear. Why spend so much time for so little program functionality? Anderson (1988) suggests that the ITS must "know" the domain to teach the domain. Therefore, the expert knowledge for the domain must be encoded. Expert system shells streamline this process, but Jonassen and Wang's experience suggests that it is still not practical for instructional development purposes.

Doug Towne (1991) has begun work on an automated ITS development system. Like Merrill, he predicts that by using his system, considerable savings will be achieved. He suggested that an expert in the domain could sit down to his system and spend one hour going through it and walk away with one hour of ITS instruction. This is an honorable goal and will certainly make ITS practical. However, this system has not yet been developed, so in the meantime, we have been working on how to streamline the process without ITS specific tools.

During an Artificial Intelligence in Education class taught at the University of Georgia, we examined how to minimize the development time. As part of this class, one of the authors of this paper created an ITS to teach teachers how to link screens together using hyperCard®. This tutorial was created in hyperCard and used Xrules to program the help button. In this way, the developer could use hyperCard to interactively create the interface, use hyperTalk (a high level English-like programming language) to program the simulation, and use Xrules to represent the expert's knowledge in the domain. It took 8 hours to create the simulation and the tutor and the instruction lasted about 30 minutes. This result would suggest that ITS is not impossible or impractical.

At the root of this development process is the idea of what an ITS is. Orey and Nelson (in press) have suggested that there is not a dichotomy between traditional computer-based tutors and ITS, but that there is a continuum. This continuum is based on the level of interaction that is required of the learner. A passive system would be at one end of the continuum. By passive, it is meant that the learner sits in front of the computer screen and information is presented. At most, the learner presses a key to continue. This kind of instruction can be enhanced with questions. The learner is presented with a couple of screens of information followed by a question that is based on the information on those screens. Finally, the other end of the continuum is anchored by the ITS where the information is presented in the context of solving real problems. The information can be presented at the time of an error (a key learning time) or can be presented when requested in the context of solving a problem (when the learner needs to know the information). This system would "know" the domain, "know" what instructional strategy is best for this particular learner, and "know" what the learner does and does not know (or incorrectly knows) about the domain. The tutoring systems described here are not at this end point, but are towards this end. Error feedback is provided based on what the learner has done or left undone, and the advice that is provided upon request is based on the same criteria as error feedback. These systems do not "know" the domain, learner, or pedagogy in any real sense, but can provide appropriate feedback. We now turn to an analysis of two development environments.

COMPARING DEVELOPMENT ENVIRONMENTS

The content was operations procedures for one of the Army's mobile telephones (the Mobile Subscriber Remote Telephone or MSRT). We examine two different development environments -- an off the shelf hypermedia tool and the C programming language. Two programmers worked on this project. One used IBM's LinkWay® to develop a hypermedia-based ITS. The other programmer used a collection of C routines that had been used in a previous ITS to develop a new ITS in a slightly new domain. The C routines were originally developed for both operations and troubleshooting procedures for one of the switching shelters used in the Army's communications equipment. The idea was that considerable time savings could be achieved if the developer of the new tutor (the MSRT tutor) used the existing C routines (libraries). The original tutor, ALM (see, Orey, et al., 1991), essentially was a hypermedia based ITS. Its primary functions were to provide appropriate advice throughout the problem solving process and to provide corrective feedback. The second programmer chose to use a hypermedia tool and build the tutor from scratch. The constraint for both systems was that the system must run on an MS-DOS machine in EGA mode and use 640 K or less of RAM. LinkWay® is a hypermedia tool that meets those requirements. From here on out, we will refer to the LinkWay version as MSRT-L and the version based on ALM and written in C as MSRT-C.

It should be noted that the C programmer had many hours of experience with the original ALM code. He had written several alternative implementations of the ALM program prior to developing his MSRT tutor. The LinkWay® programmer had extensive experience in developing ITS and using hypermedia tools such as Apple's HyperCard and Asymetrix's Toolbook, but had no experience with LinkWay®.

Functionality of the Systems

The MSRT is essentially made up of several pieces of equipment (a radio, a phone, a power supply and various connections). Each piece of equipment had push buttons, toggle switches, knobs, and/or light indicators. Part of the ITS development and its functionality was to create a simulation that allowed lights to "light up" and allow the learner to throw switches, push buttons, and turn knobs. It turns out that the most important aspects of the simulation for learning were related to lights and sounds. That is, the lights blink, go on, or go off according to what the learner did to the knobs, switches and buttons. In addition, the radio and phone produce various tones as the result of actions performed by the learner. Much of the development time was spent on getting these equipment reactions to accurately correspond to the actions performed by the learner while allowing the learner the freedom to make errors as they attempt to operate the equipment. This freedom was constrained to a degree. Certain knobs, switches and buttons were not used at all in some procedures. If the learner were to click on these, they would receive an error message. Also, certain sequences of actions were required. If the learner performed an action out of sequence, an appropriate error message was provided.

In addition to the error feedback and the simulation programming, there were two other primary foci of the development process -- the "Coach" button programming and the navigation requirements of the system. First, the Coach button is probably the most "intelligent" part of these systems. When a learner clicks on the Coach button, the program needs to determine the most appropriate action that the leaner needs to take given the goal of the problem. This achieved through a collection of state variables and solution step variables. Essentially, every knob, switch and button has a state variable associated with it. As the item is altered by the learner, the state variable is updated. In addition, each step of the procedure is represented by a variable that indicates whether the step has been completed or not. In this way, the program can know that the learner may have set a switch to off at an appropriate time (the step variable), but later changed it to on (the state variable). This state change might then cause a conflict in the equipment (perhaps an electrical shorting of the equipment). The Coach button therefore, might check that the knob is in the correct location and that the preceding step had been completed before suggesting that the student execute a step that might cause a problem.

The final aspect of the system that requires some time is to establish links between screens. These links can be in the form of transparent buttons over pieces of equipment or buttons on the bottom of the screen that indicate the piece of equipment (say the phone). In this way, the learner can quickly and easily move from one device to the next without much cognitive effort and therefore, little interference with the carrying out of the procedure. These are the common development efforts between developing in C and LinkWay. There were also graphics that needed to be created, but just one person did this and the two systems used them. The C programmer had created these graphics prior to beginning the procedures, so times are for the tasks mentioned above, and do NOT include the time taken to create graphics. The graphics took approximately 80 hours. The images were scanned into the computer, edited, and colorized. There was a total of 10 images and it took about a day to do each one.

Limitations of MSRT-L

1. There is not much control over the placing of buttons. The vertical displacement is approximately 20 pixels and the horizontal is about 10 pixels. This was a problem when trying to place buttons over the drawings that represent the MSRT equipment. LinkWay Live (the new version) allows pixel by pixel movement (this project began with LinkWay 2.01 and finished with LinkWay Live).

2. The Paint program would not allow you to select more than about 20 percent of the screen at a time. So if you wanted to move the image to the right a bit, you could not (or shrink it or any other manipulation).

3. While LinkWay has built in utilities for using sound boards from IBM, it does not have any ability to make differing tones (something quite useful when building a phone tutor). The only sound resource available was to play a simple beep. However, you could write the tone routine in another language and run it from within LinkWay (which is what was done).

4. One of the ways LinkWay handles the limitations of a 640K environment is to limit the objects and scripts associated with those objects. The limit is 10K of memory for each screen. This limit was reached several times. The recommended solution is to use multiple screens to represent the necessary actions for that screen. Therefore, you might have half of the knobs representing the Digital Secure Voice Terminal (DSVT) on one screen and clicking on any of the other half of the knobs would send you to another screen that has those knobs represented. This was deemed a poor alternative. We found that the literals associated with error and status messages could be provided on a separate screen without limiting the simulation fidelity. Therefore, pop up messages were eliminated and messages were provided by taking the learner to another screen to display the text. Clicking on anything would return them to the equipment on which they had been working.

5. Another limitation is that each object is limited to 4K of script. However, other objects can contain procedures that the object can then call. The only problem that this caused was that everything that was added to a script could get lost (in our case the most lost was about ten minutes).

6. The programming utilities were minimal. You could not find words, replace words, select pieces of code, debug code, or move code easily. Copying a piece of code from one place to another could only be done one line

at a time and the manual did not indicate that this could even be done (the LinkWay programmer learned from a friend who has used LinkWay a great deal how to copy code and delete more than one line of code).

7. Apparently, as the page size increases, strange things begin to happen. One of those things is that when you leave an MSRT screen to go another (say Advice) screen, some times a box corrupts the screen. This box is where the mouse cursor happens to be. The only fix is to hide the mouse icon before leaving the screen and show it after displaying the next screen. This is cumbersome and makes a link button useless.

8. Another problem that has arisen manifests itself while simulating a flashing light. Frequently, during the procedures of operating the MSRT a flashing light appears on the equipment. This was simulated by placing a button over the light and alternating a filled circle and open circle icon on top of the light. However, if the learner moves the mouse pointer into the flashing light area, stray lines appear in that area. As long as the mouse is in that area, these lines continue to appear. Again, the fix is to hide the mouse icon before changing the icon of the light and showing it after the light icon is displayed (or use Bitputs instead of icons).

9. LinkWay uses its own graphics formats. This is very IBM. Rather than use one of the many standard formats, IBM chose to use its own. Hence, we needed to use a graphics utility to display the PCX file and then use LinkWay's screen capture utility to save it in LinkWay's format. However, we wanted the system to be in a EGA format and the machine we were using had a VGA monitor. Every time we tried to capture a screen, it saved it in VGA format. The solution was to find and use a machine that had an EGA monitor.

10. The ALM architecture allows the user to either click the mouse button or press a function key (although this is not support for the interaction buttons like a knob for example). There is no apparent way to have a button execute with either a mouse click or a specific function key press. Buttons only execute when they have been activated by a mouse click.

11. The ease of development comes at the expense of loss of control. Many things are automatically handled by LinkWay (the way the mouse cursor looks and behaves, how buttons look and behave, how screens are displayed, how scripts are executed). It may occur that you do not want these things to happen the way that LinkWay handles them. In ALM, these things are modifiable. In LinkWay, it is unclear if they can be changed. For example, the mouse cursor changes from an arrow to a hand when it comes into the area of a button in LinkWay. This may be too subtle for the learners to react to. In ALM, when you encounter a button, a square area is depicted around the "hot" area of the button. It clearly indicates that this is a button.

Limitations of MSRT-C

1. The way that screens are drawn within MSRT-C is that the system calculates what goes on the new screen, then it removes what is currently on the screen and pops up the new screen. This takes about 3 seconds on a Zenith 286 EGA system. What this means is that the learner clicks on a button to go to the next screen and nothing happens for 3 seconds. During that delay, the learner could try to do something else. The result is the first thing that was clicked gets executed at the time the second thing is clicked. This causes some confusion.

2. A second problem in MSRT-C is that buttons (their location, size, and nature) are defined in a textual resource file. This means that to place an MSRT-C button on top of a picture of a switch, you need to guess about the screen coordinates of its location and size, enter this information, run the program, go to the correct screen, see where the button actually showed up, quit, adjust parameters, and do it again. In MSRT-L, you place and size it on the screen with the picture already displayed. This is much easier and faster.

3. When you want to write the program that attaches to the button you just placed, you need to do the following. First, you need to create the case instance to turn control over to the procedure that you need to write for the button. Second, you write the procedure for the button. Third, you recompile the code and relink it. Fourth, you run the program and test the button to see if it works. Last, you revise the procedure and do it again. In MSRT-L, when you create a Script button, you are provided a small editor and you can write the code to be executed. When you click on the close box, you can then double click on the button to see if the code works and revise it accordingly. This is tremendously faster.

4. When you want to add an additional screen to the tutor, MSRT-C uses a similar process to the one for adding a program to a button. The programmer for the MSRT-C tutor estimated that the act of creating a new screen (if the graphic is already created) is about 20 minutes. The process includes copying the screen loop code, writing the case instances for each screen that would call the new screen, write any code that needs to execute when the screen first displays (like current icon settings), and creating definitions for the screen in the resource file. In MSRT-L, it took 26 seconds to do the same thing. Over a large project this can be a big difference.

5. The ALM architecture is not an event driven object oriented architecture like LinkWay. The event loops are separate loops with giant case statements that have to be modified with each addition and deletion of objects in the program. When you delete an object in MSRT-L, you also delete its code. There is no clear way to do this in MSRT-C. You have to delete code in many different parts of the program and resource files.

6. MSRT-C is not an integrated environment. However, there are several tools that can be integrated. For example, you can leave MSRT-C and start PC Paint to create your graphics. Both MSRT-C and MSRT-L have poor icon editors. Multiple Fonts are not possible in MSRT-C unless you paint them on in another package.

7. When you reach a memory limit in LinkWay, you lose the code you were just working on and LinkWay informs you that you have reached a limit. MSRT-C reaches a limit and system crashes occur at non-systematic times. This makes MSRT-C a much less stable environment. There have been cases where a small modification to the program required a week of reprogramming to get the system to recompile and run again.

Summary

Although there were many limitations to the LinkWay environment, most of the limitations could be resolved in some way (which implies that they are not limitations). Our overall impression of this programming environment is positive. It is a powerful hypermedia development tool that meets the needs of the Army. The MSRT-L tutor runs in 640 K, displays on an EGA monitor, and fits on one 3.5 inch diskette. These are all positive aspects. Development time, however, is the overwhelmingly positive aspect.

Time Comparisons

The times that it took to develop each version of the MSRT tutor were quite disparate. It should be noted that a conservative estimate for the time that it would take to achieve mastery of the procedures that were simulated (there were four in total) is about 3 hours of instruction. The programs were essentially the same. They covered the same procedures and had the same screens. Factoring out the graphics production, the MSRT-C tutor took 180 hours to develop using the existing C libraries. Included in this estimate is the time taken editing the icons so that all the knobs, switches and buttons could be represented.

The MSRT-L programmer was required to do the same tasks. The only difference was that this programmer also had to learn LinkWay. The outcome is that it took him 75 hours to create the same system. It took the C programmer 2.4 times longer than the LinkWay programmer. This in itself is reason to choose to use LinkWay over C to develop these programs. The functionality of these systems was essentially the same. If anything, the MSRT-L version was more flexible. For example, the MSRT-L version would allow the learner to set a knob in an incorrect position. The MSRT-C version would inform the user that an error occurred if they moved the knob beyond where it was supposed to be set. The former is more like the actual equipment and therefore, higher transfer would be expected.

t is estimated that the graphics took about 80 hours to create. Combining this estimate with the development time for the MSRT-L version, the total time to develop the MSRT-L tutor was 155 hours. Given the conservative estimate of 3 hours of instruction, this translates to about 50 hours of development time for every hour of instruction. This is a considerable savings over the 160 hour development time that was involved in POSIT (Orey & Burton , 1989/90). In fact, instructional developers ought to take notice. ITS is a doable technology. It is time to take the ITS development out of the research labs and put it into the development labs.

COMPARING THE RESULTING INTELLIGENT TUTORING SYSTEMS

The above comparisons were conducted by the research and development staff at the Fort Gordon Army Research Institute. In addition to this subjective comparison, we conducted a formal evaluation that compared the two systems. This comparison included subjective measures and objective measures. It should be noted, however, that the two systems were implemented by two different developers and specific design decisions were implemented differently. The most apparent difference in the systems manifests itself in the form of the advice statements provided by the Coach. Anderson (1990) has suggested that when people learn procedures, that they go through a variety of stages and use a variety of processes. The basic idea is that the learning of procedures proceeds from the small verbalizable component parts of the procedure to the automatic (non-conscience) execution of the procedure. During the initial learning of these component pieces of the procedure, processes can be used to intervene or aid in the acquisition of new components. For example, if the learner is trying to execute one of the MSRT operations procedures and they know that they need to connect one piece of equipment to another (but they have not done this before), they may use knowledge about how to connect other pieces of equipment (say a VCR to a TV) to aid the acquisition of the new component. In this way, the learner reasons from a known example to the new example. This process aids both the execution of the procedure as well as the retention of the new component of the procedure.

When building an ITS that provides advice, this theoretical idea is very important. Essentially the Coach function in the MSRT tutors determines the next appropriate component of the procedure that the learner needs to execute. It then needs to provide the learner with this information. Both MSRT tutors do this. However, the nature of the messages provided differ. The MSRT-C developer decided to provide strictly procedural information. For example, at one point in a procedure, the learner is required to remove a piece of equipment from another. There are several component parts of this procedure. The MSRT-C version of the tutor provides this message:

Set KYK-13 Z/ON/OFF CHECK switch at OFF CHECK. (Go KYK-13 screen).

This is the first step to executing this procedure. The MSRT-L version provides this message for the same step:

Now return to the KYK-13 screen and begin to disconnect the cryptovariable cable. This process begins with the setting of the KYK-13 to the OFF position.

The intention of the first developer was to be direct and limit the message to strictly procedural information. The intention of the second developer was to tie the components conceptually together.

A second difference between the two versions was in the ability to navigate through the system. The original approach to navigation was to present a battlefield map that indicated a variety of communication equipment. The learner would then click on the MSRT and be shown typical layout of all of the MSRT components. The learner could then click on the piece of equipment that they wanted to work with and the system would take them to a screen depicting that device (with all of its lights, buttons, switches, and knobs). The learner could then perform any operation on that device that was appropriate at that time for that procedure. When they needed access to another device, they would click on the "backup" button and it would show the main MSRT screen again. The MSRT-C developer decided to include buttons on the individual screens that would allow the learner to go to another device without returning to the main MSRT screen. However, because it is difficult to alter screen layouts in the MSRT-C version, these buttons may be at the top of one screen and on the right side on another screen. Further, not all devices were made available from each device. This is not necessarily good interface design, but it was easiest way to implement this change.

The MSRT-L developer decided that the idea of being more "hyper" about access to equipment was a good idea. However, in about half an hour, the MSRT-L developer created a button for each screen on the base page and separated these buttons by placing them in a color coded area at the bottom of the screen. What this means is that the learner could now get to any device from any other device with one mouse click. Beyond this difference, the two systems were essentially the same. Figure 1 depicts a typical depiction of a device for the two versions. Given these differences, we conducted the following evaluation.

METHOD OF EVALUATION

Participants

he participants for this evaluation were 24 (13 in the MSRT-C group and 11 in the MSRT-L group) Signal Corp soldiers (enlisted) that were receiving training on basic communication equipment. Ages ranged from 18 to 35 years and averaged about 21 years. Education ranged from twelfth grade through completion of 2 years of college and the average was about a half year beyond the high school diploma. Because the collection of data took place over a period of several weeks, participants were at varying points in their communication equipment curriculae. They ranged from the fifth week of training to the ninth week of training and averaged about six weeks. All soldiers were required to know how to operate the MSRT and therefore, the participants had an interest and need in learning from the systems. The abilities of the soldiers in the two regroups were roughly equivalent. The MSRT-L group (m=111.2, sd=4.98) had slightly higher IQ estimate than the MSRT-C group (m=110.2, sd=7.88). This measure is described below. Further, the correlation between this ability measure and the performance measures used was quite small (the largest correlation was -0.19 with the time spent on the first procedure).

Materials

The two tutors collected specific data. They saved what procedures were worked on, how long it took to solve the procedure, how many errors were made, and how many times the participants used the Coach button. The errors, advice, and time on the procedure were the main objective measures of learning. Besides the two tutors, there were three other instruments used in the study. To make sure that groups were of equal ability, the Shipley measure of ability was used. The Shipley has a test-retest reliability of 0.80 and correlates with the WAIS intelligence quotient at 0.90. In addition to this ability measure, two questionnaires were constructed. The first asked eight four-choice questions (bad, poor, good, excellent) related to the tutors. This was followed with four open ended questions about the tutors and a general comments section. The second questionnaire was the same as the first, except that it included a third page that addressed the comparison of the two tutors. It also included two items about how much experience with computers they had (none-monthly-weekly-daily) and the other asked what type of computer they used.

Procedure

The idea of the procedure was suggested by Richard Goodis (an employee in the ARI lab). The basic idea is to get people to learn one of the tutors and then have them evaluate both systems. They probably would become attached to the system that they learned on and would likely not evaluate the other system as highly as "their own". However, if the system they learned on was much poorer than the other system, then they might rate the other system higher. Essentially, this is the procedure that was used. First, the participants went through each of the four procedures twice and in sequence (approximately 80 minutes). At this point, we asked them to do the first procedure again. This was the third time they did this procedure and the time to complete, errors, and accesses to coach were the performance measures that we used as objective measures.

After completing the learning phase and on-computer testing, we asked them to fill out a questionnaire that evaluated the system they learned on. Having completed the questionnaire, we then asked them to examine a different tutor (the other tutor) by performing the first procedure again. We also asked them to do this keeping in mind that they were going to be asked to compare the two versions. After completing the procedure, they were again asked to fill out a questionnaire which evaluated this "new" system.

Analysis

There are a variety of comparisons that were conducted on these data. First, t-tests were used to determine if significant differences were to be found between the performance of the two groups on the first procedure and the test procedure. The very first procedure they solved would give an indication of how easy the system was to learn. The test procedure (third time the first procedure was solved) was used to see how well they learned the

procedure from the system. An ANCOVA was considered if the random assignment failed to yield equivalent ability groups, but since they were considered equivalent, a series of t-tests were used.

For subjective comparisons, t-tests were run on each of the 16 multiple choice questions (there were 8 of these questions on each questionnaire). The open ended questions were analyzed by frequencies of categories of responses. In other words, the responses were listed and similar comments were grouped and counted. The most common comments and a selection of others are reported here.

RESULTS AND DISCUSSION

To begin the discussion of results of the evaluation, we will examine the objective measures. There were essentially three performance measures at three points in the evaluation (a total of 9 t-tests. see Table 1). The first point when we collected the performance measures was on the very first procedure that the learners solved. The reason for including this measure is that this would give an indication of the ease of learning the operating mode of the intelligent tutoring system (not necessarily learning from the system). As can be seen from Table 1, all of the performance measures with regard to the first procedure are significant at the 0.05 alpha level. The MSRT-L group took less time, used the coach button less frequently, and made fewer errors than the MSRT-C group. The reason for this quite likely is due to the difference in the structure and content of the text messages. There are 15 steps in the first procedure. The MSRT-L group used the coach button on the average 18 times. Therefore, assuming that the soldiers had no prior knowledge of MSRT (which they did not), there is almost a one-to-one correspondence between the coach button and the number of steps. The MSRT-C group, on the other hand, needed to access the coach button 26 times. One explanation is that the coach messages in the MSRT-C version were more difficult to understand than the MSRT-L version. Another possible explanation is that the navigation buttons in the MSRT-L version made it easier to understand how to execute the message that was provided by the coach button.

The fact that the MSRT-C group also made a large number of errors on the first procedure (14) relative the MSRT-L group (4) does not help decide which explanation is more plausible. The errors may have occurred as the result of misunderstanding the message or may have resulted from misunderstanding how to carryout the action described in the message within the two alternative systems. The differences in time do not shed any more light on these alternative explanations. After all, the MSRT-C group was busy making errors and asking for advice from the coach over and over again. Of course it would take them longer to finish the procedure. The differences in the two designs are related to the two development environments. It is much more straight forward to create navigation buttons that are easily understood and create large elaborate text messages that can be used by the coach function in MSRT-L than in the MSRT-C architecture.

Another way of determining ease of use is to look at the objective measures when the groups switched to the other system. After learning each of the operations procedures from one system (for a period of approximately 80 minutes) the participants were asked to examine another version (the other version). Therefore, the MSRT-L group examined the MSRT-C version and the MSRT-C group examined the MSRT-L version. When looking at the performance data from the "other" version, only one measure was significant at the 0.05 alpha level (see Table 1). That measure was the frequency of accessing the coach button. The MSRT-L group (using MSRT-C, about 18 times) used the coach button more frequently than the MSRT-C group (using MSRT-L, about 12 times). The MSRT-L group returned to a level of coach use that was nearly as high as when they first used the MSRT-L version. They used coach about 18 times on the first procedure and 17 times on the other system, even though this is the fourth time they have performed this procedure. The data from the test procedure (the third time they performed the first procedure) show that they only used the coach button about 12 times on average. The MSRT-C group only slightly increased there use of coach (about 2 extra accesses) from the third time (same system) to the fourth time (different system). This may indicate the ease of use of the MSRT-L version is the explanatory factor rather than the differences in the text messages.

Table 1. Tables of means, standard deviations, t values and probabilities for each of the performance measures at each learning point in time.

	Groups					
	MSRT-L				MSRT-C	
	Mean	s.d.	t	p	Mean	s.d.
First time	939.00	317.17	2.07	*0.05	1295.92	492.30
First coach	18.64	7.37	2.03	*0.05	26.39	10.64
First errors	4.00	3.66	3.58	*0.00	14.39	8.99
Test time	242.27	96.35	-0.47	0.65	225.08	83.80
Test coach	11.91	4.01	-0.70	0.50	10.54	5.40
Test errors	0.64	1.21	2.62	*0.02	2.31	1.80
Other time	740.82	246.25	0.39	0.70	661.62	636.27
Other coach	17.73	6.68	-2.34	*0.03	12.15	4.98
Other errors	7.00	4.47	-1.95	0.06	4.15	2.58

Note: Times are in seconds. Errors and coach accesses are frequency counts. First indicates first procedure. Test indicates the last procedure on the system on which the subject learned. Other indicates the procedure solved on the system opposite from the one on which they learned.
*significant at the 0.05 level

Recall that the first procedure requires 15 different steps to successfully "power up" the MSRT. Both groups had reduced their access to advice by the third time to less than the number of steps. Therefore, they must have remembered the other steps and could rely less on the coach button. When they were introduced to the "other" version, the MSRT-C group was able to perform some of the steps without asking for help from the coach button (they used the coach button 12 times on average). A possible explanation is that the MSRT-L version was easy enough to understand that they could simply carryout their learned procedures in the new environment. Another explanation is that the MSRT-C version is better for transfer. However, combined with the subjective data, this seems like an unlikely explanation. Therefore, the best explanation considered to this point is that the MSRT-L version is easier to operate.

A final comment on the data collected on the "other" system is that the pattern of results is similar to the first procedure. That is, the MSRT-L group took longer, used coach more frequently, and made more errors than the MSRT-C group when they were using the "other" system. While time and errors were not significant at the alpha level, it is interesting to note that the pattern of results based on comparing means is similar when people use the MSRT-C system. Again, this can be attributed to an explanation which is based on ease of use.

The final group of performance data are those related to the "test" phase of the procedure. After having performed each procedure twice, the subjects were asked to perform the first procedure a third time. This was used as a measure how well they had learned this procedure. The only statistically significant result was that the MSRT-L group made fewer errors than the MSRT-C group. A probable explanation for this is that the MSRT-L version displayed errors messages on a bright red screen that had a title at the top stating, "Errors". The MSRT-C version displayed a window with a red background and no clear note that the message was an error message. In addition, the MSRT-C version used the same error message for all errors. The MSRT-L version used error message specific to the error that was committed (i.e., the MSRT-L version was more ITS like than the MSRT-C version). The reason for this is that the error handling is difficult and time consuming in the MSRT-C version. It is much easier to implement in the MSRT-L version. Therefore, MSRT-L may be better suited for ITS development than ALM. The other performance data do not indicate any differences. It took both groups about the same amount of time and used coach with almost equal frequency when performing the start up procedure.

Table 2. The means, standard deviations, t values, and probabilities for each question both for the system the subject learned on and the "other" system.

	Groups					
	MSRT-L				MSRT-C	
	Mean	s.d.	t	p	Mean	s.d.
Learned on						
Interface	3.64	0.67	1.41	0.17	3.23	0.73
Access	3.82	0.41	3.50	*0.00	2.92	0.76
Realistic	3.55	0.52	1.94	0.07	3.08	0.64
Coach	3.64	0.67	1.46	0.16	3.15	0.90
Enjoyment	3.46	0.93	1.37	0.18	2.92	0.95
Errors	3.09	0.83	0.52	0.61	2.92	0.76
Confidence	3.46	0.52	1.59	0.13	3.00	0.82
Overall	3.64	0.67	0.64	0.53	3.46	0.66
Other system						
Interface	2.55	0.93	2.81	*0.01	3.46	0.66
Access	2.00	1.00	5.77	*0.00	3.77	0.44
Realistic	2.91	1.14	0.89	0.38	3.23	0.60
Coach	2.55	1.04	2.22	*0.04	3.31	0.63
Enjoyment	2.18	1.17	3.18	*0.00	3.31	0.48
Errors	2.36	1.29	1.85	0.08	3.08	0.49
Confidence	2.36	1.12	1.79	0.09	3.00	0.58
Overall	2.09	1.05	3.59	*0.00	3.23	0.44

Note: The means are based on an integer scale from 1 to 4 where 1 is bad and 4 is excellent. The two important categories are the evaluation of the system on which the subject learned and evaluation of the other system on which they did NOT learn.

In summary, both the performance data from the first procedure and the "other" procedure indicate that MSRT-L is easier to use than the MSRT-C version. Further, on the learning measures, MSRT-L may be better for learning than the MSRT-C architecture. Therefore, the MSRT-L version is easier to use and students learn better from it. We now turn to the question of preferences based on the questionnaire data.

The results of questionnaire component of this evaluation are presented in Table 2. To begin this discussion, refer to the first group of questions. These are labeled "Learned on". After the subjects completed using the first system that they learned on, they were asked to fill out a questionnaire. This section represents the results of this phase of the procedure. If you examine the column of probabilities, the only significant result is that MSRT-L group rated the MSRT-L version higher when asked about how easy it was to go from screen to screen. As described above, this was easily done in LinkWay. Not only was it easy to program, but it makes it easier for the learners to navigate through the system. The only other observation to be made with regards to the first group of questionnaire responses is that although the other areas do not indicate a significant difference between groups, the means for each question are higher for the MSRT-L group than the MSRT-C group. While this is not statistically significant, it certainly suggests an interesting pattern, especially given the results of the second part.

After the subjects had used one version for about 80 minutes, they were asked to examine the "other" system. They were asked to perform the start up procedure for the fourth time, except that this time it was on the other system. In addition, they were told that they would be evaluating this system using similar (to the first one) questionnaire. The results of this questionnaire are in the second part of Table 2.

The first point to be made about the second part is that the subjects now had a basis for comparison. They could compare this "other" version to the one they had learned on. Prior to using the "other" system, most of the subjects had not used a computer-based ICAT simulation. Having seen both systems, they had a basis for making comparisons.

The first thing to consider in these results is the number of questions that are within our 0.05 alpha level. Of the 8 questions, 5 are within the level. The overall pattern is the same as the first set of questions -- the participants rate the MSRT-L version more highly than the MSRT-C version. The difference is scale. The extent of the difference is greater. The five specific questions are -- 1) Ability to interact with knobs, switches, etc.; 2) Ability to go from screen to screen; 3) Information provided by coach is sufficient to solve each step of the procedure; 4) As a training device, how enjoyable did you find the program; 5) Overall rating of the program. Two of the other questions that had a big difference, but not a statistical difference were -- 1) How helpful was the text of the "red" error messages?; and, 2) How confident are you that you could start up and use a MSRT stand alone? Clearly, the subjects preferred the MSRT-L version.

As was mentioned when describing the methodology described here, it was thought that people would "bind" with the system that they originally learned from. For MSRT-C group this is somewhat true. If you look at the overall rating of the programs the MSRT-C group rates MSRT-C an average of 3.46 and MSRT-L 3.23. They seem to be equally confident in the two versions rating them both an average of 3.00. However, on every other aspect, the MSRT-C group rates the MSRT-L version higher than "their" version. We use the plural possessive form for the MSRT-C group's relationship to MSRT-C because of the preconceived idea that the subjects would have developed an allegiance to the system they spent all that time on. To some degree, this is true for the MSRT-C group. However, they rate MSRT-L better on every aspect of the system (the first six questions), so this binding is not that strong. If there is a binding that occurs, it is clearly demonstrated in the MSRT-L group's responses.

On virtually every aspect of the questionnaire, the MSRT-L group rated the MSRT-C version lower than "their" version. The range of difference are from a low of 0.64 for how realistic the program was to a high of 1.82 for how easy it is to go from screen to screen. Clearly, the MSRT-L group did not like the MSRT-C version. In the previous discussion, the MSRT-C group had a certain attachment to the MSRT-C version as indicated in their overall rating, but after that they indicated that they preferred all the attributes of the MSRT-L version. Apparently, the magnitude of the difference between the two systems was great enough to overcome any sentimental attachments to the system they had learned on.

In the questionnaire, along with rating the systems, subjects were also asked to comment on what they liked, disliked, changes to the systems, confusing aspects and general comments. (Appendix). The evaluation of the systems "learned on" first were similar, with most subjects enjoying this method of instruction to include the use of the coach and the realistic presentation of the equipment. Furthermore, with the exception of the slowness of the ALM environment, a majority of the subjects reported no need for changes to these environments and found the environments to be an enjoyable means of training.

However, when asked to evaluate the "other system" differences were noted in that most subjects disliked the ALM system; whereas those whose "other system" was Linkway found it to be quick and easy. Additionally, those who used Linkway as the "other system" were more likely to report that they disliked nothing in the program and found no need for changes than those who used the ALM environment as the "other system". Moreover, all the subjects who initially learned on Linkway preferred and found it easier, while only 6 of the 13 who learned on ALM preferred it and 5 found it to be easier.

CONCLUSION

The results of this project all tend to support the idea that off-the-shelf hypermedia packages can be used to streamline the development time for ICAT systems. The functionality comparison indicated some initial problems with re-purposing LinkWay for ICAT development, but most of these obstacles have been overcome. In the end, the MSRT-L version had all of the functionality that the MSRT-C version had, if not more. A second consideration is whether the off-the-shelf version is as effective as building the system within a programming language. The answer is that not only is it as effective, but perhaps more effective. If it is more effective, it may be due to the fact

that it is easier to use. All of our additional performance data suggest that the system is easier to use. Finally, the question of whether there may be some affective reason not to use an off-the-shelf package has been put to rest by the evidence presented here. In summary, the off-the-shelf version was more effective, easier to use, the subjects preferred to use it, and it took less than half the time to build it. We would recommend that people considering the development of an ICAT system ought to consider building it in an off-the-shelf hypermedia system.

Other systems, other content areas, and other needs might not permit the adoption of the decision to use an off-the-shelf system, but there is more to consider. Towards the end of this project, we decided that since we had implemented the system in LinkWay Live, that we might try building a multimedia version of the tutor. In not much time (a couple of weeks), we had a version that would play a movie with narration on a screen that described each procedural step in a text format when the user clicks on the coach button. This multimedia version would not have been considered if the system had been developed in only in C. In addition, we built a gaming environment that required the learner to complete a procedure without error, without using coach, and within a time limit in order to transfer from one army base to another (from Fort Industrial Waste to Fort Fantasy for example). Again, these changes were quite simple to implement and may have taken a total of two weeks to have a bug free version working.

There are two issues that ought to be discussed before closing this paper. First, many people will think that the off-the-shelf system clearly is the way to go and could have told me before I began. Unfortunately, there was no clear evidence that ITS type systems could be developed and could be as effective as systems that are built in more flexible environments like C. Now there is. Second, some people might suggest that the differences that we found between these systems could have been removed if the MSRT-C program was designed exactly like the MSRT-L program. This is true, but that would have added more time to the development of that system and it still would take longer even if the system had similar programming implementations. The implementations that were programmed in the C architecture could be revised and a system could be build that does exactly what LinkWay Live does, but why would anyone want to do that? It has already been done.

We should add a word of warning about time estimates for this development. The programmer who worked on the MSRT-C version took over the development process when the data collection began. We brought in a Subject Matter Expert (SME) and added two more procedures. This process included rewriting the program so that sounds were correct and lights flashed, when on, and went out as appropriate as the learner interacted with the system. All totaled, they worked on the system an additional 250 hours to make it bug free and subject matter correct and complete. The program now would take about 5 hours for someone to achieve mastery. Combining the 75 hours of initial effort with this 250 hours, the total time was 325 hours or 65 hours of development for each hour of instruction. However, a great deal of time could have been saved if the SME had worked on the project during the design phase. It is much easier to implement a design than it is to make large scale changes to an already implemented program. Nonetheless, 65 hours is still an improvement over the 800 hours some suggest it takes to make good quality ICAT systems.

REFERENCES

Anderson, J.R. (1988). The expert model. In M. Polson & J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 21-53). Hillsdale, NJ: Erlbaum.

Anderson, J.R. (1989). The analogical origins of errors in problem solving. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon* (pp. 343-372). Hillsdale, NJ: Erlbaum.

Merrill, M.D., Li, Z., & Jones, M.K. (1990). Limitations of first generation instructional design. *Educational Technology, 30*(1), 7-11.

Jonassen, D., & Wang, S. (1991). *Building an intelligent tutor from hypertext and expert systems.* Paper presented at the annual meeting of the Association of Educational Communications and Technology. Orlando, FL.

Orey, M.A., & Burton, J.K. (1989/90). POSIT: Process oriented subtraction interface for substraction. *Journal of Artificial Intelligence in Education, 1*(2), 77-104.

Orey, M., & Nelson, W. (in press). Development principles for intelligent tutoring systems: Integrating cognitive theory into the development of computer-based instruction. *Educational Technology Research and Development.*

Orey, M.A., Park, J.S., Chanlin, L.J., Jih, H., Gillis, P.D., Legree, P.J., & Sanders, M.G. (1992). High bandwidth diagnosis within the framework of a microcomputer-based intelligent tutoring system. *Journal of Artificial Intelligence in Education, 3*(1), 63-80.

Rosenburg, R. (1987). A critical analysis of research on intelligent tutoring systems. *Educational Technology, 27*(11), 7-13.

Towne, D. (1991). *Instruction in a simulation environment: Opportunities and issues.* Paper presented at the annual meeting of the Intelligent Computer Aided Training conference. Houston, TX.

APPENDIX

LEARNED ON ALM*	
LIKED	
interesting way of learning	4
directions the coach gave	2
graphics	2
improves one's ability to operate a computer	1
working at own pace	1
easy to understand	1
realistic	1
nothing	1
this training aid works	1
DISLIKED	
wait between screens	5
some of the instructions were confusing	2
nothing	1
F6 action couldn't be displayed on every screen	1
prefer hands on	1
CHANGES	
none	7
make it faster	2
better instructions when first start	2
reword coach instructions	1
voice messages	1
put F-6 ACTION button on every screen	1
CONFUSION	
Go To F-6 ACTIONS	2
coach	2
environment difficult at first	2
"having to jump around on the equipment"	1
COMMENTS	
good training device	2
no explanation of the purpose of this program	1
prefer hands on instruction	1
LEARNED ON LINKWAY*	
LIKED	
easier to learn with computers than with instructors	4
coach	3
equipment realistic	2
fine program	1
easy to understand	1

the sequence of instructions	1
DISLIKED	
nothing	8
red error messages	2
checking all of the lights	1
switching screens	1
preferred hands on	1
had to follow the sequence	1
CHANGES	
none	7
timing between screens	1
red error messages	1
put KYK-13 on same screen as DSVT	1
make the program faster	1
let the trainee know his/her mistakes	1
switch to turbo ball	1
none	10
beginning needs more instructions	1
COMMENTS	
enjoyable	4
ALM AS OTHER SYSTEM*	
LIKED	
nothing	10
similar to first	1
coach	1
liked everything	1
DISLIKED	
wait between screens	7
didn't understand directions	6
outlined boxes confusing	1
everything	1
location of instructions	1
CHANGES	
make it faster	2
throw it away	2
make it user friendly	1
get trained by someone else	1
make it more like the first program	1
getting "fill cap off" was difficult to find	1
better information about directions	1
better graphics	1
CONFUSION	
some directions	4
nothing	3
map	1
coach complicated and indirect	1
buttons hidden	1
screen layout	1

everything	1
COMMENTS	
program was bad	3
prefer Linkway	3
too frustrating	1
both programs were excellent	1
LINKWAY AS OTHER SYSTEM*	
LIKED	
easy	6
quick	4
coach	1
fun	1
DISLIKED	
nothing	6
seemed like there were more steps in this program	1
hot spots not outlined	1
didn't like the way you had to find things	1
there was too much information in coach	1
equipment not precise	1
different from ALM	1
CHANGES	
none	7
make coach more understandable	2
outline hot spots	1
make the knobs on the DSVT more precise	1
CONFUSION	
none	6
coach difficult	2
some places used lights and other places used switches	1
beginning needs more instructions	1
the computer	1
the beginning about the antenna	1
COMMENTS	
good program	1
ideal teaching device	1

* Not all subjects chose to respond; some gave more than one response.

Michael Orey is an assistant professor at the University of Georgia in Instructional Technology. He is interested in emerging technologies for learning.

Ann Trent is a consortium student with Augusta College and the Army Research Institute and is interested in experimental psychology.

James Young is a consortium student with Augusta College and the Army Research Institute and is interested in Computer Science and Intelligent Tutoring Systems.

Michael Sanders is a research psychologist and is the field unit chief at the Army Research Institute at Fort Gordon. He is interesting in training innovation.

On the Construction of a Knowledge-Based Intelligent Tutoring System (KB-ITS)

P. Ahmed, A. Al-Dhelan and A. Shah
King Saud Universit
Department of Computer Science
P. O. Box 51178/Coll. of Comp. & Info.Science
Riyadh, SA 11543
96610114675423
F60C033@SAKSU00.Bitnet

The dedicated Computer-Based Training (CBT) systems are commercially available for teaching and training purposes. These systems are not constructed by using formal data modeling and knowledge representation techniques rather they are developed by traditional language-based approaches. One of the major limitations of these systems is that they are architecturally closed and rigid, and amendments in the source program is the only viable option for incorporating new concepts or functions in the systems. Furthermore, these CBT systems cannot evolve with time; learn and deliver new concepts and fully and effectively utilize technological advances without modifying source programs. Therefore, an CBT application development environment other than traditional environment is required. To provide such an application development environment a project is in progress to construct an intelligent shell.

The shell takes the advantages of the recent advances in: (a) multimedia concepts and techniques developed for effectively managing and manipulating graphics, high-resolution images, motion video, high-fidelity audio; (b) database methodologies for handling the well-formatted textual data known as record-base and (c) knowledgebase methodologies for adding the pertinent knowledge manipulation capabilities. The shell, through its intelligent graphical user interface, allows a user to provide full functional specifications of the application systems which is to be developed and selection of appropriate record-base, image, graphics and audio database, and knowledgebase systems which are commercially available.

Currently, we are exploring the potential applications of the proposed Intelligent shell for the construction of application systems. One of the applications we are involved is the construction of a knowledge-Based

Engineering Intelligent Tutoring Systems

Kimberly C. Warren Bradley A. Goodman
Artificial Intelligence Center[1]
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730-1420
kim@linus.mitre.org
bgoodman@linus.mitre.org

ABSTRACT. We have defined an object-oriented software architecture for Intelligent Tutoring Systems (ITSs)[2] to facilitate the rapid development, testing, and fielding of ITSs. This software architecture partitions the functionality of the ITS into a collection of software components with well-defined interfaces and execution concept. The architecture was designed to isolate advanced technology components, partition domain dependencies, take advantage of the increased availability of commercial software packages, and reduce the risks involved in acquiring ITSs. A key component of the architecture, the *Executive*, is a publish and subscribe message handling component that coordinates all communication between ITS components.

We implemented critical components of the architecture as a simple hypermedia training system, the Macintosh Maintenance Training System (MMTS). The domain for the prototype training system is the maintenance of Apple Macintosh IIcx computers.

This project has shown that the use of a modular software architecture for the development of ITSs, and complex integrated artificial intelligence applications in general, has several important benefits. Its use allows for rapid development, incremental integration and testing of components, and a more maintainable, extensible, and reusable end-product. Even more evident was the benefit of an Executive component that facilitated the integration of commercial software packages with custom developed software in a well-defined manner.

INTRODUCTION

This paper describes the design and implementation of a generic architecture for Intelligent Tutoring Systems (ITS). The architecture uses object-oriented technology to provide the ITS developer with a solid, reusable framework for the development and testing of particular functional modules of an ITS and the ability to easily integrate COTS products with custom-developed software.

Prior to the selection and use of Commercial Off-the-Shelf (COTS) products, custom-developed functionality, and non-developmental software, software system designers benefit from a systematic definition of functions and subsequent interfaces that will be required of them, i.e., the definition of a software architecture. In the case of an ITS to be run on a single CPU with media-providing peripherals, a software architecture partitions the functionality of the ITS into a series of software components with well-defined interfaces and execution concept. The development of a robust software architecture can improve the overall software quality of a system.

---

1 This work was supported by the Computer-Based Training Specialty group as a MITRE technical overhead project.

BACKGROUND

We identified the need for a reusable and extensible software architecture to be used across multiple ITS for various reasons. For example, a considerable savings in the development cost of multiple tutors will result if common software components are identified, developed once or assigned to an existing COTS or non-developmental software (NDS) product, and incorporated into all tutors. Large scale software reuse also means that the tutors can have a similar "feel" to the student and would be less expensive to maintain. The use of appropriate COTS products also provides the users/developers with emerging COTS interface look and feel standards. An extensible architecture attempts to achieve goals including making it possible to incorporate changes in technology, in such areas as multimedia interfaces, student modeling, expert modeling, and instructional strategy, without having to re-implement the entire system. We felt that building tutors on a generic framework of loosely-coupled software components had the best chance of achieving these goals. We developed such an architecture and specified it in terms of functional components, detailing the function, interfaces, and suggested behavior of each [1].

Other ITS researchers have defined what they call an "architecture" for an ITS to address issues of easing the cost associated with developing ITS. Most of this work relies on their particular definition of the term "architecture." Work done by John Anderson (with Boyle and Yost [3], Reiser and Farrell [4], and Corbett [5]) on the Geometry, Lisp and ACT tutors is likely the most advanced application of a reusable architecture to date. However, as Yazdani observed (in [6]), these tutors suffer from limited student modeling and instructional design strategies. These strategies are prescribed (even required) by the "architecture" and do not allow for modifications that would allow them to better deal with each individual domain. Also, none of these implementations used COTS, they relied upon custom developed software.

NASA/Johnson Space Center has followed a different path by making an attempt at developing a "generic architecture" for ITS in the form of a set of integrated tools [7]. Their general-purpose ITS design uses a COTS knowledge engineering tool, CLIPS, for encoding and manipulating production rules and a blackboard system component, custom-developed, for coordinating communication between modules. The developer of an ITS can build on top of the kernel provided by the NASA general architecture to reduce development time and to provide some consistency across ITS. However, their selection of specific tools constrain the ITS designers/developers to those methodologies supported by those tools. For example, the particular COTS knowledge engineering tool that they selected requires knowledge to be encoded in terms of production rules. Additional design and architecture changes would be required if the ITS designer required an alternative formalism, and/or alternative COTS tool, for the representation of expert, student, or instructional knowledge. Also, most of the ITS must be re-built for each new domain.

In contrast to other ITS architecture efforts, we proposed that general interfaces to the required functionality of an ITS could be specified and implemented. Components of an ITS such as an expert model, student model, and instructional environment could be specified in terms of their required functionality and that, indeed, when knowledge representation and implementation details are set aside, standard interfaces to the specified functionality emerge. Such encapsulation of functionality allows particular components to be "unplugged" and replaced by alternative components conforming to the same interfaces. This proves to be a very powerful capability in the development of ITS.

Although knowledge representation schema differ widely in ITS research into expert modeling, student modeling, and instructional design, the conditions under which such functional components are activated, i.e., their input, and the responses that are expected from them, i.e., their output remain similar across knowledge representation and manipulation methodologies. For example, expert modeling functionality receives input that includes student action/statements and will generate output that includes expert evaluations of those actions as part of its function as the expert problem solver in an ITS. In the case of student modeling, "higher" level measurement of the student's ability (e.g., "higher" than a simple recounting of student actions during problem solving) is expected to be available from the student modeling functional component during the lesson. For example, student modeling functionality usually

receives student action/statements as input and generates some measurement of overall student ability as output. Knowledge representation, manipulation, and presentation differ widely in ITS research into instructional environments. However, the conditions under which an Instructional Environment operates and the responses that are expected from it, e.g., student action/statements and individualized responses/feedback to the student, respectively, remain similar across methodologies. These similarities allow general interfaces to these functional components in an ITS to be specified [2]. With these general interfaces, we have opened the door to the use of appropriate COTS products to implement the specified functionality.

APPROACH

The architecture of a digital system consists of the hardware and software components, their interfaces, and the execution control that underlies system processing [8]. A software architecture should partition the functionality of ITS into a collection of software components with well-defined interfaces and execution concept. A software architecture for an ITS should be designed (i.e., ITS functionality should be partitioned) to meet the following goals:

a. Technology insertion and isolation: The architecture should provide logical functional separation to allow the insertion of relevant new technologies as they become available and provide a way of isolating dependencies on existing technologies.
b. Ease of acquisition: The architecture should reduce the overall development and maintenance costs associated with designing and building multiple ITS.
c. Domain dependency isolation: The architecture should isolate domain dependencies of the ITS. While this isolation will not be complete, it should at least define and localize the effects of any domain changes on the remainder of the ITS.
d. Isolation of instructional design strategies: The architecture should allow differing instructional strategies to be implemented and tested, with minimal impact on the rest of the ITS.

In order to achieve the above goals, we specified and began implementing a generic ITS architecture in terms of an object-oriented design for the functional components of an ITS. The highest level objects in the ITS architecture correspond to the software components that support the individualized training environment. Each of these components is assumed to be an independent entity that can act on its own accord. As an object-oriented system, the software components communicate with each other by sending and receiving messages that either contain control information that affect components' operation (e.g., starting or stopping a simulation) or queries for data about some aspect of the course (e.g., state of a simulated system component.)

In one instantiation of our architecture, each ITS can consist of the ten software components shown in figure 1. (A *software component* is defined to be an independent functional entity that performs a logical set of functions. Each software component communicates with other software components via well-defined interfaces. One can come up with many different partitions of the functionality found in an ITS as the basis for an architecture definition; however, the resulting architecture should follow the guidelines and goals stated above for modularity of advanced technological (i.e., Artificially Intelligent), instructional strategic, COTS integration, and domain-dependent components. If this is the case, it should adapt to changes in implementation and domain gracefully, i.e., without requiring the re-coding of all components.)

The Executive is the core of the ITS architecture. It is the software component that implements the execution control that underlies system processing and fosters ease of COTS integration. The Executive provides one of the basic functionalities that is characteristic of object-oriented architectures, message passing between encapsulated processing entities.

The functionality of an ITS student interface, i.e., process student actions in order to generate individualized feedback, has been refined into multiple components in our architecture: Multimedia Interface, Presentation Controller, Explanation Generator, and Instructional Environment. This refinement was meant to leave the Instructional Environment to deal with high level instructional design strategy processing. With the availability of COTS tools for multimedia graphical user interface development and delivery, we isolated the Multimedia Interface to allow for the use of COTS. The Multimedia Interface provides the physical means by which the student interacts with the Instructional Environment. Specifically, the Multimedia Interface processes multimodal input from the student, via such devices as keyboard and mouse, and generates and displays multimedia presentations including text, graphics, sound, and video. Input in terms of input events (mouse clicks, key input) is passed on to the Presentation Controller for further refinement. Output requests in terms that the particular multimedia tools can interpret are generated by the Presentation Controller.

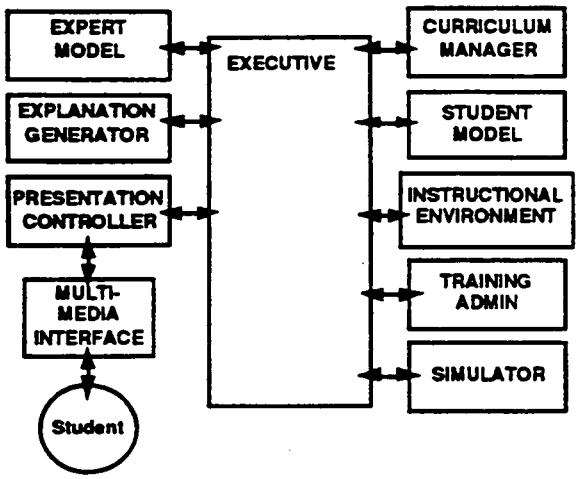


Figure 1 Components of the ITS Architecture

Depending upon the tools used to implement the Multimedia Interface, a particular "language" will be required to control presentation of information. The Presentation Controller organizes and sequences the information to be presented to the student. Given an Explanation Generator script containing several display directives such as "AT THE SAME TIME: Display video indices 1230 to 1259 and text 'This is how you ...' ," the Presentation Controller synchronizes the execution of requests for their physical display by the Multimedia Interface. The Presentation Controller is also responsible for the translation of the physical user interactions (e.g., mouse movement, keyboard input, etc.) supplied by the Multimedia Interface into actions on the problem domain elements for the Instructional Environment.

Considerable research and development is ongoing in the area of tailored explanations and multimedia explanation generation (see [9,10] for examples). For this reason, we isolated the functionality that generates the content and sequence of explanations given to the student in the Explanation Generation component. The Explanation Generator, in conjunction with the Instructional Environment, is the part of the training system responsible for generating individualized advice. The output of the Explanation Generator is a script denoting information to be displayed in terms of media items organized in terms of temporal and spatial relationships.

While the Instructional Environment and its related components control the presentation of material to the student within a given exercise, the Curriculum Manager governs problem selection with respect to the entire course or group of exercises. Like the other modules, the Curriculum Manager uses information about the student's current problem solving skills and knowledge to generate appropriate interactions with the student. This information is compiled and maintained by the Student Model. The Student Model gathers data from the student's problem solving actions. It should map these actions into a representation that summarizes and abstracts relevant student performance measurements.

A Simulator component was broken out of the standard ITS Expert Model component. This isolates as much of the domain dependencies associated with a tutor's domain simulation and allows for the use of COTS simulation and expert modeling tools. The Expert Model contains knowledge that includes the general methods or procedures that an expert would use to solve problems in the domain. It is possible that the expert problem solving knowledge used by one ITS, or the COTS tools used to implement it, could fairly easily carry over to additional domains. This is not the case for tutor domain simulations.

Finally, the Training Administrator collects and stores measurements of ITS usage and students' problem solving skills. The maintainers of the software may use standard COTS analysis tools, such as spreadsheets, to evaluate the usage information.

THE MACINTOSH MAINTENANCE TRAINING SYSTEM

We implemented critical components of the architecture as a simple training system, the Macintosh Maintenance Training System (MMTS). The domain for the prototype training system is the maintenance of Apple Macintosh[3] IIcx computers. The initial capability for the implementation described in this paper came from a multimedia explanation generation component for an intelligent help system developed by Bradley Goodman [11]. We encapsulated this system's functionality into separate software components in accordance with our architecture. We also augmented this system with student modeling capabilities that were lacking in the original Macintosh repair system. Our development approach made use of commercially-available software products and object-oriented technology. The implementation used the DOS/Microsoft Windows operating system running on an IBM PC-compatible hardware platform. We used a combination of C, C++ and Asymetrix Toolbook to develop our prototype.

The MMTS prototype is made up of four software components that provide the initial, critical functionality required by an ITS for Macintosh maintenance: Executive, Instructional Environment, Expert Model, and Student Model. The MMTS uses separate executables for the software components (implemented using different languages and COTS products) that make up the tutoring system. The following paragraphs describe the MMTS design and implementation in some detail.

The Executive has responsibility for the dynamic coordination of all message passing between software components in the object-oriented architecture. A *message* is an object that has attributes (fields) that may include: message class, contents, priority, sender, and time stamp.

In keeping with the strategy to use object-oriented techniques to maintain a loose coupling of components in the ITS, the Executive employs a subscription-based message handling approach wherein software components request message handler permission to publish messages on selected message classes. A message class is, for example, "student action" which specifies messages containing descriptions of student actions during problem solving with the tutor. Conversely, a software component can subscribe to the various message classes from which it wants to receive information.

Software components publish and subscribe to messages via I/O ports created for one or more message classes. The use of I/O ports for the sending and receiving of messages allow software components to easily suspend multiple message classes with a single call to suspend a port. Messages sent by a software component through an I/O port for publication are broadcast to all software components that subscribe to the message class. Message sending can be accomplished in any of the following ways:[4] asynchronous, synchronous, or synchronously timed. The flexibility allowed by these types of sends allows the ITS developer to control processing at many different levels. For example, if a particular component of the system wants to interrupt another component's processing, an asynchronous send to the relevant component(s) would be required. Message receipt can be accomplished in any one of the following ways:[5] unconditionally with wait, typed, timed, or typed and timed. These types of receipt allow the various components of the system to selectively process requests from other components.

---

3 Apple and Macintosh are trademarks of Apple Computer, Inc.

4 Currently, we are only handling the asynchronous passing of messages between processes under Microsoft Windows. We believe that, of the three types, this is the most complex to implement given a focus on incorporating COTS software into any implementation of the architecture.

5 Currently, we are only handling the unconditional receipt of messages from within the Executive.

The Executive was implemented in C as a Dynamic Link Library[6]. It functions as a layer of message publication and subscription handling on top of Microsoft Windows. This layer maintains tables of software components, messages that they subscribe to, and messages that they will publish. Software components are provided with the various send and receive capabilities described above, with the Executive handling all of the Microsoft Windows communication issues. Software components that wish to communicate need only link with the Executive and publish on and subscribe to appropriate messages. The availability of the Executive defining a standard interface between COTS and custom developed products simplified their integration to one of assuring communication with the Executive. Systems that employ multiple COTS products communicating in complex conversations require the developer to implement multiple interfaces between COTS and custom developed software, one per communicating pair of system components. When the Executive is used, however, the interface implementation is reduced to the implementation of one interface to the Executive for each COTS and custom developed component. Finally, the use of Windows allowed us to take advantage of non-preemptive multi-tasking to control the order in which messages were both sent to and received from applications linked to the Executive.

The graphical user interface of the MMTS, the Instructional Environment, is shown in figure 2. This interface was built using Multimedia Toolbook[7] . The scrolling menus on the left enumerate the actions that the student can take. The buttons down the middle of the screen provide the student with access to any of the three views of the Macintosh IIcx. The buttons on the right provide the student with textual, graphical, and video help during problem solving. The scrolling window at the bottom of the screen shows the plan that the student is creating during problem solving and allows him/her to select any of the actions in the plan for video display. This particular display shows the results of appropriate student actions in the MMTS as the listing of the English text in the "Actions to be Performed:" window.

An Expert Model, written in C++, models the procedures required to do Macintosh repairs. These procedures were implemented as a finite state machine that encodes the rules for repair delineated in the Macintosh Maintenance Manuals. During the course of problem solving, the Expert Model evaluates student problem solving actions and responds to queries for appropriate (expert) actions in the context of the problem being solved.

A Student Model, written in C, models the student's problem solving by tracking the errors and their gravity during a problem solving session. During the course of problem solving, the Student Model may require the Instructional Environment to display additional, unsolicited advice if the students errors are of a type or frequency that the Student Model would find inappropriate.

The Instructional Environment coordinates instructional interaction with the student. For example, the Instructional Environment *publishes on* the "student action" class of messages. Conversely, the Expert Model and Student Model components *subscribe to* the "student action" class of messages. When a student completes a maintenance action on the Macintosh IIcx (e.g., clicking on the video board graphic and the menu action "pull up") via the Instructional Environment, the action is packaged into an instance of the "student action" message class and broadcast to all components that subscribe to the class via the Executive message handler. In this case, the Executive will route the message to both the Expert Model and the Student Model for evaluation and retention, respectively.

---

6 A DLL allows applications to access functions in a library without having to include the library code in the source code of the application. This also allows multiple applications to access a single copy of the library.

7 Toolbook is a registered trademark of the Asymetrix Corporation.

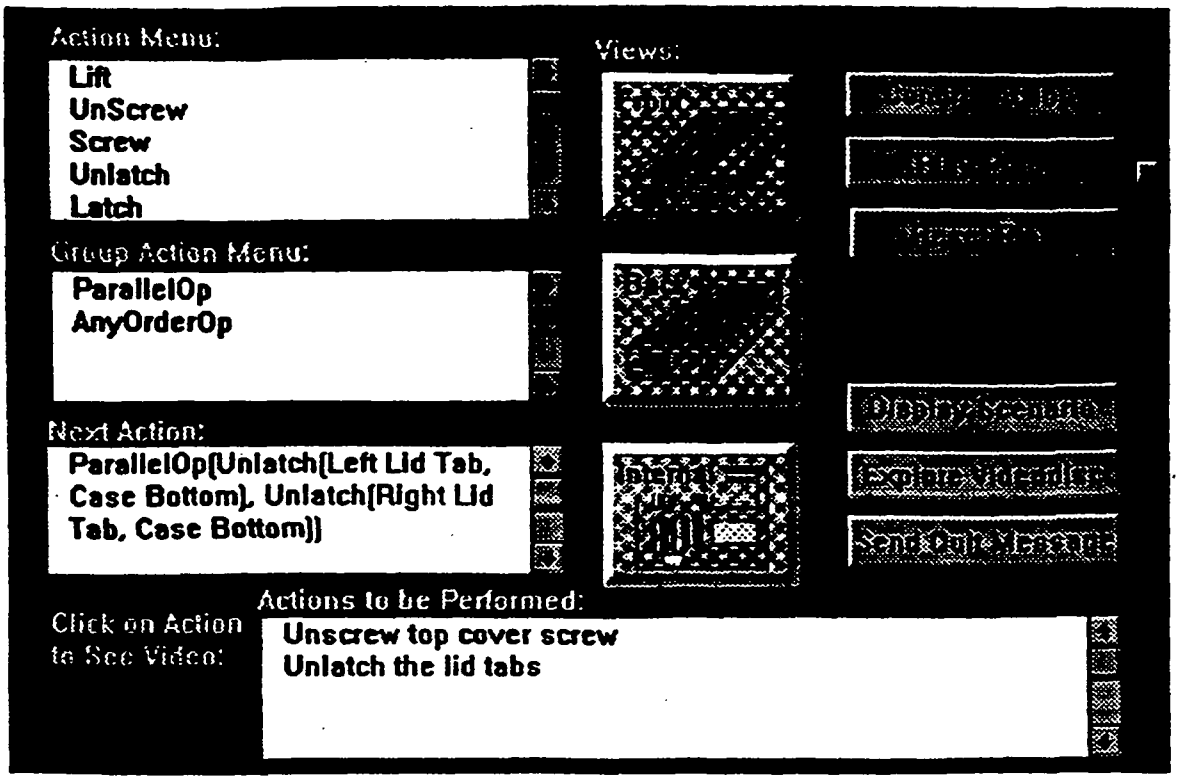


Figure 2 MMTS user interface

The Expert Model will evaluate the action with respect to an expert's problem solution. This component will then create and broadcast a message of class "student action evaluation" containing the evaluation which will be received by both the Instructional Environment and the Student Model. The Instructional Environment will process the evaluation with respect to its instructional design strategy (e.g., whether to give immediate feedback, the type of feedback, etc.) For example, the Instructional Environment may cause the creation of a multimedia display of a video board being removed with text overlayed describing the action.

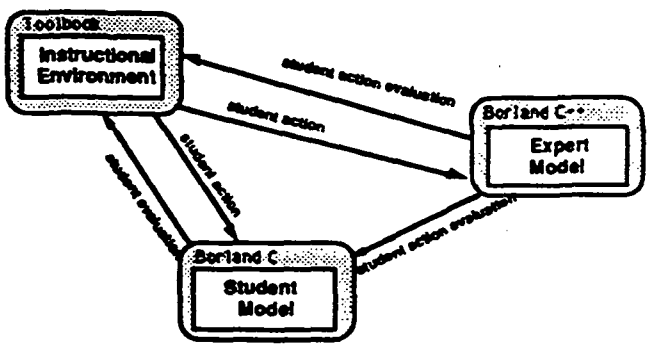


Figure 3 Sample Message Passing

The Student Model will store the action, as well as any Expert Model evaluation, and make any inferences about the state of the student's understanding that are possible. If it is the case that the student's understanding has dropped below a particular threshold, the Student Model will broadcast a "student evaluation" message which will be picked up by the Instructional Environment. At this time, the Instructional Environment may provide the student with unsolicited advice, depending upon the instructional strategy implemented. Figure 3 shows the message passing between software components that the Executive facilitates in this particular example.

CONCLUSIONS

The work done on this project to date provides evidence that the use of a modular software architecture for the development of ITS has several important benefits. Its use allowed for rapid development,

incremental integration and testing of components, ease of COTS integration, and a more maintainable, extensible, and reusable end-product. In particular, the Executive component facilitated the integration of both COTS and custom developed software in a well-defined manner. Standard integration tasks require the specification and development of multiple interfaces, one per communicating pair of components. Software components, be they COTS or custom developed, that communicate via the Executive message handler require only a single interface; that is, they need only be integrated with the Executive. Acting as an abstraction of the message passing/process communication functionality provided by the operating system, the Executive is able to take quite a bit of the integration burden away from the developers of the system.

The use of COTS products allowed us to rapidly develop an initial training capability. The freedom to choose multiple COTS tools in combination with more standard programming languages gave us the ability to implement fairly complex algorithms in a short period of time. Many problems arise from the use of COTS products (alone) in a rapid prototyping situation due to the limitations of any one particular COTS product. Other problems arise from the range of different integration tasks that developers are required to perform when combining the functionality of multiple COTS products. The availability of the Executive defining a standard interface between COTS and non-COTS products simplified the integration to one of assuring communication with the Executive.

With the savings in development afforded by the use of COTS, the increased supply of powerful COTS software, and the increased demand for its use in government and private sector development efforts, issues of maintainability of software systems that utilize these products are brought to the forefront. Prior to the selection and use of COTS products, system designers benefit from a systematic definition of functions and subsequent interfaces that will be required of them. The architecture specified for ITS in this paper was designed to take advantage of the increased availability of COTS software. The partitioning of components took into account COTS trends. For example, the separation of the Multimedia Interface from the Instructional Environment allows developers/designers to take advantage of the multitude of available COTS drivers for graphical user interface providing hardware. The separation of the simulation functionality from the Expert Model provides one with the ability to either develop a simulation given a COTS tool or the use of an existing simulation of the training domain.

The Executive emerged from this activity as a tool for the integration of software components for any application, not limited to ITS, and a software engineering methodology enforcement mechanism to be used during design and development of a system. The use of the Executive enforces "good" software engineering practices by requiring all designers/developers to conform to a standard set of interfaces defined in advance of full-scale development. It was clearly efficient to allow each of the developers to ignore the implementation details of components for which he/she was not responsible. With the Executive, the interface and communication issues must be addressed and resolved "up-front".

The generic ITS architecture and its Executive are not magic. As a matter of fact, the hard ITS design and development work is yet to be done. This work includes the advanced technology design and development issues involved in creating the various "intelligent" modules of the training system. However, with the use of such an architecture for the implementation of ITS, and integrated COTS systems in general, we feel that the users and developers will benefit. The addition of techniques, products, and technologies has been anticipated. The re-use of complex components has been enabled.

Acknowledgments

The author would like to acknowledge Glenn Karcher for his extensive contributions to the design and development of the Executive functionality, George Huff and Steve Litvintchouk for their extensive Software Engineering advice, and Michael Sayko for his contributions to the design of the Intelligent Tutoring System architecture.

References

[1] Warren, K. C., Karcher, G. W., Sayko, M. S., Goodman, B. A., "The ITS Software Engineering Reference Guide," (draft) MITRE Working Paper, WP-92B0000085, 1992a.

[2] Warren, K. C., Huff, G. A., Michlowitz, E. M., "A Generic Architecture for Intelligent Tutoring Systems," MITRE Technical Paper, MTR 92B0000200, 1992b.

[3] Anderson, J., Boyle, C. F., and Yost, G., "The Geometry Tutor," in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1985

[4] Reiser, B. J., Anderson, J. R., and Farrell, R. G., "Dynamic Student Modeling in an Intelligent Tutor for Lisp Programming", in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1985

[5] Corbett, A. T., and Anderson, J. R., "Student Modeling and Mastery Learning in a Computer-Based Programming Tutor" in *Intelligent Tutoring Systems, Second International Conference, ITS '92*, Springer-Verlag, 1992.

[6] Yazdani, M., " Intelligent Tutoring Systems: An Overview," in *Intelligence and Education, Volume One*, eds. R. W. Lawler and M. Yazdani, Ablex Publishing, 1987.

[7] Computer Sciences Corporation, "Programmer's Guide to the General Architecture of Intelligent Computer-Aided Training Systems," Report CSC-A000154, December 1991.

[8] Horowitz, B. M., MITRE, "The Importance of Architecture in DOD Software," MITRE Technical Paper, M91-35, July 1991.

[9] Feiner, S. K. and McKeown, K. R., "Generating Coordinated Multimedia Explanations" in *Proceedings of the 6th International Conference on Artificial Intelligence Applications*, Santa Barbara, CA, 1990.

[10, 11] Goodman, B. A., "Multimedia Explanations for Intelligent Training Systems," presented at Conference on Intelligent Computer-Aided Training, Houston, TX, 1991.

An Idealized Computer-Based Patient Record For Teaching Medical Diagnosis And Care Planning

Rodger Marion, Bruce R. Niebuhr, Chris Renten, and John Bernstein[1]
School of Allied Health Sciences
The University of Texas Medical Branch
Galveston, TX 77555-1028

INTRODUCTION

The Health Information System Simulation (HISS) Project is developing and using a series of windows-based computer applications to teach medical diagnosis and care planning to students at the University of Texas Medical Branch's School of Allied Health Sciences (SAHS).[2] We have established an interdisciplinary training program for students in Health Information Management (HIM), Occupational Therapy (OT), Medical Technology (MT), and Physician Assistant Studies (PAS).

A major goal of the project is to teach students problem solving skills in diagnosis and treatment planning. Using a simulated health information system, students learn how to gather and manage patient data, and use it in diagnosis and treatment planning. Students participate in computer-based activities in their professional courses, then use computer systems similar to the classroom simulations in their clinical placements. The project will evaluate the degree to which classroom training using simulations generalizes to actual clinical settings. We expect students will have increased confidence and higher expectations due to the project. The purposes of this paper are to present (a) the software environment developed, and (b) evaluation results of the software's use by students.

OVERVIEW OF HISS SOFTWARE ENVIRONMENT

The simulations center on three primary computer applications, all written using *Asymetrix ToolBook* and include: an idealized computer-based patient record called the Electronic Patient Record (EPR), applications relating to specific domains of health care delivery and special topics, called MainBooks, and a supplementary application, the HelpBook. These applications run on the SAHS local area network (LAN) and operate interdependently, sharing data among themselves.

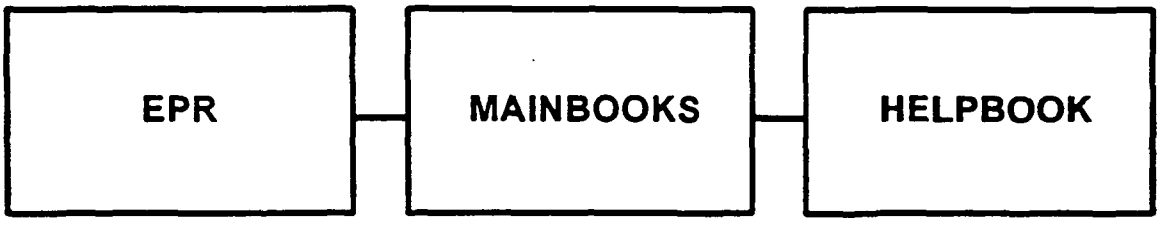


Figure 1. HISS Project primary applications.

A recent survey indicates health care providers use a variety of specialized applications that communicate with each other and across networks to record, transmit, and store patient information effectively and efficiently (Healthcare Informatics and Management Systems Society, 1992). Though *Microsoft Windows* is not a true multitasking environment, it allows easy access to multiple applications, depending on hardware limitations and

[1] *The authors wish to thank Ms. Lynda Baldwin, M.A. and Ms. Teresa Graham, M.S.W. for their assistance as content experts and instructors.*

[2] *Primary funding for the HISS Project comes through Allied Health Special Project grant D37 AH00156, Bureau of Health Professions, Department of Health and Human Services.*

software requirements. The combination of a network with windows-based applications, allows the HISS to provide a training simulation with the flexibility demanded by state-of-the-art health information systems.

Our students access the HISS on forty-five 80386 computers in the SAHS Learning Resource Center (LRC) and other laboratories. This allows students the flexibility to take notes while reviewing a patient record, access reference resources, or other applications. In planning assignments using the HISS, faculty members can employ various combinations of individual study or classroom instructional modes.

The EPR accesses patient database files (created with *Borland dBase III Plus*) and provides a graphical user interface (GUI) for displaying data.

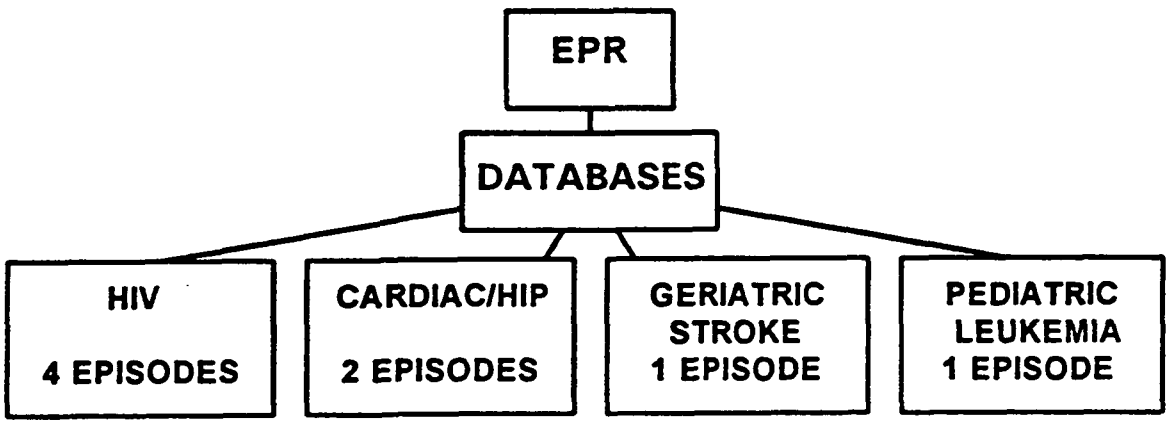


Figure 2. EPR and patient databases.

The EPR provides information on demographic and referral data, complete histories and physicals, laboratory results with normal values, rehabilitation assessments, and graphics of x-rays, EKGs, and CT scans. We have created four patient cases for simulation, based on a variety of demographic, diagnostic, care planning, and instructional criteria. Two patient cases, a Geriatric/Stroke patient and a Pediatric/Leukemia patient, contain single episode inpatient scenarios. An HIV case consists of four episodes, two inpatient and two outpatient, spanning a seven year period, from a negative HIV test through the development of AIDS. A Myocardial Infarction (MI)/Hip Fracture patient involves two episodes with the MI occurring four months before the hip fracture.

The domain specific MainBooks allow students and faculty members to access and interact with the EPR and other HISS applications. Here students login, choose a patient case and episode, and record findings for assignments. The MainBooks record this and other student usage information, such as the duration and number of times a student uses a specific application, or goes to certain pages or topics within an application. MainBooks also define how students use the EPR, according to the MainBook's specific domain and the patient case and episode. Faculty members use the MainBooks to assess student findings and record feedback on student performance. Students can view feedback once an instructor completes their assessment. The domain specific MainBooks are: Primary Care Diagnosis (PCD), Rehabilitation Care Planning (RCP), and Clinical Laboratory Science (CLS).

The special topics MainBooks were designed to focus on a specific element of health care delivery. Our Hand Evaluation Station (HES) introduces students to computerized assessment. The HES is connected to a computerized dynamometer and pinch gage and measures hand and pinch grip strength. HES software automatically records this information in a simulated patient database over the LAN.

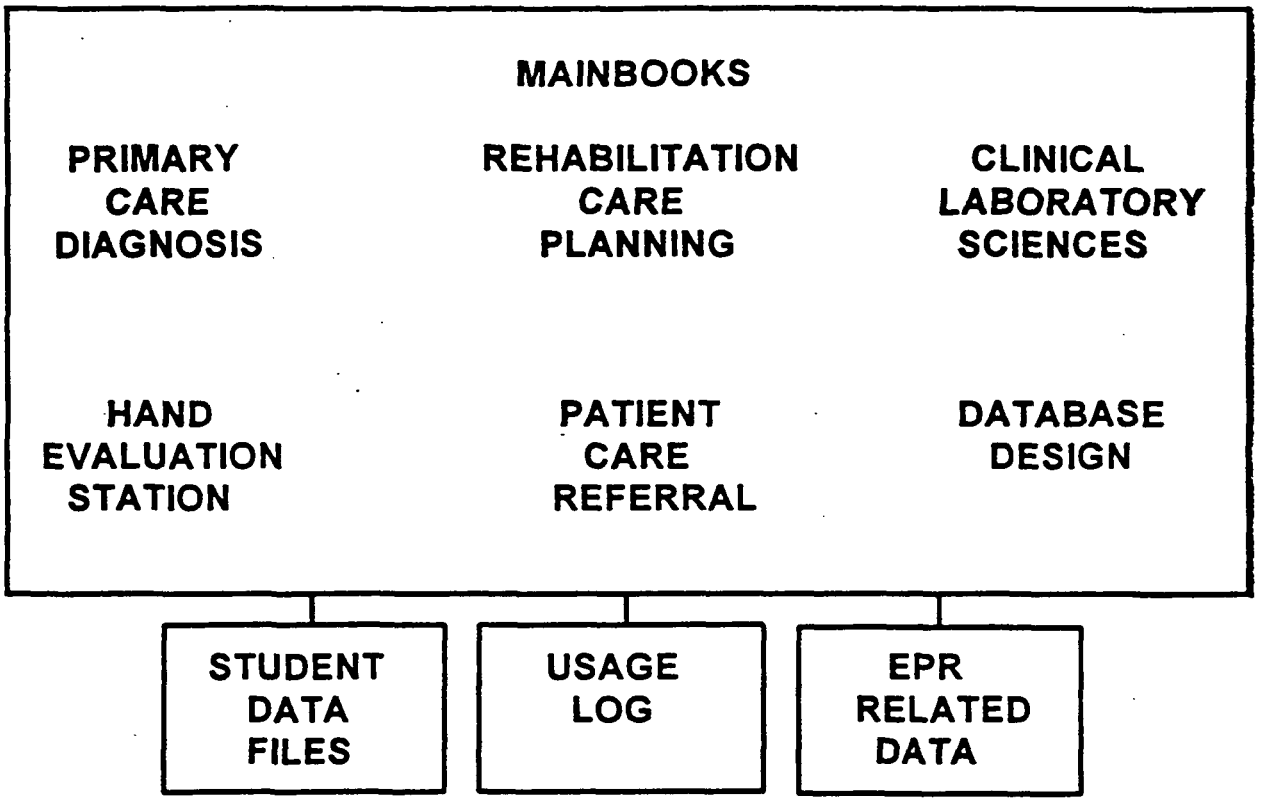


Figure 3. HISS MainBooks and information storage files.

The Patient Care Referral (PCR) MainBook requires students to learn about patient care coordination by using the PCR to refer their simulated patients to colleagues in other disciplines. With the Database Design (DBD) MainBook, HIM students study the underlying structure of the EPR's database using patient records as examples.

Figure 3 illustrates where the MainBooks store information. Student Data Files record each student's assignment work and instructor feedback. The Usage Log records student and instructor logins, usage time , and application use. EPR Related Data contains information concerning patient cases and episodes.

The third primary HISS application, the HelpBook, provides varying types of assistance to HISS users. It contains basic instructions for using the HISS applications, a multidisciplinary glossary of acronyms and abbreviations, supplemental information on related health care topics, access to related tutorials and third-party software applications such as the *Iliad* medical expert system and the *Adaptive Device Locator System*.

QUESTIONS

This is an evaluation design and there were several questions about student usage of the HISS Project applications that we wanted to answer.

1. How long did it take the students to do each assignment?
2. What sections of the Electronic Patient Record (EPR) were used by the students?
3. Is there a correlation between student usage (time, number of sections read, HelpBook usage) and student performance?
4. Where did students have difficulty with the applications?
5. How did the students use the HelpBook?

METHODOLOGY

We studied two classes of OT students during the Spring semester of 1992 and 1993. In 1992, 34 OT students used the Rehabilitation Care Planning (RCP) MainBook, the EPR, and the HelpBook to write a care plan for the Geriatric/Stroke patient. In 1993, 41 OT students wrote two care plans, one for the HIV and another for the Cardiac/Hip patients. All three applications had been revised extensively for the 1993 class.

Usage data was collected by the applications. When the student exited the MainBook, it wrote the following information to a log file; date, login and logout times, student name, application name, number of times student selected from three options: (a) select a patient and run EPR, (b) create/edit a care plan, (c) print a care plan, if the student read the instructor's critique, and a text message consisting of a concatenated list of various errors and user actions. When the student exited from the MainBook, it instructed the EPR and HelpBook to close also. The EPR wrote usage data to a separate log file that included: date, login time, log out time, student name, application name, and a variable list of three letter acronyms for each section of the EPR visited by the student during that session. The HelpBook wrote these variables to a third log file: date, login time, logout time, student name, application name, number of times student visited each section of the HelpBook. The separate files may seem excessive, but the practice simplifies later data retrieval and provides some redundancy in case of accidental file loss. This was valuable since over half the 1993 EPR log file was lost. The redundant data in the RCP log files reduced some consequences of this loss.

Performance data was obtained by having the instructors evaluate each student's care plan according to objective criteria and assign a point value to the student's work. In 1992, the student work was graded on a scale of 0 to 4 with 4 being excellent. In 1993, a scale of 0 to 100 was used with 100 being excellent. One instructor graded the 1992 care plans and the 1993 Cardiac/Hip care plans, and another instructor graded the 1993 HIV care plans. Both instructors were masters of their profession and qualified to evaluate the quality of the student's work. The instructor's collaborated on the criteria, but ultimately each applied her own standards.

The data in the log files was accumulated by students, resulting in one line of data for each student. The following variables were used in 1992 for each of the three applications: (a) number of times student used the application, and (b) total number of minutes application used. In 1993 there were different variables for each application. RCP - number of times RCP used, total minutes used, number of times selected a patient and ran EPR, number of times created/edited a care plan, and number of times printed a care plan. EPR - number of times visited a section of the EPR grouped into three categories: primary care, diagnostic tests, and rehabilitation. HelpBook - number of times ran HelpBook, and total number of minutes HelpBook used.

The list of various errors and user actions, which was part of the RCP log file, was tabulated into the following categories: user login error, user attempted to open second copy of RCP, user canceled login process, and user selected HelpBook (for login help).

RESULTS

Question 1 - How long did it take the students to do each assignment?

The number of sessions the students used the RCP and the amount of time they used it appear in Table 1. The students worked about 2 1/2 to 3 1/2 hours on each assignment; this includes all aspects of the assignment

Question 2 - What sections of the Electronic Patient Record (EPR) were used by the students?

The students spent an average of about two hours in the EPR book (Table 2). EPR usage data are missing for the 1993 assignment #1 and are incomplete for assignment #2. Primary care sections were chosen most often, rehabilitation sections next often, and the diagnostic sections very little (Table 3).

Question 3 - Is there a correlation between student usage (time, number of sections read, HelpBook usage) and student performance?

A summary of the students' performance is given in Table 4. All students achieved at least minimally acceptable performance of the assignment objectives. The correlations of performance with usage are given in Table 5. Performance was negatively correlated with frequency and total time in the EPR for the 1992 assignment. None of the other correlations were statistically significant. Performance for the 1993 assignment #2 was not related to use of sections of the EPR (see Table 6). EPR usage data are missing for the 1993 assignment #1 and are incomplete for assignment #2.

Question 4 - Where did students have difficulty with the applications?

Measures of the student's success at the login process during both assignments are illustrated in Figures 4 and 5. The figures show frequencies of successful logins plotted against login errors. Login errors were due to either (a) failure to select name from the listbox, or (b) enter a correct password. The data in both figures do not appear to indicate any improvement over time in the student's ability to successfully use the login process. The gross error rates (errors divided by total attempts to login) are 32% and 24% for the two assignments respectively. This indicates some improvement.

There were 23 instances during assignment #1, and 12 instances during assignment #2, when a student canceled the login process. This behavior was not counted as an error. Also, there were 52 (assignment #1) and 121 (assignment #2) attempts to open the MainBook twice. This indicates the student's were not sure if they were still in the MainBook; perhaps they lost it behind the *Program Manager* window? There were six times when a student created a care plan for the wrong patient. Five of the six corrected this error themselves, and one sought staff assistance to "locate the lost care plan." Finally, during the login process students could select a tutorial, which was a part of the HelpBook, about "how to login," and three students chose this option.

Question 5 - How did the students use the HelpBook?

Usage of the HelpBook during both assignments was less than expected. During both assignments the application was opened to the main menu page 74 times, but only seven students actually went to a content page in the HelpBook a total of eleven times. These usages were to consult the section titled *FIM Scales*. This was an appropriate usage because the OT students needed to know how to interpret the patient's scores on the *FIM Scale* to write their care plans. Further, the log file shows several instances where the same student opened the HelpBook repeatedly within a short period. We concluded students tended to exit applications rather than switch between them.

Assignment	N	Sessions		Total Time (Minutes)	
		Mean	SD	Mean	SD
1992	34	4.9	2.5	140.4	78.5
1993 #1	41	4.3	2.0	154.0	84.5
1993 #2	42	6.8	3.4	210.0	129.6

Table 1. Use of the RCP MainBook: The number of sessions and time in the application.

Assignment	N	Sessions		Total Time (Minutes)	
		Mean	SD	Mean	SD
1992	34	6.2	4.8	120.8	153.6
1993 #2	33	3.9	3.3	127.3	118.4

Table 2. Use of the EPR book: The number of sessions and total time in the application.

Section of the EPR	Mean	SD
Primary Care	17.0	17.1
Diagnostic	2.6	5.8
Rehabilitation	13.3	11.0

Table 3. *Use of the EPR book:* Frequency of choosing the primary care, diagnostic, and rehabilitation sections.

Assignment	Scoring Scale	N	Mean	SD
1992	0 - 4	34	3.3	.50
1993 #1	0- 100	41	92.8	6.07
1993 #2	0 - 100	42	89.0	6.75

Table 4. Summary of performance scores for the assignments.

Assignment	N	RCP		EPR		Help	
		Frequency	Total Time	Frequency	Total Time	Frequency	Total Time
1992	34	.076	.272	-.429*	-.514*	.301	.273
1993 #1	41	.276	.038	-.058	—	-.071	.103
1993 #2	42	.115	.215	.017	.238†	.165	-.006

* $p < .05$ † N = 33

Table 5. Pearson correlations of peformance with frequency and total time of use for the RCP, EPR, and HelpBooks.

N	Sections of the EPR		
	Primary Care	Diagnostic	Rehabilitation
33	.069	-.111	.129

Table 6. Pearson correlations of performance with usage of sections of the EPR for Assignment #2.

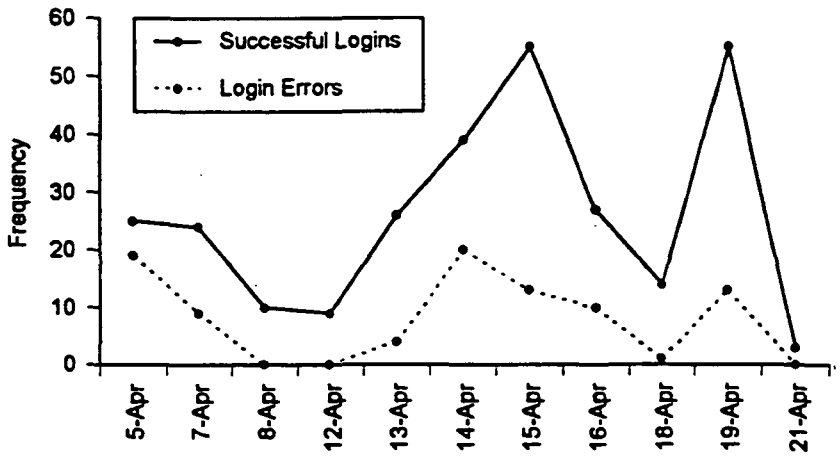


Figure 4. Successful logins vs. login errors during assignment #1.

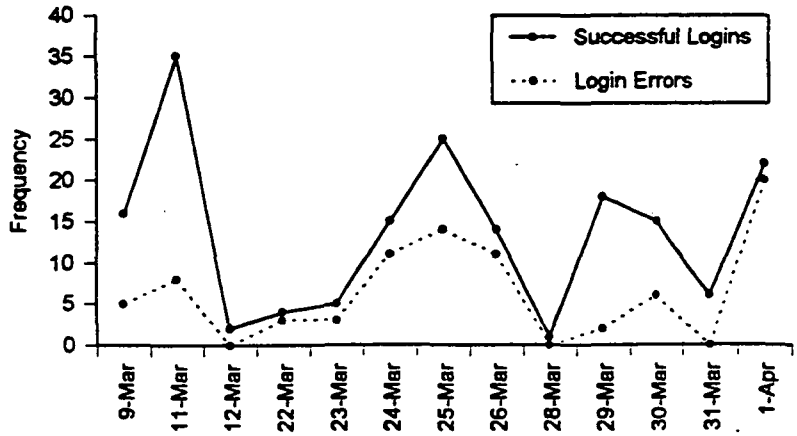


Figure 5. Successful logins vs. login errors during assignment #2.

DISCUSSION

There are two main goals of the HISS Project: (a) Students learn to use an electronic patient record and to understand some of its underlying concepts; and (b) Students learn the content in their disciplines concerning clinical problem-solving skills. The results reported here provide evidence that we are moving toward the achievement of these goals.

We uncovered several interesting findings concerning how the students used the EPR and the other applications in the Windows environment. It appears some students were confused by multiple open windows. A window may be hidden, partially or completely, by another window; some students would close the active window to reveal an underlying window. Then the student would have to rerun the just-closed application. We will make changes in the way students are introduced to Windows and the HISS applications, so that the students can become more knowledgeable and comfortable in managing multiple windows and applications. Another problem is that sometimes a student would not exit the MainBook when done, leaving the student's file open for the next user. A similar problem, leaving an unattended terminal logged in, applies to mainframe systems. However, unlike a mainframe operating system, the Windows or network software does not provide a way for the system to disconnect an idle computer. Use of the HelpBook seemed lower than expected. We suspect that the assignments to the students were designed such that the information in the HelpBook was not extensively needed, or students used other sources for reference. We plan to work with the core faculty members to redesign the assignments to make the HelpBook more integral.

On the whole, the students understood the use of the applications well enough to successfully complete their assignments. In prior years the students had to complete similar assignments using paper-based patient cases. We do not know if the students learned better or more quickly using the HISS software. Excepting some correlations in the 1992 assignment, there were no significant relationships between the students' performance and how they used the HISS applications. These results point up a difficulty in assessing the efficacy of innovative teaching technologies. It is not always realistic to expect that the teaching innovation will yield better learning (Mitchell et al., 1992). Other measures of efficacy may be used, such as how efficiently the students learn or how resources are used or that students may learn new concepts and skills (Marion et al., 1982). In the case of the HISS software, besides learning about clinical problem-solving through the assignment, the students also learned new concepts of computer use in health care and the EPR.

NOTE

A demonstration of the EPR, MainBooks, and HelpBook developed by the HISS Project will be available after November 1, 1993. It will be distributed on 3_" MS-DOS disks. For a free copy write the authors.

REFERENCES

Healthcare Informatics and Management Systems Society. (1992). Trends in healthcare computing. *Healthcare Informatics*, 9 (6), 36-39.

Marion R, Niebuhr BR, Petrusa E, & Weinholtz D. (1982). Computer-based instruction in basic medical science education. *Journal of Medical Education*, 57 (July), 521-526.

Mitchell JA, Bridges AJ, Reid JC, et al. (1992). Preliminary evaluation of learning via the AI/LEARN/ Rheumatology interactive videodisc system. In Frisse ME (Ed), *Proceedings of the 16th Annual Symposium on Computer Applications in Medical Care* 16:169-173.

Case-Based Teaching System for Radiology

Katarzyna J. Macura, Robert T. Macura, Victor E. Toro
Eugene F. Binet and Jon H. Trueblood
Department of Radiology
Medical College of Georgia
Augusta, GA 30912
kmacura@ai.uga.edu

ABSTRACT

Radiologists rely on their visual memory when analyzing new images. Image patterns that correspond to cases are developed in memory over years of experience. Our goal is to teach residents diagnosis-making skills by exposing them to cases in ways that will help them integrate those cases into visual memory. We developed the *Case-Based Consultation System* as a library of cases containing both digital images and case characteristics. The system is composed of three parts: *Case Retrieval, Teach Me,* and *Brain Atlas.*

The *Case Retrieval* module assists the user in reaching a diagnosis by providing images from cases that are relevant to the specific case being evaluated. The present library containing 128 cases of brain tumors with 648 digitized images (CT and MRI) is integrated with an indexing system that is based on case features (case history and radiological findings). In the current version of the system, the features of the new case are compared to the corresponding features of each case in the library. The *Teach Me* module is an educational program that supports visual memorization of a particular tumor pattern. The *Brain Atlas* module is designed to help the user localize a brain lesion and to offer detailed explanations of the visible anatomical structures.

INTRODUCTION

An essential part of the diagnostic process in medicine involves referring to accumulated knowledge and experience. This is especially true of radiology in which visual analysis of new images relies on a comparison to meaningful image patterns, corresponding to radiological cases, that are developed in the radiologist's memory over years of experience (Schmidt et al, 1990). In our research, we explored the way we can couple the newest multimedia technology with theories of learning and reasoning to build a computer-based instructional system that will capitalize on natural learning methods. Our goal is to teach medical students and radiology residents to make a diagnosis by showing them relevant radiological images in a way that will help them integrate those images into memory effectively. To accomplish this goal, we provide the users with a database of cases, which is structured around relevant radiological features, that contains a digital image archive with case characteristics. In our project, we expect the radiologists to use their powers of perception while analyzing radiologic images; the computers are used to store and search for information and to provide quality radiologic images relevant to a diagnostic situation.

WHY CASE-BASED TEACHING ?

Case-Based Reasoning is an artificial intelligence paradigm for reasoning from experience. It assumes a *memory model* for representing, indexing, and organizing past cases and a *process model* for retrieving and modifying old

cases and assimilating new ones (Slade, 1991). The process of case-based reasoning involves two steps: *reminding* (the reasoner retrieves potentially useful cases from memory) and *evaluating* (the reasoner draws the appropriate conclusions for the current case) (Schank and Edelson, 1990). Case-based teaching is based on the principle that students should be presented with more than just the structure of a subject in order to learn it. Instead, students should be encouraged to construct their own structure through the consideration of cases (Schank and Edelson, 1990). Conceptually, reasoning from relevant past cases corresponds to the process the experienced radiologist uses to solve new diagnostic problems. When making a diagnosis, the radiologist first extracts (from the images representing a new case) the features that appear to be significant (interpretation stage), then retrieves past cases from memory that include the same features (matching stage); the past cases contain the prior solution. The next step is to judge the similarity between the images representing a new case and the images retrieved from memory, then to make a decision or appropriate modification to account for changes in various features (adaptation and classification stage). The success of the process depends upon the ability of the reasoner to identify those attributes that make prior cases relevant, to find in the memory applicable cases, then to draw the appropriate conclusions.

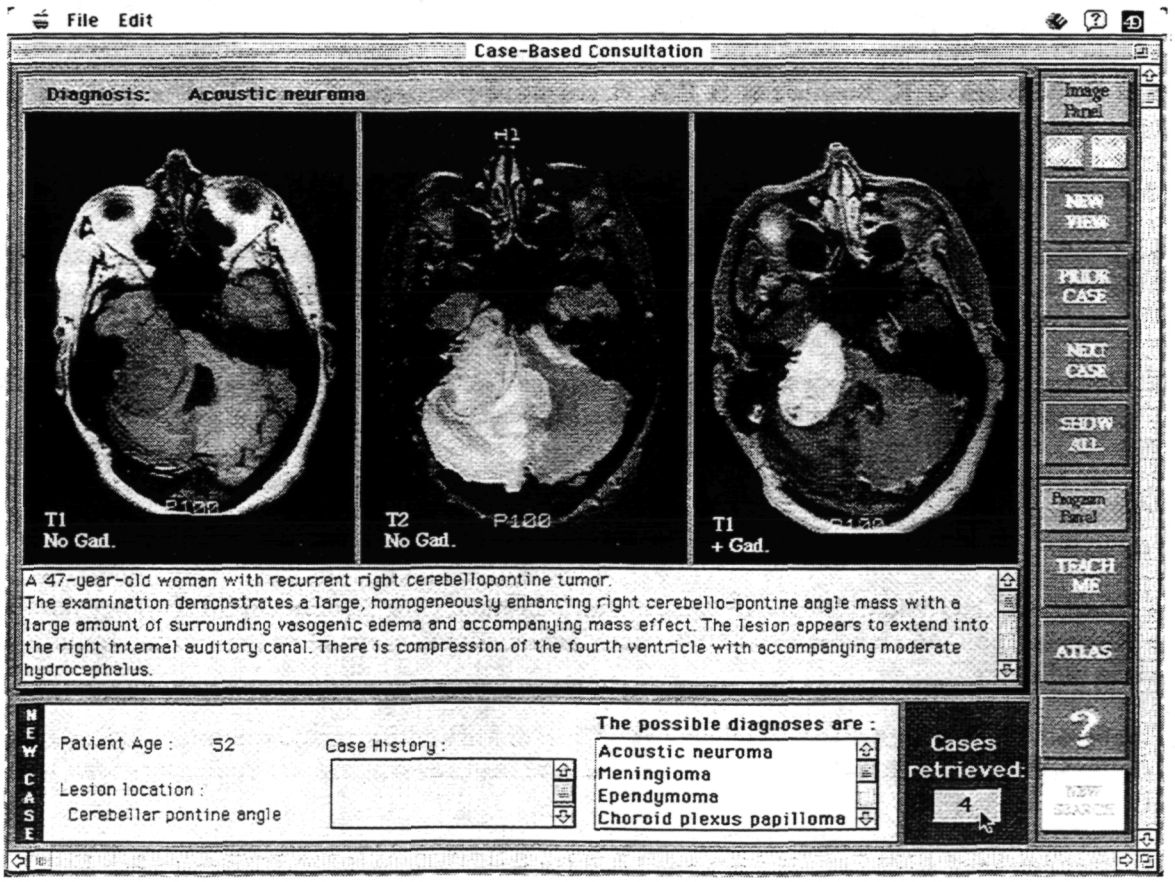
What obstacles confront the radiologist when analyzing a new case? Proper interpretation requires careful perception of abnormalities, the correct classification of those findings, and a comprehensive understanding of their significance. Errors in perception may be caused by failure to recognize a finding. Once perceived, the abnormality must be correctly classified. Once recognized and classified, the significance of the abnormality must be understood. During the decision-making process in radiology, there is a perception-problem solving relationship, named the *visual interaction process* (Rogers et al, 1991). Problem solving is affected by what is seen; what is seen and perceived is affected by the current state of the problem solving process. A radiologist extracts information from a visual display and then uses this perceptual information in a decision-making task: important descriptive features are linked to findings in the image, the findings in the image are linked to a set of candidate diagnostic hypotheses represented as case models in the memory, the hypotheses are linked to particular kinds of evidence which, if found, can guide the differentiation between diagnostic hypotheses. Radiologists in training, because of their lack of experience, have problems defining significant radiological findings, have an insufficient case memory with adequate memory models representing past cases, and frequently make classification errors. Our goal is to teach residents to make a diagnosis by providing them a case database, which is structured around relevant radiological features, that contains a digital image archive with case descriptions. By requesting that critical cues be input to the system, we want the residents to learn the key features of a particular radiological subject so they may organize their knowledge around them. We want to extend the residents' memory by providing them a large case library in which cases are structured in a hierarchy. To support the adaptation and classification phase, we offer case characteristics in addition to the retrieved cases. The system presents focused information in a way that allows the radiologist to make a more informed judgment. Case-based reasoning is a natural cognitive process used in radiology, and case-based teaching seems to be the best method of facilitating that process.

SYSTEM DESIGN

Designed in a tutorial format, the *Case-Based Consultation System* is composed of three modules: *Case Retrieval, Teach Me,* and *Brain Atlas.*

The *Case Retrieval* module helps the user reach a diagnosis by providing images from cases relevant to a specific case being evaluated. The present case library, containing 128 cases of brain tumors and 648 digitized radiological images (CT and MRI), is integrated with an indexing system that is based on case features (case history and radiological findings). To initialize the search, the user is required to input the following information: 1) the image modality (CT or MRI), 2) the age of the patient, and 3) the location of the tumor. Other input, such as the case history and radiological findings, is optional and will narrow the search space if entered. The search direction is controlled by rules incorporating critical diagnostic cues for brain lesions (age of the patient and location of the lesion) and the judgment of neuroradiology experts regarding the probability of a given diagnosis. In the current version of the system we use knowledge-based indexing in conjunction with nearest-neighbor indexing. The knowledge-based indexing approach applies existing radiologic knowledge to each case in the library to determine which features are important for retrieving each case. Presently there are equal weights on all features. The nearest-neighbor approach lets the user retrieve cases based on a sum of features in the input case that matches

cases in memory. The system prefers a case that matches on many features to a case that matches on few features. Based on the information input to the system, a list of probable diagnoses is generated and displayed. Relevant images including a textual description are produced for comparison to the case in question. The user may browse through cases, change the plane of the brain section, or view all the retrieved cases at once. After comparing the new case with those offered by the system, the user may select an image that best matches the case in question, then go directly to the case of interest. The *Teach Me* module is an educational program that supports visual memorization of a particular tumor pattern. In interactive mode, the user is presented with colored features of tumors superimposed on the pathologic images. The goal of color-coding information (radiological features) is to enhance the performance of the display. The use of color has been shown to facilitate speed and accuracy of identification and enhance short- and long-term cognitive effects (Chapman, 1993). The *Brain Atlas* module is designed to help the user localize a brain lesion. Sections of normal brain in three different planes and two modalities (CT and MRI) are presented. By clicking on the selected cut, the user may open the window to get a detailed explanation of the visible anatomical structures and the level of the section. The program was implemented in 4th Dimension™ (ACIUS Inc.) on an Apple Macintosh.



Insert Figure 1 here

Figure 1 Diagnostic Window. Based on the information input to the system, a list of probable diagnoses is displayed. Relevant images including a textual description are produced for comparison to the case in question. Using this window, the user may browse through cases, change the plane of the brain section, open the Show All Window to view all the retrieved cases, go to the *Teach Me* module, or open the *Brain Atlas* module to find an anatomic reference.

ACKNOWLEDGMENTS

The authors would like to thank the following people for their invaluable effort on developing this project: Kai Ji - programming, Kirah VanSickle - designing user interface, Claire Raeuber - anatomical references, Cliff Garzzillo - image digitization.

REFERENCES

Chapman, W. Color Coding and the Interactivity of Multimedia. Journal of Educational Multimedia and Hypermedia 1993; 2(1): 3-23.

Rogers, E., Arkin, R.C., Baron, M. Visual interaction in diagnostic radiology. In Computer-Based Medical Systems Proceedings of the 4th Annual IEEE Symposium. IEEE Computer Society Press 1991: 170-177.

Schank R C, Edelson D J. A role for AI in education: Using technology to reshape education. Journal of Artificial Intelligence in Education 1989/90; 1(2) Winter:3-22.

Schmidt H G, Norman G R, Boshuizen H P A. A cognitive perspective on medical expertise: Theory and implications. Academic Medicine 1990; 65(10):611-621.

Slade S. Case-based reasoning: A research paradigm. AI Magazine 1991; Spring:42-55.

Katarzyna J. Macura, M.D., Ph.D. is a Research Scientist in the Department of Radiology at the Medical College of Georgia. Her main research interests include applications of the methods of artificial intelligence in medicine, computer based education and cognitive models of reasoning.

Robert T. Macura, M.D., Ph.D. is an Assistant Professor in the Department of Radiology at the Medical College of Georgia. His current research area includes cognitive basis of vision and knowledge engineering.

Victor E. Toro, M.D. is an Assistant Professor in the Department of Radiology, Neuroradiology Section, at the Medical College of Georgia. His major research efforts include computer-aided instruction in radiology and brain tumors research.

Eugene F. Binet, M.D. has held the post of Professor and Chairman in the Department of Radiology at the Medical College of Georgia since 1987. His major research efforts include digital medical imaging and computer-aided instruction in radiology.

Jon H. Trueblood, Ph.D. is an Associate Professor and Chief of Medical Physics, Department of Radiology, Medical College of Georgia. His research areas include digital medical imaging and computer-aided instruction in radiology.

Intelligent Computer Aided Training Systems in the Real World: Making the Technology Accessible to the Educational Mainstream'

Madeline Kovarik
HyperTech Systems, Inc.
4497 Pinewood Road
Melbourne, FL 32934

Abstract

Intelligent computer aided training systems hold great promise for the application of this technology to mainstream education and training. Yet, this technology, which holds such a vast potential impact for the future of education and training, has had little impact beyond the enclaves of government research labs. This is largely due to the inaccessibility of the technology to those individuals in whose hands it can have the greatest impact, teachers and educators. Simply throwing technology at an educator and expecting them to use it as an effective tool is not the answer. This paper provides a background into the use of technology as a training tool. MindLink, developed by HyperTech Systems, provides trainers with a powerful rule-based tool that can be integrated directly into a Windows application. By embedding expert systems technology it becomes more accessible and easier to master.

INTRODUCTION

Education stands at a crossroads, the decision to continue on straight, maintaining the status quo, or to take an alternate route rests not only on the educational community but on businesses, legislature, research and development groups, the military, and the community. The road to take must reflect the existing and future technology and the job skills necessary for the 21st century.

Technology

From the crib to adulthood, technology is everywhere. It's our microwave, TV., VCR, and stereo system. It's reflected in our children's Nintendo games, Walkman stereos, and hand held video games but it's not in our schools. As the school bells across America ring each day, students are transported back in time to our 18th century agrarian roots [1]. Technology is left at the door (pity the child who brings a walkman to school) and the modernized version of the school primer emerges from student's desks.

School Technology

School technology has been shamefully outpaced by the outside world. Computers, older than the children who operate them, still provide the core of the technological environment. TVs and VCRs are most widely used while videodisc players, TV studios and interactive video systems are in the minority [8].

% of Schools Using These Tools	
Television Set	97.1
Videocassette recorder	95.6
Video Camera	82.9
Videodisc player	19.5
TV studio	7.9
Interactive video system	7.2

Source: Corp. for Public Broadcasting, 1991

Figure 1: Technology in the Classroom

Teachers, lacking staff development time to prepare innovative lessons and insecure about the technology, are inadequately prepared to inform our students. It wasn't until 1992 that the National Council for Accreditation of Teacher Education added Educational Technology to the national accreditation standards for teacher trainer institutions [8]. The impact of this decision will not affect education until the current Sophomores graduate and become involved in the classroom - another 3 years away.

WORKER PREPARATION

Businesses, R&D, and the military cry that workers are inadequately prepared to meet today's needs but continue to develop systems that do not reflect the current state of education. A bridge must be built to span the ever widening technology gap. Money alone will not do it. Although corporate spending on schools has been limited (only about 10% of the 2.2 billion dollars that corporations give annually goes towards K through 12 education) corporate spending for remediation of poorly skilled workers reaches over $20 billion annually. Over one-half of the Fortune 1000 companies have had to put remedial courses in place to bring workers to the level of a HS graduate.

Former President Bush's New American Schools Development Corporation (NASDC) planned to spend up to $200 million in corporate donations to develop future school models. NASDC, however, is controlled by CEOs from large corporations lacking little or no knowledge of life in the school trenches, where the war is being won or lost. Many decisions are based on what is already known and proven with little or no push taking or forward thinking. The end users, the teachers, and the children are given little thought or have little influence over when major decisions are made.

Internet, a "superhighway" for information, promoted by President Clinton, reflects such a decision. Although a valuable resource tool, it is overwhelming and fails to reflect that less than 2% of classrooms have access to a phone line or a modem [5]. Resources accessible only to research facilities and universities must not be the sole factor influencing technical decisions.

Spending

Only four countries, Finland, Switzerland, Canada, and Australia, spend more than the United States on public expenditure for education but little is allocated towards technology [3]. Many state budgets don't reflect a technological focus. In 1992 Alabama and Maryland had no specific funds generated or allocated to support educational technology.

Underscoring this problem is the fact that the movement is not towards spending more but less. In 1987, 34.2% of the State General funds and 11.5% of the Federal General funds to states were spent on elementary and secondary education. In contrast to 1991, 33.6% of state funds and 11.0% of federal funds were spent. Money alone however, is not the answer. Besides monetary gaps, a large gap exists between material development and current trends in education. Much of the computer software focuses on drill and practice or rote memorization rather than the development of critical thinking skills. Our education system, now modeled after the industrial revolution, focuses on basic skills needed for basic jobs. We must begin teaching process and the ability to teach oneself to learn [2].

WHAT'S NEEDED

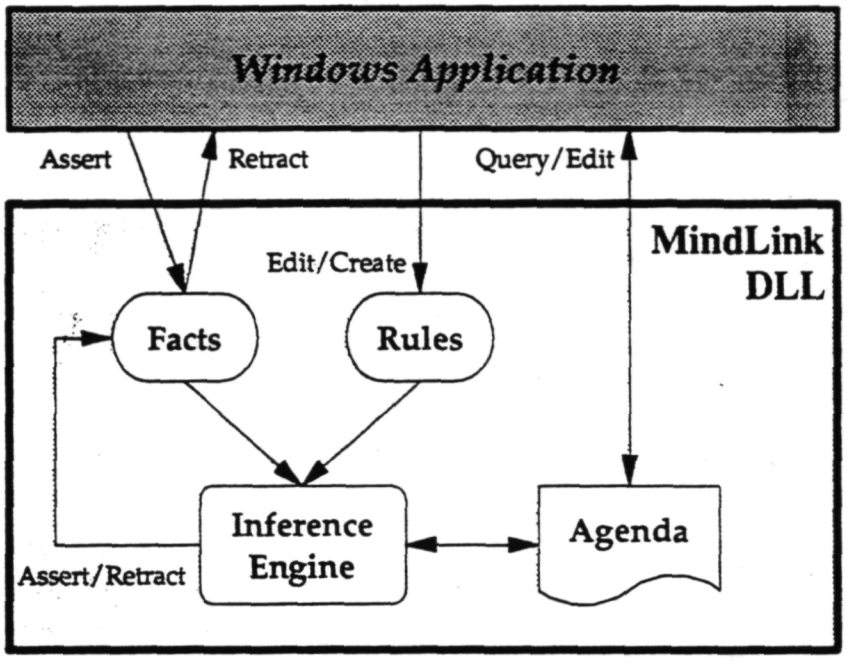
We must begin to coordinate the efforts of educators and other influences. Existing paradigms of students must be altered to begin to look at students as customers and education as the product [4]. The product must reflect the upcoming needs of the job field in the 21st century. The SCANS Report [9] attempts to look at the skills students will need to enter the workforce of the future. It focuses on competencies and skills necessary to succeed.

SCANS is composed of 5 workplace competencies and 3 foundation skills. The workplace competencies focus on the technology, systems, information, interpersonal skills, and resources that effective workers of the 21st century need to use productively. Foundation skills are the basic skills, personal qualities, and thinking skills necessary to succeed in the future.

Technology and training are keys which will drive the curriculum beyond basic skills and enter learning as a process and search for information.

Availability of Technology

How then can we begin to make a difference? We can neither assume that a national initiative will develop to take care of the problem, nor can we do nothing, hiding behind the excuse that our local problems are such a small part of the whole that changes would have no effect.



For the mainstream teacher there is little accessibility to cutting edge research. Simply throwing technological tools at the educational developer has little or no meaning. Authoring systems for computer based training and development such as Toolbook, Owl and Authorware are powerful but require a tremendous effort on the instructional plan designer to be used effectively. The time to develop mastery and expertise of such systems is beyond the scope of the mainstream teacher. What is needed is a powerful tool, which is simple to use, doesn't take a significant amount of time to master, and allows educators and developers of training systems to insert the technology directly into the ICAT application with minimal effort. MindLink, developed by HyperTech Systems, is the first step towards this goal. It takes the full range of functionality of an expert system but contains it in a well-defined shell. MindLink is implemented as a Windows Dynamic Link Library (DLL). This provides compatibility and accessibility to a wide range of Windows applications. A rule-based inference engine, MindLink allows the user to specify rules which describe what can be inferred or should be performed based on the known facts. as a DLL, integration with applications is simply a matter of linking the DLL at run or compile time. this allows maximum flexibility of student interaction while still working within the parameters, or rules, specified by the educator. MindLink provides an extensible environment for developing and delivering intelligent applications in ToolBook, ObjectVision, Smalltalk/V, C, C++ and other development tools and languages which support linking DLLs.

THE FUTURE AND CONCLUSIONS

Building on the power of MindLink, HyperTech is moving towards the development of an intelligent lesson plan assistant. This tool will provide the core set of capabilities including student tracking, performance, evaluation and remediation recommendations for any training system.

Education is at a crossroads. Technology can be the catalyst and facilitator of change that allows educators to take an alternate route towards a successful future that meets the needs of businesses, military and the community.

REFERENCES

[1] Bruder, I., Buchsbaum, H., Hill, Maggie, Orlando, Louise C. *School Reform: Why you need Technology to Get There*, Electronic Learning, May/June 1992, Vol 11, No 8.

[2] *Workplace Basics: The Skills Employers Want*, American Society for Training and Development, 1630 Duke St., Alexandria, VA, 22313.

[3] Saks, Judith Brody, *Education Vital Signs*, The American School Board, Dec 1992, Vol. 179, No. 12, p32-45.

[4] Blokker, B., *Educational Leadership for the 21st Century*, Phase II, 1992-1993

[5] La Quey and Stout, Connie, *Technology: Hastening the Future*, America's Agenda: Schools for the 21st Century, Spring 1993, Vol. 3, No. 1, p25-33.

[6] Dyrli, Odvard Egil, *Beyond Drill and Practice*, Instructor and Teacher, Oct. 1989, Vol. XCIX, Issue 3, p36-37.

[7] Toch, Thomas, *Tomorrow's Bottom Line*, America's Agenda: Schools for the 21st Century, Vol. 2, No. 2, Spring 1992, p21-31.

[8] *Industry News*, Electronic Learning, Vol. 11, No. 8, May/June 1992, p6-7.

[9] *What Work Requires of Schools: A SCANS Report for America 2000*, U.S. Department of Labor, June, 1991, p xvii - xviii.

ICAT and the NASA Technology Transfer Process

Noah Rifkin, Hans ten Cate, Alison Watkins
The Egan Group
1701 K Street, NW
Washington DC 20006
202-775-0720 202-293-1408

This paper will address issues related to NASA's technology transfer process and will cite the example of using ICAT technologies in educational tools. The obstacles to effective technology transfer will be highlighted, viewing the difficulties in achieving successful transfers of ICAT technologies.

Dr. Bowen Loftin, an ICAT developer from the University of Houston, used the technology to create the Intelligent Physics Tutor, an interactive training program for high school physics students. Several companies have expressed interest in marketing the Physics Tutor and in developing comparable tutors for other subjects. Despite the high level of commercial interest, the transfer remains incomplete.

The problems encountered in transferring the ICAT Physics Tutor illustrate commonly cited barriers to transfer and commercialization of NASA technologies. Those barriers will be discussed in more detail. We will present the attributes of the ICAT Physics Tutor that make it commercially desirable and will focus on technology transfer obstacles which to date have prevented agreements with interested companies. We will also present this in the larger context of NASA technology transfer, discussing the many barriers facing transfer of NASA's commercially applicable technologies. Finally, we would like to present recommendations for easing the technology transfer process for ICAT and NASA technology at large with examples of successful ICAT transfers for educational purposes.

Analytic Tools For Designing Virtual Environment Training Systems: Nuclear Power Plant Maintenance Applications'

Mark S. Schlager
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
(415) 859-2881

Randall J. Mumaw & Douglas G. Hoecker
Westinghouse Science and Technology Center
1310 Beulah Road Pittsburgh, PA 15235 (412) 256-2683

1 INTRODUCTION

Laboratories in DoD, NASA, and private industry are actively investigating the potential advantages of using virtual environment (VE) technology for training in several application areas. A common approach to the development of VE training systems has been to identify a "high-profile" application--one for which conventional simulation is not feasible or prohibitively expensive--and build a prototype. While much can be learned about how to build a VE simulation using this approach, more general questions concerning the utility of VE as part of a comprehensive training delivery system remain unaddressed. For example, can more mundane applications be identified for which VE could provide effective training? How can VE capability requirements be determined for a given training application? How can instructional developers determine which part(s) of training (e.g., familiarization, knowledge acquisition, basic and advanced skill development) should employ VE (as opposed to other delivery modes)? We believe that questions such as these must be addressed if VE is to grow into a widely applicable training tool.

As part of a study of the potential applications of VE simulation to aircraft and space maintenance training for the Air Force and NASA, Schlager and his colleagues (Boman, Schlager, Gille, & Piantanida, 1992; Schlager, Boman, & Piantanida, 1993; Schlager, Boman, Piantanida, & Stephenson, 1992) developed a set of analytic tools to look more closely at factors that determine (1) the suitability of VE simulation for training a given task, (2) the appropriate combination of VE component technologies required to deliver the desired training, and (3) the relative cost- and instructional-effectiveness of VE and other modes of training. That study concluded that the appropriate configuration of VE simulation technologies could, in conjunction with other delivery platforms, provide instructional and cost benefits for certain categories of maintenance tasks (Schlager, Boman, & Piantanida, 1993).

In this paper, we describe work in progress to extend our analytic approach to another domain-- nuclear power plant maintenance. Nuclear power plant maintenance shares several features with aircraft and space, including very large, complex objects, confined workspaces, subtle (and invisible) dangers, and unpredictable conditions. Current training is also similar, primarily relying on large-scale physical mockups. Consequently we expect to find the approach highly applicable. First, we briefly describe three sets of analytic tools developed in the prior study: task selection criteria, VE requirements matrices, and cost-effectiveness factors. We then describe nuclear power plant maintenance task categories identified as potential candidates for VE training and some of the cost effectiveness issues we have identified thus far.

2 ANALYTIC APPROACH

In this section, we describe some of the methods and instruments developed in the Air Force/NASA study to analyze VE suitability for aircraft and space maintenance training. Task selection criteria help identify tasks that might benefit from VE training; VE requirements matrices are used to derive VE training system requirements; and cost-effectiveness factors form the basis of analyses aimed at judging the relative merits of VE and existing training delivery technologies.

2.1 Task Selection Criteria

The initial determination of whether a task category is a suitable candidate for VE training is based on three sets of criteria: task constraints, training impact, and learning outcomes.[1] The criteria were derived empirically. They are not meant to be exhaustive; we expect the list to grow as more experience is gained in VE training applications. (For a complete description of how these criteria are employed in the task selection process, see Schlager, Bon-an, and Piantanida, 1993.)

Task constraints are attributes of the task environment that impose obstacles or complexities on training that can be overcome by simulating the environment with VE technology. They include:

- Size of the environment (very large or very small)
- Limited physical access
- Unusual physics
- Teaming or coordination
- Hazards, risks

Training impact criteria are practical, cost/benefit concerns. To be considered, VE technology must fill a perceived gap in training and have a large enough impact on existing training to justify its use. In the aircraft maintenance study, several tasks that were otherwise deemed appropriate for VE training (e.g., ordnance disposal) were dropped from consideration due to these criteria. They are:

- Perceived need for additional/better training
- Task frequency
- Number of personnel who perform the task
- Availability, cost of alternative technologies

Learning outcomes, the third set of criteria, were the most difficult to identify because we could not point to any directly-related VE training applications. Based on an analysis of other successful VE applications, we identified job skills that appear to be supported or enhanced by VE simulation and reasoned that those skills could also be developed through the kind of experience offered by VE technology. We arrived at the following list:

- 3-D visualization of an inaccessible object or environment
- Psychomotor control, accuracy
- Planning and coordination of activities
- Performance of complex or non-routine procedures

The criteria represent several categories of cognitive, perceptual, and motor abilities that have been shown to benefit from practice in a simulated training environment. Little empirical data exist to suggest whether or when VE simulation conveys advantages over other forms of simulation in developing these abilities. Our assumptions are based on rational analysis of major VE capabilities (e.g., scalable, 3D representation; 360 degree field of regard; position tracking; risk-free interactively; functional models) not available in other forms of simulation.

2.2 VE Requirements Matrices

The capabilities displayed by VE simulation systems are achieved by combining several component technologies on a single platform. Boman and Piantanida (1993) identified nine categories of VE component technologies. Each category contains several options (e.g., dual CRT display; dual LCD display) and sophistication levels (e.g., low, medium, or high resolution; tracking range and sensitivity). The system developer's responsibility is to select among the options to achieve the most appropriate (and cost-effective) configuration for the given application (every category need not be represented to achieve the goals of the application). To assist the developer (or cost-effectiveness analyst) in selecting the appropriate components, we developed a set of three matrices. The matrices parallel the three sets of task selection criteria, matching VE component technologies to the training application's task attributes, instructional user interface requirements, and desired learning outcomes. Figures 1, 2, and 3 show abbreviated versions of the matrices. Across the top of each matrix are the nine categories of component technologies. In the actual matrices, each category is further divided into subcategories representing subcomponents (e.g., simulation, instructional, and control software) and sophistication levels.

The Task Requirements Matrix (Figure 1) matches the technologies against task attributes, which are divided into four categories: environment participants, activities, and knowledge/skills. Again, in the actual matrix, specific task attributes, derived from a task analysis, would be listed under the appropriate category. Each attribute would indicate one or more component technologies. For example, if the task environment is large, say a C-17 cargo aircraft jet engine, students might need a way to move around it in the virtual hangar. This might require a high-range tracker. If the activities they perform require movement in three dimensions, a hand-held 3D/6D input device might be called for. If, however, manual skills are used in the task (working with hand tools), then navigation via hands-free input (e.g., spoken commands) may be required.

Task Attributes	DISPLAY	TRACKER	AUDIO	3D/6D INPUT	GESTURE	HAPTIC	SPEECH	COMPUTER	SOFTWARE
ENVIRONMENT									
PARTICIPANTS									
ACTIVITIES									
KNOWLEDGE. SKILLS									

Figure 1. Task Requirements Matrix.

The Instructional Requirements Matrix (Figure 2) matches the technologies against requirements for the student and instructor user interface. For example, if the student is expected to be able to read instructions presented within the simulation, high-resolution display technology is required. (Note that this may counter-indicate a requirement identified in the Task Requirements Matrix.) Instructional interface requirements, such as selectable menus, labels, session control, and instructional feedback are especially valuable for self-pacing. However, not all applications require these features. In networked battlefield simulations, for example, they may not be desirable.

Instructional Requirements	DISPLAY	TRACKER	AUDIO	3D/6D INPUT	GESTURE	HAPTIC	SPEECH	COMPUTER	SOFTWARE
STUDENT INTERFACE:									
INSTRUCTOR INTERFACE:									

Figure 2. Instructional Requirements Matrix.

The Outcome Requirements Matrix (Figure 3) matches the technologies to classes of learning objectives that are traditionally considered in selecting a delivery platform. This matrix is the final arbiter of whether a particular technology should be incorporated into the system. For example, a feature such as haptic feedback may have been called out as a task-related requirement, but the skill requiring haptic feedback (e.g., judging the resistance in a torque wrench) may not be an outcome of interest for VE training (torque wrench training is better delivered using an actual torque wrench). Consequently, haptic feedback would not be required.

Training Outcomes	DISPLAY	TRACKER	AUDIO	3D/6D INPUT	GESTURE	HAPTIC	SPEECH	COMPUTER	SOFTWARE
Knowledge of facts									
Knowledge of procedures									
Part-task performance									
Full-task performance									
Task mastery									

Figure 3. Outcome Requirements Matrix.

TASK ATTRIBUTES	VE REQUIREMENTS
Environment:	
Large aircraft	Display:
Sound of engines	Low resolution
Taxiways, hangars	3-D visual
Participants:	360 degree field of regard
Marshal, pilot	3-D audio
Other aircraft, technicians	
Activities:	
Hand-signal communication	Tracker:
Knowledge, skills:	Medium resolution
Real-time spatial reasoning	Variable range
Signal codes	Batons, arm
INSTRUCTIONAL REQMTS.	Gesture Recognition:
Instructional. feedback	Arm/hand position or none
Authoring	
Lesson modification	SoftwarE:
Session control	Simulation modeling
Data management	Manual or auto simulation control
	Instructional feedback
TRAINING OUTCOME	
Part-task performance	

Figure 4. Partial VE system requirements analysis output.

Figure 4 illustrates the partial output of a system requirements analysis for a task called marshaling, in which a technician directs the movement of aircraft on the tarmac by standing in front of the aircraft and using batons and hand signals to communicate instructions to the pilot. Training would involve directing simulated aircraft into position without damaging equipment or injuring personnel. The aircraft could be controlled either by a confederate (the instructor; no gesture recognition) or by the system.

2.3 Cost-Effectiveness Factors

Our objective in conducting the cost-effectiveness analysis is to help determine the conditions under which VE simulation could offer a reasonable alternative to competing training technologies for a single task category (recall that the need for enhanced training has already been established in the task selection phase). A complete description of our cost-effectiveness analysis approach is presented in Schlager, Boman and Piantanida (1993). Here, we briefly describe the general design and the cost and benefit factors employed. The analysis is structured around three training scenarios, current training, which serves as a baseline, and two future scenarios. The baseline scenario documents current training practices, deficiencies, technologies, and costs. One future scenario employs existing technologies to alleviate current training deficiencies. For example, a new suite of hardware trainers and/or a new library of CBT/IVD lessons might be specified. The second future scenario employs VE technology either in addition to, or in place of, existing technology for a specified part of training. For example a VE simulation walk-through of a containment area might replace or supplement an IVD-based lesson.

In conducting the analysis, we look at several categories of costs and benefits. The following categories of training system life-cycle costs are considered:

- Development costs
  - Full-scale development
  - Hardware
  - Software
  - Simulation
  - Courseware
- Maintenance and housing costs
- Upgrading costs (system and courseware)
- Cost avoidance factors
- VE R&D costs[2]

Full-scale development cost is traditionally used for the first implementation of a hardware trainer; the remaining development cost categories are used for computer-based technologies (including VE). Cost avoidance factors account for reductions in risk of injury or damage to equipment, down-time, training time, and development costs associated with the training technology.

Non-monetary benefit factors are divided into the following categories:

<u>Instructional Support Capabilities</u>
- Testing and data management
- Authoring
- Instructional feedback
- Session control

<u>Generic Attributes</u>
- Transportability
- Availability
- Reconfigurability
- Fidelity

Instructional Utility
- Training Methods
- Training Outcomes
- Setting

Instructional Support Capabilities are features found in many computer-based training systems that enhance (1) the instructor's ability to control instruction and (2) the student's learning and practice opportunities. *Generic Attributes* are inherent attribute of a technology (not built into a training technology deliberately), which are commonly cited as factors in the superiority of one technology over another. For example, the *availability* of IVD lessons is claimed to contribute to reductions in training time (Baughman, Hudspeth, Kenrick, & Thore, 1991). *Instructional Utility* reflects where, and how well, the training technology being considered fits within the curriculum. For Example, common maintenance training methods for large mechanical systems include tutorial, walk-through, teaming, and freeplay (interestingly, advanced troubleshooting skills are learned primarily on the job). IVD is most appropriate for tutorial lessons; it is less effective for skill development. In contrast, VE could support walk-throughs, teaming, and possibly freeplay. Finally, we consider the setting(s) in which the technology can be used. Traditionally, maintenance training progresses through three stages: classroom instruction, hands-on lab, and on-the-job-training (OJT). Some training delivery systems are best suited for the classroom setting, while others are useful in the hands-on stage of training. Some systems are available to students in OJT, while others are not. Systems that offer the flexibility of operating in more than one setting could confer benefits over single-settings systems.

3 VE APPLICATIONS IN NUCLEAR POWER PLANT MAINTENANCE

Nuclear power plant maintenance involves thousands of tasks ranging from routine valve maintenance to refueling. We have focused initially on maintenance tasks performed in containment. The term "containment" refers to the fact that, in a pressurized water reactor, all radioactive systems are housed inside a negatively pressurized building that prevents the escape of radiation. Maintenance crews must pass through an interlock to work in the containment part of the plant. This occurs primarily at 18-month intervals, as part of refueling outages. In the following sections, we describe containment maintenance task categories that fit our task selection criteria and some of the cost factors that should be considered in the decision to develop a VE training system for containment maintenance tasks.

3.1 Task Constraints and Learning Outcomes

Containment maintenance reflects four of the five task constraint criteria and all four learning outcome criteria. Maintenance activities are high risk endeavors. Many tasks are unique to the situation, practice is infrequent, teamwork is critical, and time pressure is a factor. Safety and equipment damage are major concerns in both in performing actual maintenance procedures and in the logistics of maintenance. The containment environment can best be described as a three-dimensional maze of functional devices and support structures tens of stories high, having both visible and invisible hazards. Devices within the containment area, such as the reactor coolant pump shown in Figure 5, can be several stories in length, contain many internal subsystems, and may be difficult to access. As was the case in the Air Force/NASA study (Schlager, Boman, Piantanida, & Stephenson, 1992), we are finding that several categories of preventative and restorative maintenance tasks in such devices could be suitable candidates for VE training. However, in this discussion we have chosen to focus on a class of tasks that was not identified in the prior study--maintenance logistics.

Planning and crew coordination, spatial reasoning in three dimensions, and psychomotor control and accuracy are all skills required to successfully implement many containment maintenance activities. During refueling outages, multiple teams may be in close proximity; each team needs to avoid being a hindrance to other teams., In some situations, scaffolding and protective barriers must be erected before actual maintenance work can begin. Large tools or replacement components, which are too large or heavy to carry in, must be brought in with mechanical devices (e.g., winch and pulley) and lowered into the work area through several stories of interconnected components and support structures. Radiation is another critical factor in determining how maintenance technicians set up and perform tasks. Radiation is not uniform throughout containment. It is the technician's responsibility to know where the high radiation areas are and avoid them, if possible. Gaining access to the work

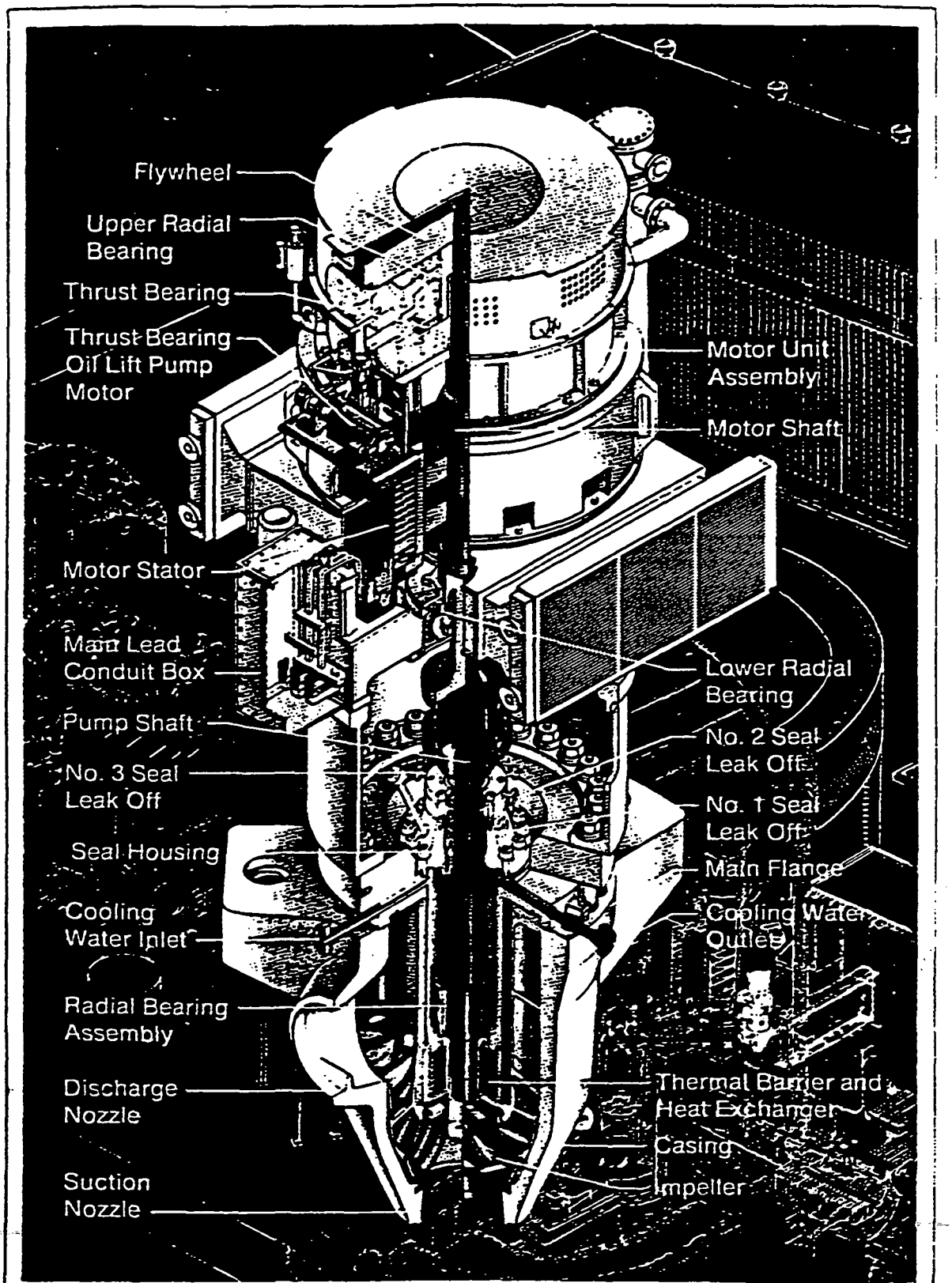


Figure 5. Cutaway of reactor coolant pump.

area is also a concern. Large components that are in the way of the crew must be removed temporarily and set aside somewhere. Some equipment components are both hard to handle and subject to damage by incidental contact. For example, a cover can be damaged by contact with tools or other equipment so that it no longer fits adequately. These are very expensive and difficult to replace. Finally, technicians must perform under adverse physical conditions. Technicians often must work in a constrained space and wear protective clothing in containment (gloves, full body suit, hood and visor), which impairs vision and manual dexterity. Heat build-up in the suit can cause discomfort and further impair the technician's ability to perform tasks.

3.2 Training Impact

In this section, we consider the need for additional training and the potential impact of VE simulation. The scenario above describes tasks that need to be performed in relatively novel ways each time. It suggests a need to understand the "big picture" of containment to plan and rehearse maintenance tasks. Clearly, trial-and-error is to be avoided in the actual environment. Realistic practice and rehearsal under a large number of conditions could have great benefit for the development of strategies for planning and performing logistical procedures. Such practice would require a large-scale simulation. Currently, maintenance training employs very expensive full-scale, physical mock-ups of major devices3 but no simulation of the environment in which these devices reside. VE could simulate the entire containment space and aid technicians in exploring the most efficient means for erecting scaffolding, identifying potential conflicts between teams and completing the task. VE could also simulate the difficulty of seeing, hearing, and moving in containment encumbered by protective clothing and shields.

VE simulation and display software could also help teach technicians to recognize and avoid areas of high radiation.4 Current methods of training technicians to avoid radiation are primitive. Information on intensity values is presented on a sheet of paper that shows values at multiple specific spots and only in two dimensions. How the intensity fields vary in three dimensions is not communicated. There are also a handful of heuristics that can be used to avoid major single radiation sources but those cannot be used effectively to find the areas of lowest overall exposure.

3.3 Cost Factors

In nuclear power plants, the cost justification may have to come from multiple purposes of the technology the more applications, the lower the cost per application. In this section, we look at some of the potential cost avoidance factors associated with the introduction of VE training simulation. We recognize that some operational tasks could also benefit from VE simulation. These include the planning and rehearsal of specific maintenance tasks and visualization of radiation fields while in containment. In nuclear power plants costs are directly related to the operational state of the plant. The plant requires that a certain set of equipment is available for operation. Operators must shut down the plant if that equipment is not available. If performed incorrectly, maintenance tasks can cause the plant to shut down automatically. This is called a *plant trip*. For example, a plant trip might occur if the incorrect piece of equipment is taken off-line. A trip may also be caused by a technician who creates a malfunction somewhere. A plant trip is extremely undesirable because of the enormous costs involved. When the plant goes off-line, the utility must buy replacement power from another source, costing the plant approximately $1 million a day. Consequently, increases in maintenance speed and accuracy and reductions in errors due to more effective training is likely to result in fewer or shorter shutdowns and a corresponding cost savings.

Technician expertise is also an important commodity. Each technician is permitted only so much exposure over a period of time. If a skilled technician receives his entire dose while completing just a few tasks, another person must be found to complete the full slate of maintenance tasks. For this reason, low-skill, high-exposure tasks are given to technicians called "jumpers" who receive minimal training (but a large hourly wage). Estimates for the cost of 1 rem exposure to one man (a man-rem) are on the order of 10-40,000 dollars. Consequently, reductions in exposure can quickly lead to cost savings.

4 CONCLUSIONS AND NEXT STEPS

We are in the process of applying analytic tools for the design of VE training systems to our third domain, nuclear power plant maintenance. Based on our analyses to date, we are confident that VE simulation could play an

important role in containment maintenance training. The tools have helped us analyze VE training applications similar to those found in other maintenance domains. More importantly, they have helped us uncover potential applications in a new class of tasks—maintenance logistics. The work reported in this paper represents the initial steps in the development of a long-range strategy for incorporating VE simulation into nuclear power plant maintenance training. We plan to continue this analysis by specifying VE system requirements for the logistics planning and radiation visualization tasks.

NOTES

1 While our approach of selecting tasks to fit a training technology may appear to contradict Instructional Systems Development (ISD) methodology, the reader should not assume that we advocate a reversal of ISD methodology. In an actual ISD process, our criteria would be used to determine the suitability of VE for a given task and training objective.

2 VE research and development costs were considered in the Air Force/NASA project because many VE technologies have not yet matured and the sponsors were interested in understanding the investment required to bring those technologies to maturity.

3 We address the relative advantages and disadvantages of VE and hardware trainers in Schlager, Boman, and Piantanida (1992).

4 Westinghouse currently has a program to build a "window on the world" VE tool to model radiation intensity in three dimensional spaces.

ACKNOWLEDGMENTS

The development of the analytic tools was supported by the U.S. Air Force and NASA under NASA Cooperative Agreement NCC2-643 to SRI international. The authors would like to thank Dr. Robert Stephenson (ret.), Special Projects Office, AFHRL, for his guidance and support in the development of the methods and instruments described in this paper.

REFERENCES

Baughman, M.L., Hudspeth, D., Kenrick, D., & Thore, S. (1991). *Workstations in education and training.* DARPA Technical Report, Order No. 6675/4, Washington, D.C.: Defense Advanced Research Projects Agency.

Boman, D. & Piantanida, T. (1993). *Virtual environment systems for maintenance training (Volume Z: Survey of the technology).* SRI Technical Report No. 8216, Menlo Park, CA: SRI International.

Boman, D., Schlager, M.S., Gille, J., & Piantanida, T. (1992). The readiness of virtual environment technology for use in maintenance training. In *Proceedings of the Interservice/Industry Training Systems Conference,,* San Antonio, TX, November 4-6.

Schlager, M.S., Boman, D., & Piantanida, T. (1993). *Virtual environment systems for maintenance training (Volume 3: Cost-effectiveness analysis).* SRI Technical Report No. 8216, Menlo Park, CA: SRI International.

Schlager, M.S., Boman, D., Piantanida, T., & Stephenson, R. (1992). *Forecasting the impact of virtual environment technology on maintenance training.* Paper presented at the Space Operations, Applications, and Research Symposium, NASA JSC, Houston, TX, August 4-6.

Assessing Human Performance in Virtual Environments

Donald R. Lampton James P. Bliss, and Bruce W. Knerr
U.S. Army Research Institute
STRICOM, Orlando Field Unit
12350 Research Parkway
Orlando, FL 32826-3276

The Army has made a substantial commitment to the use of a simulated, electronic battlefield for combat training. Current and next-generation training systems provide effective training for soldiers fighting from vehicles, but not for individual dismounted soldiers. Virtual environment technology has the potential to provide that capability.

As the first step in a research program to investigate the use of virtual environments for training dismounted soldiers, we developed a set of tasks and performance measures to assess human performance and the effects of immersion as a function of training and system characteristics. The tasks were developed for presentation using two 486 PCs with Intel DVI display boards, Sense8 WorldToolKit software, and a Virtual Research Flight Helmet. They include vision (acuity, color, distance estimation, and search), simple and complex reaction time, locomotion through various simulated structures, and several types of object manipulation.

The results of the research and future plans will be presented.

Marshall Space Flight Center's Virtual Reality Applications Program 1993

Joseph P. Hale, II
Crew Systems Engineering Branch/EO23
Marshall Space Flight Center
MSFC, AL 35812
(205) 544-2193

INTRODUCTION

A Virtual Reality (VR) applications program has been under development at the Marshall Space Flight Center (MSFC) since 1989. Other NASA Centers, most notably Ames Research Center (ARC), have contributed to the development of the VR enabling technologies and VR systems. This VR technology development has now reached a level of maturity where specific applications of VR as a tool can be considered.

The objectives of the MSFC VR Applications Program are to develop, validate, and utilize VR as a Human Factors design and operations analysis tool and to assess and evaluate VR as a tool in other applications (e.g., training, operations development, mission support, teleoperations planning, etc.). The long-term goals of this technology program is to enable specialized Human Factors analyses earlier in the hardware and operations development process and develop more effective training and mission support systems.

The capability to perform specialized Human Factors analyses earlier in the hardware and operations development process is required to better refine and validate requirements during the requirements definition phase. This leads to a more efficient design process where perturbations caused by late-occurring requirements changes are minimized. A validated set of VR analytical tools must be developed to enable a more efficient process for the design and development of space systems and operations. Similarly, training and mission support systems must exploit state-of-the-art computer-based technologies to maximize training effectiveness and enhance mission support.

The approach of the VR Applications Program is to develop and validate appropriate virtual environments and associated object kinematic and behavior attributes for specific classes of applications. These application-specific environments and associated simulations will be validated, where possible, through empirical comparisons with existing, accepted tools and methodologies. These validated VR analytical tools will then be available for use in the design and development of space systems and operations and in training and mission support systems.

MSFC VR SYSTEM CONFIGURATION AND DEVELOPMENT

The MSFC VR systems reside in the Computer Applications and Virtual Environments (CAVE) Lab in Building 4610. System components consist of VPL Research, Inc. Eyephones (Model 1 and LX), DataGloves, and software (Swivel 3D, Body Electric, and ISAAC), Polhemus Isotrak and Fastrak spatial tracking systems, two Macintosh IIfx computers and two Silicon Graphics Inc. graphics computers (4D/310VGX and 4D/320VGXB). Two single-person configurations are possible. These are the Eyephone Model 1 (stereo views) with one DataGlove and the Eyephone LX (stereo views) with one or two DataGloves (i.e., both right and left hands, simultaneously). A two-person configuration is possible and consists of both Eyephones (monocular views) with one DataGlove associated with the Eyephone Model 1, and up to two DataGloves associated with the Eyephone LX. In the two-person configuration, both people are in the same Virtual World (VW) simultaneously and each is able to see and interact with the computer-generated image of the other.

Several VR development activities are underway at MSFC and with the Johnson Space Center (JSC). EXOS, Inc. is under contract to develop a Sensing And Force-Reflecting Exoskeleton (SAFiRE) for the hand. This device will provide force-reflecting feedback to the fingers and hand as the user touches and grabs virtual objects.

Tomorrowtools is under contract to develop a 30 sensor spatial tracking system. Using ultra-sonics to determine the location of each of the sensors and infrared to transmit data to the base units, this system provides an untethered method to track up to 30 body points and/or other objects simultaneously. This will be particularly useful in dynamic work envelope analyses. In-house, work is underway to broadcast the video signals to the Eyephones, removing the video cable to the user. Integration of the two latter activities' products will free the user from existing I/O cables.

MSFC is cooperating with JSC in their efforts to develop a "long distance" two-person capability using their in-house-developed VR system. This means allowing two people to see and interact with the computer-generated image of the other in the same Virtual World, simultaneously, with one person at JSC and the other at MSFC! Basic capabilities have already been demonstrated. As VR technology evolves, this virtual link between, and eventually among, the NASA Centers will provide major foreseeable and unanticipated benefits.

COMPUTATIONAL HUMAN FACTORS

Human Factors issues and considerations in hardware and operations development present a large class of potential VR applications. VR technologies and techniques currently provide some limited macro- and micro-ergonomic analytical tools for consideration of operational, viewing and reach envelope requirements, in both one-gravity and micro-gravity environments.

Macro-ergonomics analyses for the topological design of work areas can consider what one is able to see from a variety of eye reference points. These analyses can also include operationally-driven components such as translation paths among the various worksites.

Combined with scaleable user anthropometry attributes, micro-ergonomics analyses for the spatial layout of workstations can consider what one is able to see from a variety of eye reference points and what one is able to touch from a variety of shoulder and seat reference points and/or foot restraint locations, using a range of virtual anthropometric sizes.

Many analyses that use "Fomecor" mockups, the KC-135 (provides approximately 30 seconds of weightlessness during each cycle of parabolic flight), or the Neutral Buoyancy Simulator (underwater facility for simulating weightlessness) are candidates for VR. It is not that VR would completely replace these other technologies and techniques, but it adds another tool to the analytical toolkit.

In some instances, VR might be considered for use in an analysis that would have otherwise not be undertaken. Resources (time, people, materials, etc.) required for a "standard" simulation or mock-up analysis may be greater than the expected return. In this case VR, due to its relatively low utilization costs, would surpass the cost/benefit ratio threshold and enable an analysis that would have otherwise been forgone.

Similarly, VR can enhance and enable more effective utilization of standard simulations and mock-up analyses. By preceding these analyses with preliminary VR analyses, both the hardware and operations can be refined so that the return from the standard analyses is increased. This is accomplished by either reducing the magnitude or number of standard analyses and/or improving the fidelity of those analyses with a more mature design.

Because the VWs are nothing more than computer files, design changes can be done more quickly and more candidate configurations can be subsequently analyzed than is currently possible with existing, "standard" Human Factor tools (e.g., Fomecor mockups).

VALIDATION STUDIES

Several VR applications validation studies are underway at MSFC. The selected applications are those that can utilize existing VR technology and capabilities. Before VR can be used with confidence in a particular application, VR must be validated for that class of applications. For that reason, a specific validation study has been proposed for classes of applications. The validation approach is to compare the results from the VR analyses or application with a specific past or present analysis, design, and/or actual flight experience.

Early applications utilizing the VR system currently in place at MSFC fall primarily into viewing analyses (including visualization of spatial layouts) and reach envelope analyses. Three studies are underway to begin validating VR for these classes of applications.

The objectives of the first study are to develop, assess, and validate VR as a macro-ergonomics analysis tool for the topological design of work areas. Two existing control rooms and their corresponding virtual counterparts will be used to collect subjects' qualitative and quantitative judgments on a variety of measures. The Spacelab Payload Operations Control Center (POCC) and the Data Control Room (DCR) have been selected, based on their apparent separation on a variety of continua (e.g., large/small, spacious/cramped, aesthetically well/poorly designed, etc.). A corresponding Virtual POCC (VPOCC) and Virtual DCR (VDCR) have been developed that contain the basic elements (e.g., tables, monitors, printers, communication panels, etc.) and spatial layout of their real world counterparts.

Forty-eight subjects (twenty-four males and twenty-four females) will participate in a within-subjects design study. Overall Independent Variables (IVs) include World (Real/Virtual) and Design (Good/Poor) with Gender as a blocking variable. Nested within the Design variable, the subjects will either estimate the range to items (Range Estimation) or choose (forced) which of a pair of objects is closer (Relative Range). The Range Estimation IVs are Item (Object/Surface) and the Item's Range from the observer (Near/Far). The Relative Range IVs are Field-of-View (FOV) (Same/Different, i.e., whether or not the subject can see both objects simultaneously in the same FOV) and the objects' Range from the observer (Close/Away). Adjective pair Likert scales, range estimation, relative range forced choice, and elapsed time to answer range questions will be collected as dependent variables.

The second VR validation study involves a proposed redesign of the Crew Interface Coordinator (CIC) console. The CIC console is part of the POCC. The CIC position is analogous to the Mission Control Center's "Capcom" position, communicating directly with the Spacelab payload crew about payload operations issues. In this study, the objective is to develop, assess, and validate VR as a micro-ergonomics analysis tool for considering operational, viewing, and reach envelope requirements in the spatial layout of workstations and worksites and to develop, assess, and validate scaleable user anthropometry attributes. It compares the proposed redesigned CIC console with a Virtual Crew Interface Coordinator (VCIC) console.

An algorithm has been developed to rescale user anthropometric attributes to any desired virtual anthropometry. Thus, a 95^{th} percentile male could view and reach as a virtual 5^{th} percentile female and vice-versa.

Test scenarios will be performed on both a "Fomecor" mock-up of the CIC console and the VCIC console and their results compared to ascertain what, if any, distortions arise in a VW. The test scenarios focus on what one can see from a variety of eye reference points and on what one can touch from a variety of shoulder reference points using a range of real and virtual anthropometric sizes. Results of these analyses are also compared to determine the relative merits of VR vis-a-vis an existing, "standard" Human Factor's tool (i.e., "Fomecor" mock-up).

The objectives of the third study are to develop, assess, and validate VR as a Human Factors design analysis tool for considering operational, viewing and reach envelope requirements in a micro-gravity environment and provide some of the various advantages and disadvantages of reaching and maneuvering in micro-gravity. It compares the results from a "virtual analysis" with a previously conducted analysis relating to the operation of the Electromagnetic Processing Facility (TEMPUS), an experiment to fly on the Second International Micro-gravity Laboratory (IML-2) Spacelab mission.

TEMPUS is a levitation melting facility for processing of metallic samples in an ultra-clean environment. Sample positioning and heating can be controlled separately by two independent RF oscillating circuits. The issue driving the previously conducted analysis was whether a crewmember could adequately control the position of a sample in the facility with controls located in the right half of Rack 10 while monitoring the results on a CRT in the right half of Rack 8. The CRT was co-planar with the rack faces and 42 inches away from the controls. A full-scale part-task "Fomecor" mock-up of both racks was fabricated to determine the crewmembers ability to view the CRT while touching the controls.

A virtual mock-up of racks 8 and 10 has been developed and placed inside of a virtual Spacelab module. A method to enable the various advantages and disadvantages of reaching and maneuvering in a micro-gravity environment has been developed, within existing VR technology capabilities and limitations. In particular, the user can manipulate the attitude of the Spacelab, as a whole, while "grabbing" a handrail, giving the egocentric perception of micro-gravity mobility. Viewing and reach envelope analyses are conducted and the results compared with the previously conducted "Fomecor study." This study also includes the scaleable user anthropometry, developed and assessed in the VCIC study.

MSFC VR APPLICATIONS

Current Applications

The MSFC VR capability has already be utilized in a couple of activities. Both primarily involved immersive visualization of architectural spaces. One supported the recent move of the CAVE Lab into its new quarters. Two different lab layouts were developed and modeled in Swivel 3D. Several of the lab staff then "entered" the virtual labs and evaluated the configurations as both users and visitors. Layouts were modified near real-time and re-evaluated. Based upon these evaluations, one modified layout was chosen and implemented.

In a second activity, support was provided to the 30% design review of the Space Station *Freedom* Payload Control Area (PCA). The PCA will be the payload operations control room, analogous to the Spacelab POCC. Several configurations of the console floor plan layout, large video screens, and Public Viewing Area were modeled in Swivel 3D. Designers, management, and the Public Affairs Office (PAO) utilized the system to immersively visualize the options. PAO evaluated the view from the Public Viewing Area, at various raised floor heights, and performed a preliminary camera viewing analysis, "flying" to various possible camera locations to inspect the composition of the possible camera fields-of-view. The ability to pan and tilt and change "lens" (i.e., narrow to wide angle fields-of-view) in real-time was especially useful.

Proposed Training Applications

Two Spacelab training applications are proposed. The extent to which they are possible using the existing VR technology capabilities and limitations will be determined largely by the earlier validation studies. The implementation of these applications is, therefore, somewhat tentative and will be modified by the results of the validation studies. These proposed applications require a translation capability to translate existing Spacelab Computer-Aided Design (CAD) data files into VR compatible formats.

The first proposed application is related to initial crew and POCC cadre training. As a new crew or cadre member is assigned to a Spacelab mission there is a familiarization phase for both mission independent Spacelab systems and capabilities, and mission dependent payload systems and capabilities.

Early in the mission planning and development process, this training is accomplished in the classroom, and through Spacelab systems and mission documentation. The full-scale Payload Crew Training Complex (PCTC) training mock-up and simulators are not yet available. For personnel assigned later, the PCTC training mock-up may be in place, but access may be limited due to simulator development and other training activities.

In either case, there is a period during which the newcomer must quickly assimilate a large amount of information into his or her concurrently evolving schema or mental model of Spacelab. A tour through a virtual Spacelab may initialize the newcomer and provide insights into system functionality and capabilities. If successful, this could provide a basis for a more accelerated training program and a better integrated understanding of Spacelab systems and payloads.

The essential feature of this application is one or more Virtual Spacelab Modules (VSLMs). Depending on the focus of the "lesson", there may be several VSLMs, each configured to support that lesson objective. For example, Spacelab Program Overview may use standard Spacelab systems in both the long and short modules and perhaps even the "pallets only" configuration. Mission specific training could use a Spacelab systems-only VSLM and/or an

integrated systems/payload VSLM. In addition, each system and payload could be "exploded" to permit visualization of its constituent components and their interrelationships.

The second proposed Spacelab training application, utilizing existing VR technology capabilities, involves using VSLMs during the last nine-to-six months before launch. There are always late changes to on-board stowage. As changes are made, the PCTC Training mock-up is updated. It is desirable to allow the crew the opportunity to tour the mock-up to "see" the latest stowage configuration. This helps to "internalize" the location of items within the Spacelab module. Unfortunately, as the launch date approaches, access to the crew becomes more and more limited, particularly during the last three months.

A VSLM with the updated stowage configuration would enable a more convenient, even remote, method to "visualize" changes in stowage locations. Updated VSLM files could even be electronically transmitted to the JSC for the crew to "tour" on the JSC VR system. Validation of this application, like the previous real-world application, will be based primarily upon subjective data gathered through questionnaires.

This ability to electronically transfer VWs further enhances the familiarization/ initialization training application discussed above. Using both the MSFC and JSC VR systems simultaneously, the users could enter and interact within the same VSLM. This would permit, for example, a "tour guide" for the Spacelab Program Overview or a Mission Specialist accompanied by the stowage manager or a Payload Specialist for the stowage "walk-thru."

Future Applications

A demanding and comprehensive application for VR is support of unplanned Inflight Maintenance (IFM). That is, subsets of the features and VR capabilities required to support this application are used in a variety of other applications. Support to unplanned IFM requires Human Factors analyses (e.g., viewing, reach, and dynamic work envelope analyses), operations development, training, mission support, and even simultaneous participation by physically separated users in the same virtual environment.

An example of an unplanned IFM occurred on Spacelab 3. This actual Spacelab mission experience will also be used for comparison in the validation of this application. The goal would be to actually recreate the IFM environment and operation, then compare this virtual IFM experience with the actual flight experience. This would include reference to video and audio recordings of the on-board operation, written logs, and participation of the actual Spacelab crew involved in the IFM operation.

During Spacelab 3, the Drop Dynamics Module (DDM) developed a problem with a power supply module. One of the three DDM AC power supplies had failed. No procedures or plans had been developed pre-mission for this particularly malfunction. No spare power supplies were stowed. It was decided to cut the wires from the failed power supply and re-route them to one of the other power supplies that could handle the extra load. Procedures had to be developed and validated on the ground and approved by both MSFC and JSC before uplink to the crew. The procedure required removal of the rack front panel before the Payload Specialist (PS) entered head first. Only his legs remained visible outside of the rack. Inside the cramped rack interior, the PS successfully re-wired the power supply modules and continuation of the science objectives resumed.

It is anticipated that enhanced VR would have been capable of supporting many of the activities and analyses that occurred on the ground in support of this unplanned IFM. Viewing analyses, reach envelope analyses, and, with an incorporated anthropometric model, dynamic work envelope analyses can be achieved concurrently with procedure development. Although much of this can be done in an engineering mock-up, VR offers several unique capabilities.

First, VR could provide a timely and safe method to enable the various advantages and disadvantages of reaching and maneuvering in a micro-gravity environment. This includes body attitudes and positions difficult to recreate in a one-G environment. This would be superior to existing methods for simulating micro-gravity because existing methods can not be used in a timely manner and are of limited duration (KC-135) or require ancillary equipment (Neutral Buoyancy Simulator) that can interfere with operations in restricted volumes. Second, VR would permit

anthropometric sizing to reflect the dimensions of the on-board crew. This is particularly useful for operations being planned in relatively tight spaces.

Since a payload IFM procedure has to be approved by MSFC and JSC before it can be implemented, VR would offer the mission and payload managers the ability to visualize the procedure and environment to gain a faster and more in-depth understanding of the operation. This could be accomplished while the managers are sitting at their consoles in the control center. Further, managers at both centers could enter the VW simultaneously to review and discuss the operation. This capability for direct mission support would be unprecedented, though the possibilities are not limited to unplanned IFMs.

Pre-mission operations development and validation could also be carried out in the same manner, even though the rapid turn-around capability of VR is not necessarily a requirement. Pre-mission crew training could use the same VWs developed to support procedure development. This would prove particularly beneficial for operations where the various advantages and disadvantages of reaching and maneuvering in a micro-gravity environment make a difference.

Enhanced VR technologies and techniques could also provide unique capabilities not presently possible with current simulation technologies. Currently, there is no way to practice the logistics of handling multiple, mobile objects in a simulated micro-gravity environment. The duration of the micro-gravity periods on the KC-135 are too short (approximately 30 seconds) and the viscous drag and motion-induced turbulence in water makes neutral buoyant methods unsuitable. VR, with suitable tactile and force-reflective feedback and a physics properties simulator reflecting physical laws concerning motion and collisions, could prove to be a valuable operations development and training tool for applications requiring dexterous, fine-motor movements.

SUMMARY

This paper has described the VR Applications Program at MSFC, including objectives and approaches. Current and planned applications and associated validation approaches were presented.

The three validation studies described above can be started with the current MSFC VR Applications Program resources and capability. The two "real-world" proposed applications will require augmenting current capabilities with commercially available products

Viewing analyses, reach envelope analyses, and dynamic work envelope analyses can be achieved concurrently with procedure development. VR can provide a timely and safe method to enable the various advantages and disadvantages of reaching and maneuvering in a micro-gravity environment. This would be superior to existing methods for simulating micro-gravity because existing methods can not be used in a timely manner and are of limited duration. Even where the KC-135 and/or the Neutral Buoyancy Simulator are appropriate, prior utilization of virtual mockups can result in more efficient use of these micro-gravity simulators. Hardware and operations design can be more mature, resulting in fewer and/or more productive simulator sessions.

Utilizing VR in initial Spacelab crew and POCC cadre training can provide a basis for a more accelerated training program and a better integrated understanding of Spacelab systems and payloads. A virtual Spacelab with an updated stowage configuration can enable a more convenient, even remote, method to "visualize" changes in stowage locations.

Pre-mission and mission operations development, validation, and support can utilize VR. This would prove particularly beneficial for operations where the various advantages and disadvantages of reaching and maneuvering in a micro-gravity environment make a difference.

Virtual Egocenters as a Function of Display Geometric Field of View and Eye Station Point

[1]Joseph Psotka, Ph.D.
U. S. Army Research Institute
ATTN: PERI-IIC
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

(703)274-5540/5545/5569
Psotka@alexandria-emh2.army.mil or psotka@26.1.0.50
FAX: 274-5461

ABSTRACT

The accurate location of one's virtual egocenter in a geometric space is of critical importance for immersion technologies. This experiment was conducted to investigate the role of field of view (FOV) and observer station points in the perception of the location of one's egocenter (the personal viewpoint) in virtual space. Rivalrous cues to the accurate location of one's egocenter may be one factor involved in simulator sickness. Fourteen subjects viewed an animated 3D model, of the room in which they sat, binocularly, from Eye Station Points (ESP) of either 300 or 800 millimeters. The display was on a 190 by 245 mm monitor, at a resolution of 320 by 200 pixels with 256 colors. They saw four models of the room designed with four geometric field of view (FOVg) conditions of 18, 48, 86, and 140 degrees. They drew the apparent paths of the camera in the room on a bitmap of the room as seen from infinity above. Large differences in the paths of the camera were seen as a function of both FOVg and ESP. Ten of the subjects were then asked to find the position for each display that minimized camera motion. The results fit well with predictions from an equation that took the ratio of human FOV (roughly 180 degrees) to FOVg times the Geometric Eye Point (GEP) of the image:

Zero Station Point = (180/FOVg)*GEP

INTRODUCTION

The accurate location of one's virtual egocenter in a geometric space is of critical importance for immersion. Furness (1992) and Howlett (1990) report that immersion is only experienced when the field of view (FOV) is greater than 60 degrees, or at least in the 60 to 90 degree FOV range. Why this should be so is not understood, nor are there theoretical frameworks for beginning to understand this phenomenon. The question is also important for dealing with simulation or motion sickness. Immersion environments are notorious for producing motion sickness, and an inaccurate location of virtual egocenters may be implicated in this noxious effect. Jex (1991) reports that simulator sickness is hardly ever felt with FOV less than 60 degrees (the complement of immersion FOV). Perhaps a key variable is the quality of immersion and the accuracy of self-localization. Informal comments by users of immersion environments have yielded many descriptions of surprising errors of self-localization. As a start this research begins to explore how egocenters are determined from perceptual arrays.

Some work exists that may be helpful to understand the psychology of egocenters (Howard, 1982; Ono, 1981). Kubovy (1986) provides an insightful description of the use of techniques by Renaissance artists to manipulate the location of virtual egocenters, and thus manipulate attitudes and emotions. Franklin, Tversky, and Coon (1992)

---

[1]THE OPINIONS IN THIS PAPER DO NOT NECESSARILY IMPLY OR EXPRESS THE VIEW OF THE U.S. ARMY RESEARCH INSTITUTE (USARI). THIS RESEARCH WAS FUNDED BY THE USARI BASIC RESEARCH OFFICE. I thank Lisa Traub, Harold Goldstein and Jonus Gerrits for their help in constructing the models.

have conducted a long series of experiments examining the cues that control placement of point of view in spatial mental models derived from textual descriptions.

A series of experiments by Ellis (McGreevy and Ellis, 1986; Tharp and Ellis, 1990; Nemire and Ellis, 1991) probably indirectly reflects on virtual egocenters. Ellis and McGreevy (1986) discovered a systematic error in pointing the direction of objects in a virtual display. The error was a function of the geometric FOV of the display. They developed a complex model that accurately predicted these errors on the basis of memory for the size and shape of objects and geometric "distortion" based on linear projections. Tharp and Ellis (1990) provided an explanation based on errors of estimation of the pitch and yaw of the viewing direction used to produce the perspective projection. They argued that people have acquired, through experience of observing the world, a way of determining the effects of viewpoint rotations and perspective transformations. People use this experience to build a "table" of perspective transformations relating target azimuth to projected angle. They then use the wrong table. This is a little like saying that people project themselves at the wrong point, and so it may be possible to find an effect on the location of virtual egocenters in these conditions. The regular shape of the error (see Figure 1) could be produced by an altered location of the virtual egocenter in the display such that, in their experiment, observers felt themselves closer to the objects than the geometry of the scene should have made them feel they were. In other experiments to follow these up, Ellis and his associates found the direction of these errors systematically reversed. The cues that produce these effects are unknown but may have something to do with the relationship of the actual FOV of the display and the computed geometric FOV of the display image (FOVg). When the ratio of FOV/FOVg is greater than 1, the observers may have located the virtual egocenter too near to the objects; and when the ratio of FOV/FOVg is less than 1, the observers may have located the virtual egocenter too far from the objects. It is not clear from their data which case held, but these relationships appear to be appropriate for their results.

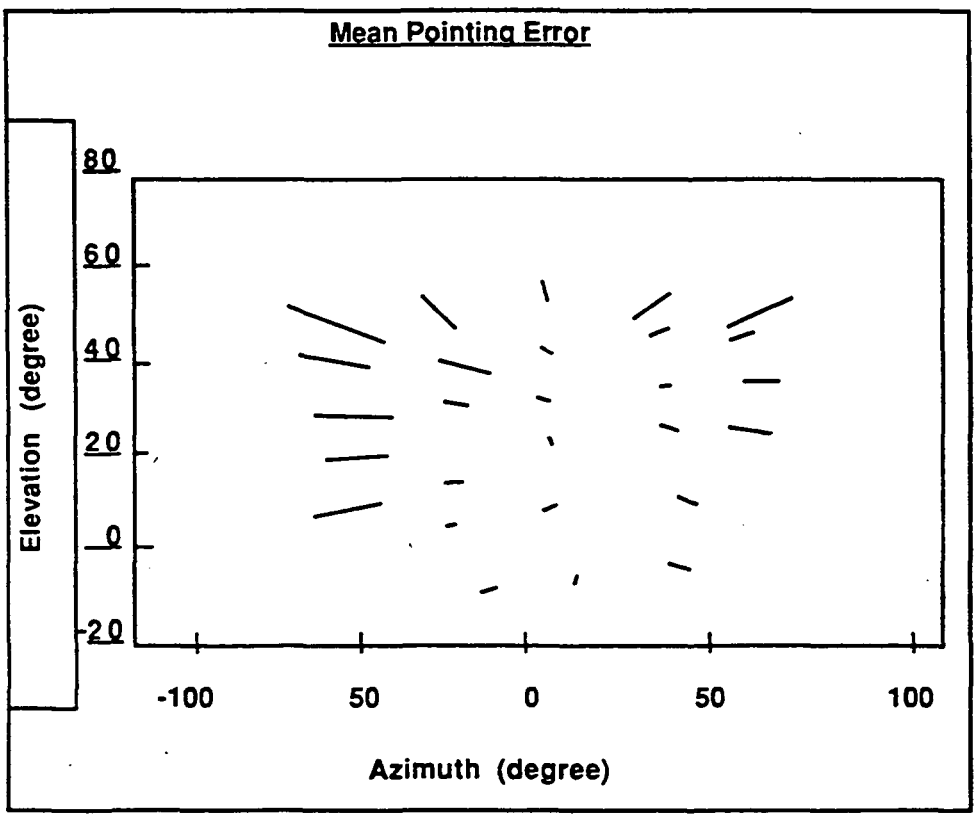


Figure 1. A schematic diagram showing the pattern of errors found by Tharp and Ellis.

Nemire and Ellis (1991) added some evidence for this hypothesis by demonstrating that the enhanced structure of a pitched optic array does bias the perception of gravity-referenced eye level. This finding is a direct replication of Kubovy's arguments about egocenters and Renaissance artists, although on a much smaller scale.

These experiments reported here are an extension of Ellis' work to confirm his findings and extend his interpretation of their source.

STIMULI

An accurate model of an office was constructed using 3D Studio on a 386 PC with VGA graphics. The model contained walls, floor, and ceiling, three tables with computers and displays, two bookshelves with empty shelves, and two wastebaskets in the room. It was rendered with Phong shading at 320 by 200 pixels with 256 colors, and looked like a reasonable cartoon of the actual office holding the equipment (see Figure 2).

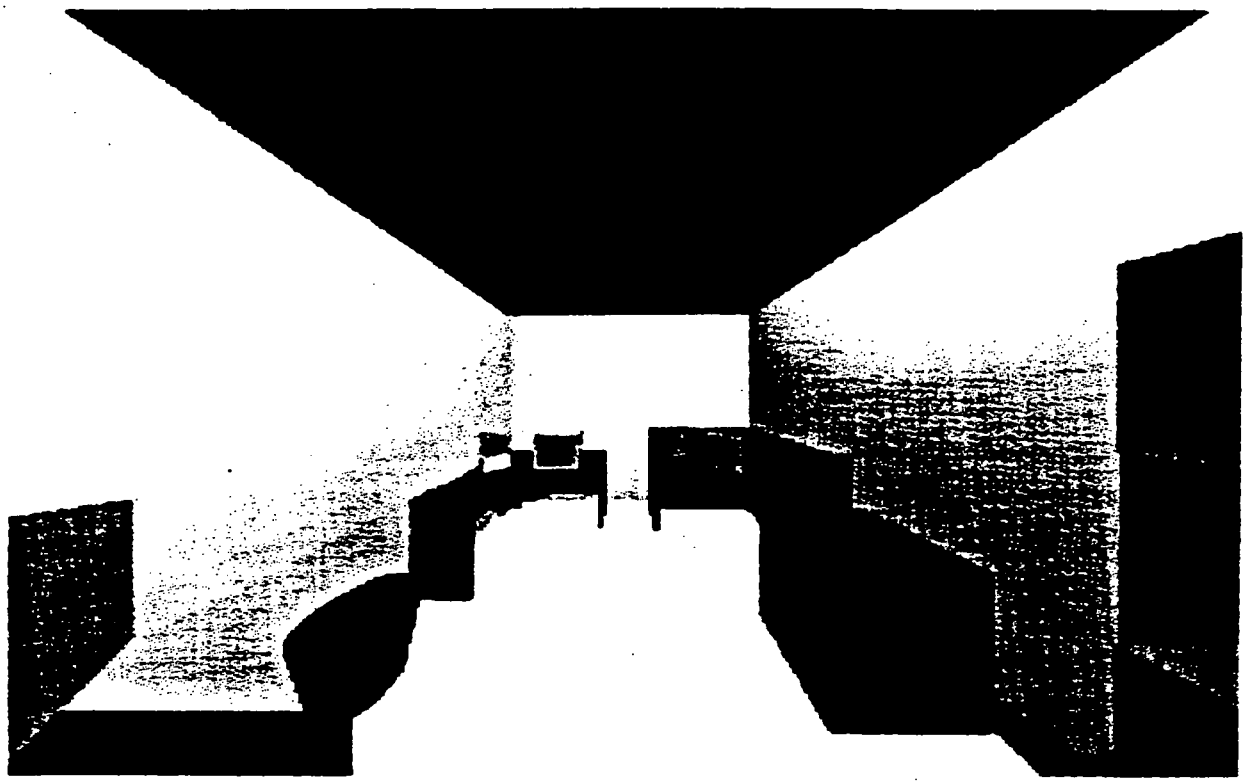


Figure 2. A black and white photocopy of the color screenprint of a 135 mm lens view of the experimental room.

Animations of this model were then created showing a stationary camera located at the geometric center of the room panning slowly 360 degrees around the room. Four animations were created with four different lenses for the scene: 17, 28, 50, and 135 mm. The geometric field of view for each of these lenses was: 140, 86, 48, and 18 degrees, respectively, where 140 degrees is similar to a fish-eye lens and 18 degrees is a telephoto view. The animations were viewed on a flat screen Zenith monitor whose screen dimensions were 190 by 245 mm. Subjects viewed the animations from two locations 800 and 300 mm from the screen. At those sites the screen subtended a FOV of 17 and 45 degrees, approximately. FOV is calculated by 2 times atan(.5 width of monitor/distance of eye point). Although their heads were not restrained mechanically, Ss held their positions reasonably well.

The geometric eye point of each of these lenses was 40, 140, 290, and 800 mm in the room. These projection points are independent of the viewer's location. They are dependent on the actual size of the viewing screen. Thus the two viewing sites for the subjects corresponded approximately to the geometric eye points for the lenses of 135 and 50 mm.

PROCEDURE

Subjects were asked to view the animations binocularly, with corrected vision, and determine the location and path of the camera in each animation. The room was normally lit by recessed ceiling lights. They were told that the animation was of the very same room where they sat. They were shown a bitmap hardcopy of the room from an overhead view and asked to trace the path of the camera on it. (See Figure 3.) They were not specifically told that the geometric "camera" was mathematically or "theoretically" stationary in the animations.

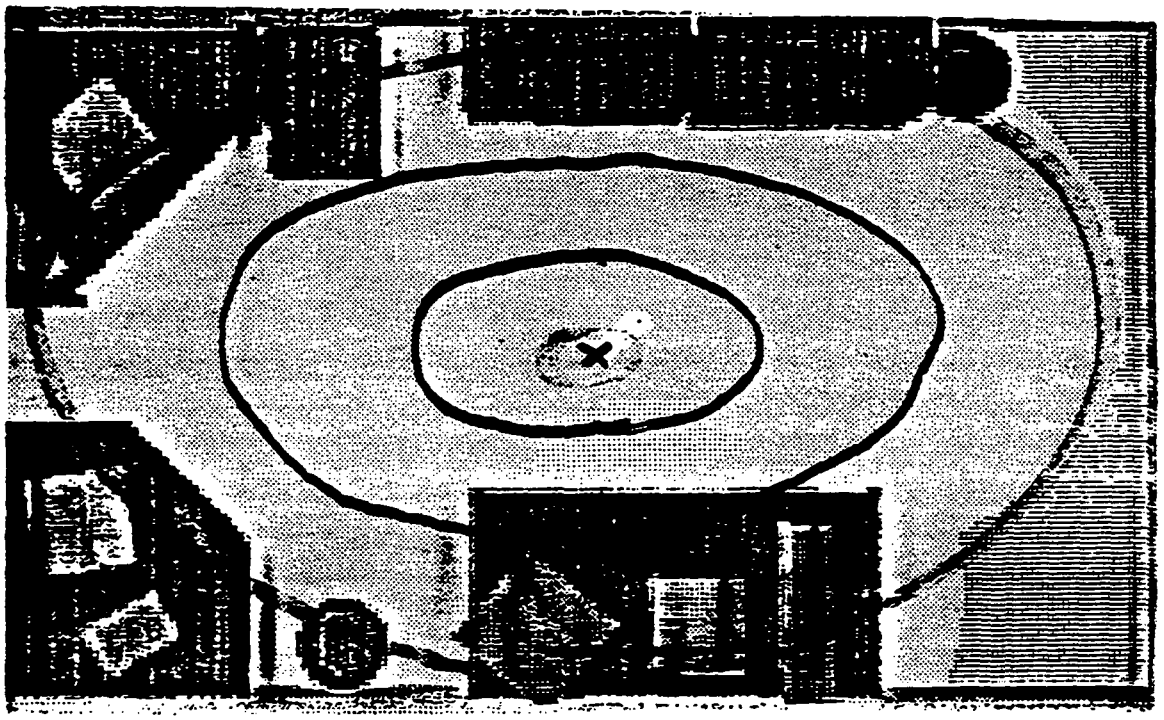


Figure 3. An overhead view of the experimental room. Subjects traced the camera path as shown with representative traces derived from the four views of the experiment.

Fourteen students and colleagues with a variety of psychological training served as experimental subjects without pay. Ten of these Ss were asked at the end of the experiment to select for each animation the viewing station that produced the least camera motion.

RESULTS

In general, the subjects had no difficulty describing the apparent paths of the camera as they saw it as oval paths of varying eccentricity centered on the geometric center of the room. The diameters of the ovals varied with the focal length of the lens. The radius of these ovals in mm for each animation and station point are given in Table 1. A positive number indicates that the virtual egocenter was behind the observer's eye station point; and a negative number indicates the camera was stationed in front of observer's eye station point. A zero would indicate the geometric center of the room, the observer's eye station point.

Both viewing station points yielded similar relationships between the radius of motion and the geometric FOV of the animations (see Table 1), but the viewing station point of 800 mm produced a concave function, whereas the viewing station point of 300 mm produced a convex function.

By interpolating these points, one can determine where Ss would have seen no camera motion.

For the 800 mm view site, the paths had 0 diameter with 60 degree FOV or a geometric eye point of approximately 250 mm.

For the 300 mm view site, the paths had 0 diameter with 80 degree FOV or a geometric eye point of approximately 150 mm.

Table 1 Radius of Camera Path (and distance from eye station point (ESP)) as a function of FOVg and Station Point

	Geometric Field of View of Room			
Station Point	**18**	**48**	**86**	**140**
Group 1 - 300 mm	-541.3	-278.7	83.7	912.5
Group 2 - 800 mm	-785.0	-77.5	416.3	538.8

The mean locations for the station points with least camera motion were 9112, 1092, 291, and 53 mm from the monitor for the four geometric fields of view of 18, 48, 86, and 140 degrees, respectively, whose geometric eye points were 800, 290, 140, and 40 mm.

DISCUSSION

It appears that the egocentric station point is affected by the geometric FOV of the displayed image; the relationship between the viewing site and the geometric eye point, and the actual FOV of the image. The location of the egocenter is NOT experienced as the same as the geometric station point of the camera under any of the conditions of these experiments. It appears that the location of the egocenter and the geometric station point of the camera would coincide only with a 180 degree FOV display.

It appears that the least egocenter motion was produced in these experiments with a FOV that varied between 48 and 86 degrees, curiously close to the required limits in order to experience satisfactory immersion (Furness, 1992). However, this appears to be an accident of the stimulus conditions in this experiment. Egocenter motion was almost completely nullified for the 17 mm lens (and 140 FOV) at a viewing site of 50 mm ; and for the 28 mm lens (and 86 FOV) at a viewing site of 290 mm. The other two animations did not appear to have a station point that yielded 0 camera path; although the station points selected by subjects did appear to reduce the absolute value of the camera path substantially. This finding needs to be explored further. It may be related to the finding that immersion is not satisfactory with displays that are less than 40 degrees because no satisfactory compromise exists between the conflicting cues of linear perspective and the visual system's need for a visual field of 180 degrees to find a stationary egocenter.

Frame Effects. Ss repeatedly remarked that they appeared to be using the frame of the monitor as the frame of reference of their retinal field. When asked to describe what was happening, they said they appeared to be contracting their field of attention to the frame of the monitor, and then treating that as if it were their entire 180 degree visual field. If they were in fact doing this at a processing level, then the geometric eye point of the animation would not be determined by the size of the monitor, but by the virtual size of their expanded attentional field, roughly 180 degrees. The geometric eye point would then be expanded by a similar ratio, yielding the enlarged path of the camera with smaller FOVg. In fact, if one proposed that the zero station point is determined by the product of the animation's geometric eye point (GEP) times the ratio of 180/FOVg, one could calculate the predicted station points for zero camera motion.

Zero Station Point = (180/FOVg)*GEP

For this experiment these predictions are: 8000, 1100, 287, and 50 mm. quite close to the empirical values of : 9112, 1092, 291, and 53 mm. This seems to indicate that when the FOVg is 180 degrees, the egocenter is located correctly, but when the FOVg is less than 180 degrees, the egocenter is displaced proportionately.

Size of the Effect. Although the eye station points for reduced motion are described by a simple relationship, the width of the camera path is not clearly related to any of the variables. For most of the variables we have only two points (the 300 and 800 mm eye station points) and that is not enough to specify the functional relationship. Qualitatively it appears that for arm's length ESPs width of path is approximately linearly related to FOVg, with small FOVg leading to the feeling of being very close up to the objects, and large FOVg leading to the feeling of being very distant from the objects. However, this effect of FOVg is moderated by eye station point in complex ways for which we need much more data. The effect of eye station point (ESP) appears to be affected by its relation to the geometric eye point (GEP) too. If ESP is closer to the object than GEP, especially when the FOVg fills the eye's field of view; then the virtual egocenter appears to be closer than the GEP. However, this effect appears only to hold for the large FOVg, larger than 45 degrees. It is also true that only with these FOVg is there a point where camera motion was found to be zero. For smaller FOVg , the camera motion could be minimized but not reduced to zero.

Effect of Size. The familiar size of objects might be affecting this illusion of virtual egocenter placement. Objects like chairs and tables and monitors have roughly expected sizes or degrees of visual angle from every distance. Egocenter location could be computed from that information as well as the perspective lines of the image or the kinetic depth effects of the turning motion. It is not clear what the role of size is. At the smallest FOVg in this experiment (18 degrees) objects had a 1:1 size ratio with objects in the real world; yet the impression was not one of being the real world distance from them, but of being very close to them, 785 mm or 98% of the distance closer, in fact. Still it is very easy to redo this experiment with objects that have no familiar size; and even to remove linear perspective cues by using balloons in a spherical room (See Figure 4). Preliminary explorations with these figures indicates some differences in the perception of relative motion, namely it is very difficult to perceive these figures as stationary, but not apparently in the main findings of this paper.

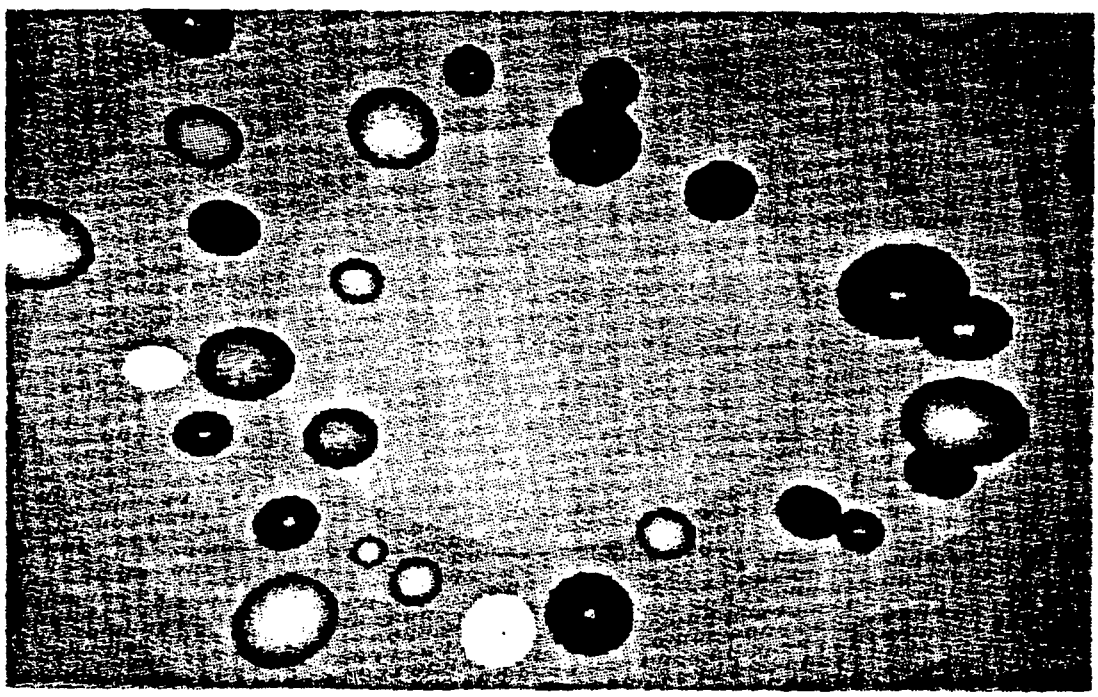


Figure 4. Top view of a room full of round balls suspended in space.

Size constancy effects may in fact be related to the egocenter effects found in this paper. A brief review of the literature (Hochberg, 1978; Yonas and Hagen, 1973) indicates no general awareness of the possible effects of FOV size or FOVg size on the perception of distance to objects or object size constancy. This appears to be a promising avenue of research. In fact there is very little research on the nature of virtual space as perceived from geometrically created views of everyday scenes.

Simulation Sickness. Although no one became nauseous, everyone reported some degree of discomfort with viewing the displays larger than 60 degrees FOVg, especially the largest. Several people asked to look away from the 140 FOVg display to reorient themselves during the experiment.

CONCLUSION

Clearly much work remains to be done if we wish to specify exactly how people interpret constructed geometric displays to select their egocentric viewing spot. Yet this work is very necessary if we wish to be able to create three-dimensional models that have the power to generate a truly satisfying and natural immersion experience.

For psychological theory, this research opens the possibility of dealing quantitatively with very abstract constructs, like virtual egocenters, in ways that were either impossible or very difficult without the new VR technologies. Clearly parametric studies need to be carried out in detail to create a nomograph of functions relating egocenter to FOVg and viewing station points. This pilot work suggests that even very close viewing station points such as those with head mounted displays (HMDs) are not immune to illusions caused by FOV that are smaller than 180 degrees. Their possible implication in more severe phenomena like simulator sickness, or less severe discomfort and dislike of HMDs, is only one further direction that needs exploration. It is clear, for instance, that these sorts of egocenter illusions adapt out very quickly in a VR environment. However, after adaptation is more or less complete, are there still physiological conflicts that can be detected in response to the conflicting cues of linear perspective and reduced FOV? Are there aftereffects that return to the real visual world?

Other, broader theoretical issues that need exploration are higher order cognitive implications of these new relations between multiple realities. When we view the animation apparently rotating on the monitor, somehow we build up a model of the room. That model is also somehow projected into the same space as the real room that we occupy. While viewing the animation, we have both an egocenter in real space, and a virtual egocenter in the space of the animation. It appears from these experiments that those egocenters interact with each other so that we feel some conflict as we rotate and move in one and remain stationary in the other. What are the long term effects of this conflict? For instance, if parts of the visual field, or even half or more of it were blocked out and replaced with active noise, would observers begin experiencing something like lateral neglect? What would happen if we decorrelated color patches from objects? We know for instance, that color is processed in separate pathways from form (Livingstone and Hubel, 1987). Using VR technologies, could these separate pathways be made explicit and what would its effects be? What are the memory implications for conflicts between one reality and another? What are the physiological processing correlates of immersion? How can MRI technologies be used to provide converging evidence for these findings? These are only some of the interesting psychological questions that need a firm base of experimental data to rest the initial creation of exploratory theoretical frameworks.

REFERENCES

Ellis, S. R. (Ed.), (1991) **Pictorial Communication in Virtual and Real Environments.** London: Taylor and Francis.

Franklin, N., Tversky, B., and Coon, V. (1992) **Switching points of view in spatial mental models.** Memory & Cognition, 20(5), 507 - 518.

Furness, T. (1992) Personal communication.

Hochberg, J. E. (1978) **Perception.** Englewood Cliffs, NJ: Prentice-Hall, Inc.

Howard, I. P. (1982) **Human Visual Orientation.** New York: Wiley.

Howlett, E. M. (1990) Wide angle orthostereo. In Merritt, J. O. and Fisher, S. S. (Eds.) **Stereoscopic displays and Applications.** Bellingham, WA: The International Society for Optical Engineering.

Jex, H. R. (1991) Some criteria for teleoperators and virtual environments from experiences with vehicle/operator simulation. In Durlach, N. I., Sheridan, T. B., and Ellis, S. R. **Human Machine Interfaces for Teleoperators and Virtual Environments.** Moffett Field, CA: NASA Conference Publication 10071.

Kubovy, M. (1986) **The psychology of perspective and Renaissance art.** Cambridge: Cambridge University Press.

Livingstone, M. S. and Hubel, D. H. (1987) Psychophysical evidence for separate channels for the perception of form, color, movement, and depth. **Journal of Neuroscience, 7,** 3416 - 3468.

McGreevy, M. W. and Ellis, S. R. (1986) The effect of perspective geometry on judged direction in spatial information instruments. **Human Factors, 28,** 439 - 456.

Nemire, and Ellis, S. R. (1991) Optic bias of perceived eye level depends on structure of the pitched optic array. Presented at the **Psychonomic Society,** San Francisco, CA.

Ono, H. (1981) On Well's (1792) law of visual direction. **Perception and Psychophysics, 30,** 403-406.

Yonas, A. and Hagen, M. (1973) Effects of static and kinetic depth information on the perception of size in children and adults. **Journal of Experimental Child Psychology, 15,** 254-265.

Cognitive Factors Associated with Immersion in Virtual Environments

[2]Joseph Psotka, Ph.D. and Sharon Davison
U. S. Army Research Institute and Catholic University

Joseph Psotka, Ph.D.
Chief, Smart Technology for Training
U. S. Army Research Institute
ATTN: PERI-ICC
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

(703)274-5540/5545/5569
Psotka@alexandria-emh2.army.mil or psotka@26.1.0.50
FAX: 274-5461

Immersion into the dataspace provided by a computer, and the feeling of really being there or "presence", are commonly acknowledged as the uniquely important features of virtual reality environments. How immersed one feels appears to be determined by a complex set of physical components and affordances of the environment, and as yet poorly understood psychological processes. Pimentel and Teixeira (1993; p. 15) say that the experience of being immersed in a computer-generated world involves the same mental shift of "suspending your disbelief for a period of time" as "when you get wrapped up in a good novel or become absorbed in playing a computer game." That sounds as if it could be right, but it would be good to get some evidence for these important conclusions. It might be even better to try to connect these statements with theoretical positions that try to do justice to complex cognitive processes. The basic precondition for understanding Virtual Reality (VR) is understanding the spatial representation systems that localize our bodies or egocenters in space (Franklin, 1992). The effort to understand these cognitive processes is being driven with new energy by the pragmatic demands of successful virtual reality environments, but the literature is largely sparse and anecdotal (cf. Benedikt, 1991; Ellis, 1992; Furness, 1992; Laurel, 1992; Pimentel and Teixeira , 1993; Witmer and Singer, In Preparation).

Although the VR literature pays a great deal of lip service to the perceptual psychology of J. J. Gibson, there is little in the ecological perception framework that might suggest sources of individual differences in the quality of immersion. Yet, anecdotally, there appear to be wide differences in how well people react to these exotic environments.

In a first step to gather more information, a very short set of questions were compiled and sent to users of virtual reality environments over the internet on the virtu-l listserv. The questions were carefully designed to cover cognitive factors that have been raised in the literature on virtual reality environments. By and large, the literature has not raised a very detailed or analytic set of questions, so many questions we developed deal with issues that have some face value only. For instance, Pimentel and Teixeira (1993; p. 105) offer the following set of factors as instrumental for deep immersion: interactivity, fast update rate, high image complexity, engaging, 3D sound, head-mounted display; stereoscopic; large field of view, and head tracking. None of these are cognitive factors, although they admit that a "holistic technique might be necessary because the experience of immersion is more than just the mere sum of its parts."

---

[2]THE OPINIONS IN THIS PAPER DO NOT NECESSARILY IMPLY OR EXPRESS THE VIEW OF THE U.S. ARMY RESEARCH INSTITUTE (USARI). THIS RESEARCH WAS FUNDED BY THE USARI BASIC RESEARCH OFFICE. We thank Peter Legree, Sandy Calvert, and Marc Sebrechts for many fruitful discussions.

DESIGN

Stimuli

The factors we considered were grouped into two categories:
A. Susceptibility to immersion; and B. Quality of immersion.

A. Susceptibility to immersion:
- I. Imagination
  - Strength of visual imagination
  - Dreaming
  - Self-consciousness
  - Daydreaming
  - Ability to willingly suspend disbelief
  - Depth of involvement in books, theater, etc.
- II. Vivid Imagery
  - Dreaming
  - Prior expectations about the virtual reality environments
  - Claustrophobia
- III. Concentration and Attention
  - Attentional filtering
  - Cognitive conflict in holding two recursive immersions
  - Spatial Navigation
  - Claustrophobia
- IV. Self-control
  - Self-control
  - Active participation and catharsis

B. Quality of immersion.:
- I. Affordances of the VR Environment for Immersion:
  - Object Persistence
  - Sensory Completeness
  - Interactivity
  - Realism of the environment
  - Amount of lag or delay
  - Size of field of view
  - Accuracy of egocenter or body image location
  - Pleasure and exhilaration from the novelty of the experience
- II. Distractions from the Real Environment:
  - Presence of others; sounds, tactile
  - Fatigue and irritation by bulky equipment
  - Restrictiveness of the equipment
  - Similarity of Real World to VR World
- III. Physiological effects
  - Simulator sickness
  - Disorientation after immersion
- IV. Other Effects
  - Prefer immersion alone
  - Surprise when HMD removed

Many other factors could be assessed, but for the first effort it was important to keep the questionnaire as short as possible to elicit as many voluntary responses as possible, and to begin winnowing these factors in a sensible way.

The constructed questionnaire used 12 questions for the susceptibility component (that might also be used as a pretest in any experimental setting) and 11 questions to measure the quality of immersion. All questions were constructed on a five point categorical (Likert) scale for responses. The questionnaire is included in the appendix A, with some changes and new questions that have been added through the preliminary analysis reported here. The questions were asked so that roughly half of them were constructed to relate positively with depth of immersion and hal, negatively. A key was constructed to convert all questions to positive relationships.

Subjects

Fifteen respondents whose gender is unknown had some experience in a variety of virtual reality environments ranging from homebred PC systems to state of the art centers at UNC, BBN, Chicago, and the HIT Lab at U. Washington. The modal virtual reality environment was the W. Industries' Virtuality arcade with 7 respondents. Most of the respondents had less than one hour of experience in the virtual reality environments. In addition 8 responses came from individuals with no virtual reality environment experience, who only answered the questionnaire dealing with susceptibility.

These are very small numbers of respondents for the factor analyses used in this exploratory study. The results must be viewed as provisional, reflecting some of our biases as well as true relationships between cognitive components and depth of immersion. Ideally, 100 respondents for each group would be welcome. This would even begin to allow comparisons among the different VR environments used by the respondents. We urge you all to complete the questionnaires in Appendix A, or have your students complete them, and send them to us for further analysis.

RESULTS

PreTest

On the susceptibility to immersion questionnaire, no significant differences were found in the scores of the group with no VR experience and the group with some VR experience, using a standard analysis of variance ($F = 0.45$, 1,21 df; $p = 0.51$). The two groups' scores correlated significantly ($R = .63$; 9 d.f.; $p < .05$). On the whole, they often dream in color; think that the quality of VR is like TV; occasionally feel claustrophobic or uneasy in small spaces; feel somewhat self conscious speaking in public; have sometimes cried watching a good, sad movie; find a good book very engrossing; have sometimes stayed up late to finish a good book; daydream often; are often able to read in a moving car, boat, or plane; can often tell where they are when some one else is driving; feel it is somewhat important to be completely in control; and have occasionally been told that their name was called when watching TV or reading a book, but have not recollection of hearing it.

PostTest

On the actual experience of virtual environments, the critical question that asked how completely people believed they were part of the virtual environment, yielded only a modest "somewhat". However, on the whole, they felt "very" exhilarated by the experience. More than half the respondents reported they were "very" or "totally" exhilarated. The difference may be that more than half of them felt a little or more woozy or nauseous from the experience. Given their short stays in these virtual environments, that may be a reason for concern.

Most of them felt somewhat surprised by the direction they were facing when they removed the HMD. On the whole, this left no aftereffect, and they reported that they were only a little disoriented by the experience. They continued on the whole to be somewhat aware of the direction they faced in the real world during the VR experience. They occasionally or sometimes thought about the other person(s) in the real world with them there. Most of them felt that other person(s) made no difference to their enjoyment of the experience.

Almost all of the respondents complained that the VR devices restricted their movements somewhat or a lot. The bulkiness of current equipment should make that a surprise to no one.

Most of them reported that when they turned their back on an object in the VR environment, it was still there (although some commented that they might not easily find it again). This is a cognitive skill called "object permanence" or _existence constancy" by psychologists. It is part of a Piagetian test of maturation for children, and may be a good index of the maturity of VR environments too.

There was considerable variability in how flat and missing in depth the VR world appeared, as might be expected from the range of equipment, but on the whole it was only a little missing in depth.

Correlations

Many significant correlations were found ($r = .47$, $df = 13$; $P = .05$). Two sets of correlations might be interpreted. One set looks at the questions within each of the two questionnaires for common factors of importance. The other looks at the correlations between the two questionnaires to try to guess at causal or dynamic connections among susceptibility and the quality of immersion. Principal components factor analysis was used with oblique varimax transformations to select the factors reported below.

Causal or dynamic connections between susceptibility and the quality of immersion: The results were examined for each question that asked about the quality of the experience for a correlation that significantly affected it. These correlations are reported in terms of rules below.

Overview: Immersion was most affected by how claustrophobic someone is. The more claustrophobic you are the more often you think about the other person(s) in the real world with you there; the more you complained that the VR devices restricted your movements; the less often you felt objects were still there when you turned your back on them; the less exhilarated by the experience you were; and the more nauseous or woozy you felt. Clearly this is a danger signal for arcade makers of VR environments. Rather than providing access to a wide open cyberspace, current equipment is still evoking claustrophobic feelings of enclosure and restriction. While almost half of the respondents reported no wooziness or nausea after the experience, this is a pretty extreme response, and arcade makers should flag the fact that more than half reported a "little" or "some" wooziness or nausea. Apparently, those who are somewhat susceptible to claustrophobia, and in this small sample 9 of the 15 reported some sensitivity, have this fear triggered by current equipment.

Depth of immersion was most strongly predicted by whether or not you dream in color, by how often you cry at good, sad movies, and by how effectively you filter out other distractions while reading a book or watching TV. You are more deeply immersed if you dream often in color and you ignore others when they call you while reading or watching TV. You also are more deeply immersed if you are more exhilarated by the experience. There appear to be two psychological factors dominant in predicting depth of immersion. One is the willingness to accept another reality and the willingness to make the effort it takes to participate in it fully and satisfactorily. This factor is dominated by crying at movies, staying up late, and being engrossed by a book. It suggests that childhood pretend play and make believe were a significant part of a person's experience. The other factor depends on the ability to shut out the distracting effects of the real world. This factor is implicated in ignoring others when they call you while you are watching TV, and being able to shut out the real world when you wake up dreaming. This factor may be seriously tested as more complex VR systems become available, that integrate virtual worlds with reality into a new augmented reality.

Exhilaration and enjoyment of the experience is also increased if you are not claustrophobic, and you do not think self control is very important. Two individuals were strongly claustrophobic, and their exhilaration scores were markedly lower than the others'. Exhilaration is also increased if you believed you were deeply immersed and that objects behind your back continued to exist.

Summary of Susceptibility for Immersion

Relationships among the questions are given as rules in Appendix B.
Several clusters are revealed by these intercorrelations.

Cluster A (35% of the Variance) Imagination: Questions 4, 6, 7 and 8 intercorrelated. These all deal with self - consciousness, daydreaming, and how engrossing a book is, even staying up late to finish reading it. Attentional control and ignoring distractions may be the dominant factors. Especially in arcades, wearing the gear required for VR experiences may make some people feel foolish and disturb their immersion. Being willing and able to use your imagination seems to be a common thread.

Cluster B (29 % of Variance) Vividness of Imagery and Claustrophobia: Questions 1, 2, 3, and 9 intercorrelated. 1 and 2 deal with how often you dream in color and how realistic you think the VR world will be visually, while 3 and 9 relate to motion sickness and claustrophobia. The link between those two is unknown. Perhaps there is an unknown relationship between being able to remember dreams in sufficient detail to remember color, (since we all dream every night, probably in color) and our ability to deal with motion sickness and claustrophobia. Our favorite explanation, derived from discussions with our colleague Peter Legree, postulates that the common thread is an ability to shut out or exclude the unwanted effects of the environment. Obviously this is important for claustrophobics when they try to control their claustrophobia in small spaces.

Shutting out reality is also important for those who want to remember their dreams. Most dream memories are only accessible in the first few moments of waking, a special twilight period when both realities are accessible to people. If you can shut out the world during this brief period, you will remember your dreams more clearly, including the fact that they are in color. Most of us who have tried to recall our dreams over an extended period have found that our ability to hold on to this twilight period when dreams are accessible increases with practice, but some of us continue to find it easier than others. It appears that parts of this ability are a skill that can be learned, but other parts may be controlled by psychological processes that are more difficult to change. It appears that they may also affect the depth of immersion in today's crude VR environments.

Why this factor contains expectations of the photorealism of the VR world is also not immediately explicable. Again, our favorite story is that the expected realism of the VR world depends on the vibrancy of dream worlds compared to the real world. Those who can remember their dreams in vivacious detail may be expected to have higher and similar expectations for the VR experience. These expectations also seem to influence the depth of the immersion experience, since those who expect too much appear to be disappointed.

Cluster D (19% of Variance) Self Control: Questions 5 and 11 intercorrelated. These both deal with some components of self control and attentional control. Crying at movies deals with issues of immersion that are under voluntary and imaginative control, rather than the involuntary and ecological or environmental imposition of immersion that we normally experience in our day to day interaction with the environment. Crying at movies also demands the willingness and skill to be able to achieve catharsis (Laurel, 1992) from the fantasy experience of the VR world. Crying at the movies is a pleasurable experience, not an unhappy one. It does not come easily, but demands that viewers participate actively in the movie, and engage its themes and personal events in a powerful way that releases emotions safely, and without fear of personal injury. It demands involvement and participation in an interactive way, and may be related to how satisfying childhood pretend play and make-believe were. Movies and VR experiences are thrilling and emotionally satisfying because they are safe and they are fantasy. If you did not know that it was a fantasy, the events would be horrifying and deeply unsatisfying; yet, because they are fantasy the experience can be deep and meaningful, but quite pleasurable too. Unlike dreaming, where it is important to shut out the distractions of the real world, cathartic release in movies depends on actively engaging and participating in the movie, and actively suspending your disbelief about its illusion. It is interesting that this skill appears to be more positive and affirming than the negative act of shutting out distractions, that appeared in the first factor.

Cluster C (17% of the Variance): Questions 10 and 12 intercorrelated. These both deal with being able to do two things at the same time: pay attention to where you are going when someone else is driving; and listening for others who call you while watching TV. This factor may hinge on the skill of holding two or more realities in mind at the same time. Being able to concentrate and disregard the opinion of others seems to be a common thread.

Table 1. Intercorrelations of each of the PreTest Subscales with the Total PreTest Score

	Pre-A	Pre-B	Pre-C	Pre-D
Pre-A	1			
Pre-B	.093	1		
Pre-C	.064	.263	1	
Pre-D	.657 *	-.087	-.36	1
PreTest Total	.38	.736 *	.473 *	.234

* $p < .05$

Summary of Depth of Immersion Responses:

The responses and intercorrelations for each question are given in Appendix C.

Several clusters are revealed by these intercorrelations.

Cluster A (32% of Variance) Distractibility: Questions 2, 3, 5, and 8 intercorrelated. They all deal with potential distractors outside of the VR environment that could diminish the depth of immersion and the feeling of presence. The direction you face in the real world, how often you think of others in the real world, how restricted your movement is, how flat and missing in depth the world appears, all potentially distract you from the immersion experience. The flatness or lack of depth question was intended to pertain to the VR world, but perhaps people took it to apply to the real world after coming out of the VR experience.

Cluster B (26% of Variance) Willingness to Suspend Disbelief: Questions 6, 7 and 10 intercorrelated. These all deal with those components of the automatic or ecological affordances of the environment that control the depth of immersion, and since they include the prime measure of depth of belief in being part of the environment, they must be more central to the whole immersion experience: sense of object permanence, exhilaration in the experience: these are all involuntary effects of deep immersion. This is perhaps why they all correlate with dreaming in color, since that too is an involuntary immersion and acceptance of another reality, the reality of unconscious imagery and dreams. If one can accept the reality of these still crude cartoons and interactive computer displays, then it may not take an act of self control and will power, but a state of mind that is open and susceptible to altered states of consciousness, like dreams. Notice that dreams are not like daydreams, a meandering of conscious ideas, and it is instructive that daydreaming does not correlate with any of these measures of ecological immersion. It is only the basic involuntary and biological but complex act of dreaming, a total immersion in a hallucinogenic other reality that provides the best measure of immersion into VR environments.

Cluster C (24% of Variance) Concentration: Questions 1 and 4 intercorrelated. They ask how much more enjoyable the experience would have been without anyone else around, and how surprised you were when the HMD was removed. Apparently this did not affect the depth of immersion of people who were distracted by the presence of others, nor make them think of others, so it is not related to their distractibility. However, some people must find fantasy in an immersion environment more enjoyable when they are alone, perhaps because they are self-conscious and cannot concentrate on the experience. Surprise on removing the HMD gives a measure of the concentration on the task of immersion.

Cluster D (19% of the Variance) Simulation Sickness Effects: Questions 9 and 11 intercorrelated. These both deal with how disoriented and how woozy or nauseous you feel. These side effects of immersion also have a strong influence on the depth of the immersion experience. Surprisingly, we did not ask about the effects of lag or apparent body position, so dizziness and nausea were only related to being aware of the direction you faced in the real world.

Table 2. Intercorrelations of each of the PostTest Subscales with the Total PostTest Score.

	Post-A	Post-B	Post-C	Post-D
Post-A	1			
Post-B	.542	1		
Post-C	.479	.34	1	
Post-D	.587	.295	.549	1
Post Total	.815	.792	.761	.686

Overall Correlations between PreTest and PostTest

The correlation between the Susceptibility measure and the Total Immersion scale was .82 (13 d.f.; $p < .01$). This significant correlation suggests that our measure of immersion susceptibility offers the minimal requirement for prediction: significant correlation of the intended measure.

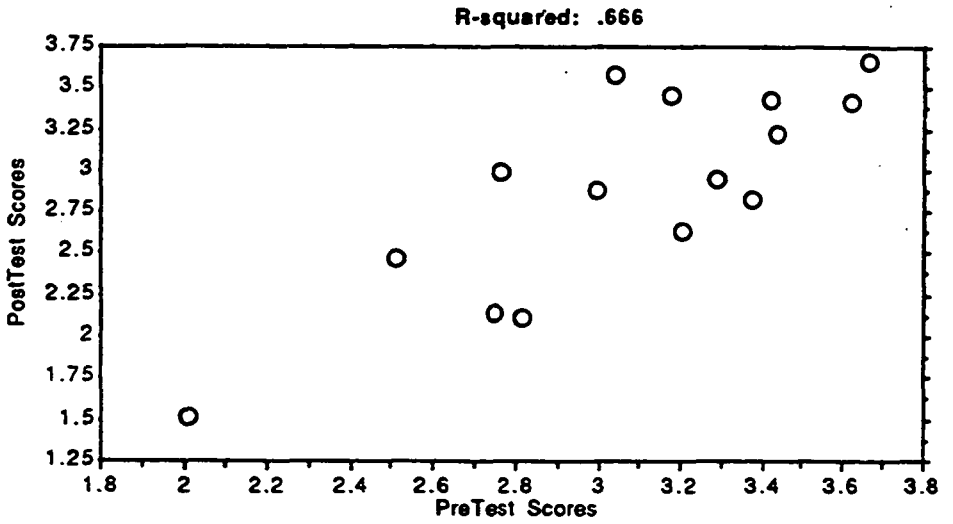


Figure 1. Scattergram showing correlation between PreTest and PostTest Scores

DISCUSSION

The most interesting component of these results is that immersion can be seen as a dual phenomenon: on the one hand dependent on implicit or subconscious biological processes and skills that invoke our cognitive machinery only when the affordances of the ecological setting is suitable; and on the other hand dependent on voluntary attentional skills that depend on self control, self-consciousness, distractibility, attention, expectations, and will power. These two factors (implicit versus conscious control of immersion) are captured in one correlation: Immersion is most complete if you dream in color. How these two components interact is a powerful mystery. The two implicit and conscious components appear to do different things and may not be capable of affecting each other directly. These factors come out so strongly that they are visible in all three sets of correlations: in the susceptibility factors, in the immersion factors, and in their intercorrelations. The fact that they are so strong in the susceptibility factors may indicate that people were interpreting these questions in a very special way, since they had already experienced VR immersion. A larger sample may introduce differences between those who are and are not experienced in VR, on the responses to the susceptibility questionnaire. It must be remembered that these results came from people who answered both questionnaires at the same time, not in the way they were inteded to be used: before and after experiencing immersion. However, this standardization group is still important for the kinds of intercorrelations it demonstrates.

Implicit factors have a kind of dominance. It may not be possible to overlook certain ecological preconditions for immersion if they are violated, no matter how intent one may be to have a deep immersive experience. For example, if the lag between intention and visual feedback of perceived hand movement in the VR world is very large, no amount of mental filtering of the image is likely to reduce that lag; and it may lead to varying degrees of simulator sickness. Or, as another example, if your visual system tells you that your egocenter or body image is in one location, but your kinaesthetic senses locate it somewhere else, there may be no way to override or integrate these two positions by cognition alone. Longer term processes of learning and adaptation that change the cognitive machinery may be required.

From these results, it appears that immersion in a VR world is not like being immersed in a book or a good movie. It appears to be more like remembering your dreams. Unlike a book or a movie there are strong visual affordances for immersion in your dreams. There you are almost always the agent of action and interaction, and there is only one world in which you are immersed. When you look at a picture or build up a representation of the space described in a book (Tversky, Franklin, and Coon, 1992) that fact that you are sitting or standing in another (real) space never quite gets forgotten, so there must always still be some sort of conflict between the two representations of your egocenter.

This conflict is most evident when you look at a picture, either a photograph or a painting. Particularly in paintings, the viewpoint of the painting is often quite controlled and striking (Kubovy, 1986). In Davinci's famous painting of the last supper, for instance, the viewpoint is elevated some 5 meters above the floor of the room where most people stand to view it. Art critics have suggested that this is to give one the feeling of elevation and levitation as one views the painting. Most viewers are unaware of the conflict between their egocenter as people in the room and their apparent egocenter as viewers in the painting. The space of the painting and the space of the room are shared easily. This is unlike VR immersion where entering the VR world depends on blocking out the real world almost completely. This happens in the real world when someone holds a mask in front of their face too. Holding a picture in froont of your face has a disturbing effect a maks does not, because the space of the picture and the space of your face are so different that it is not really possible to reconcile them. Surely conflicts between visual and kinesthetic egocenters are being resolved continuously in the real world, since all of our systems are slightly in error in our interactions with the world.

It is very important to proceed with the categorization of the cognitive and perceptual components of immersion, and continue this analysis. Take for instance the primary cognitive process of specifying an egocenter within a cognitive spatial representation system. Clearly the location of the egocenter is computed from many visual cues, derived from the structure of the optic array. In a VR system many differences between an ecologically mimetic representation and the real world could yield either less presence, or inaccurate immersion, or both. For instance, a narrower field of view could destroy the experience of immersion as others have suggested, not in and of itself, but because it also created a displaced egocenter, or created the illusion of viewing the world through a porthole or goggles from outside the world.

Entering the VR world appears to require the full and complete use of psychological spatial representation processes that normally appear to be able to allocate only one spatial egocenter at a time. Franklin, N., Tversky, B., and Coon, V. (1992) report that readers will try to encompass only one described scene at a time, and cannot easily integrate two different locations and scenes. Holding on to reality in a VR experience appears to be equally difficult, and so the distractibility of the environment and the equipment used for the VR representation become critical factors in the depth of the VR experience.

Fleshing out the cognitive and perceptual components of immersion promises to be a long but rewarding task. It is particularly rewarding for training and educational purposes because we already know that so much of cognitive representation is in the form of mental models for understanding complex systems. Vr promises to turn knowledge into experience and make education and training much more direct and effective.

REFERENCES

Benedikt, M. (Ed.), (1991) **Cyberspace: First Steps**. Cambridge, MA: The MIT Press.

Ellis, S. R. (Ed.), (1991) Pictorial Communication in Virtual and Real Environments. London: Taylor and Francis.

Franklin, N., Tversky, B., and Coon, V. (1992) Switching points of view in spatial mental models. **Memory & Cognition, 20(5)**, 507 - 518.

Howlett, E. M. (1990) Wide angle orthostereo. In Merritt, J. O. and Fisher, S. S. (Eds.) **Stereoscopic displays and Applications**. Bellingham, WA: The International Society for Optical Engineering.

Kubovy, M. (1986) **The psychology of perspective and Renaissance art**. Cambridge: Cambridge University Press.

Laurel, B. (1991) **Computers as theater**. New York: Addison-Wesley Publishing Co.

Pimentel and Teixeira (1992) **Through the looking glass**. Intel.

APPENDIX A: Two Questionnaires

Please answer these questions. They are intended to provide us some insight into the process of "immersion" and the nature of "presence". If you have already had some VR experience, please answer the second part too, but first indicate what kind of equipment you used and how often you have had direct experience of VR:
Equipment: _____
Frequency of Experiencing VR: #times_____#hours:_____
Purpose of Vr Experience: Recreation; Research; Training

PRETEST: A questionnaire to try to measure someone's susceptibility to experiencing a pleasurable immersion into VR.

1. Do you dream in color?
Never Seldom Sometimes Often Always

2. How realistic do you think this experience will be visually?
PhotoReal LikeMovies LikeTV LikeComics Blurred

3. Do you feel claustrophobic or uneasy in small spaces?
Never Seldom Sometimes Often Always

4. When you speak in public, how self-conscious do you feel?
Totally Very Somewhat A Little Not At All

5. How often have you cried watching a good, sad movie?
Never Seldom Sometimes Often Always

6. How engrossing is a good book?
Totally Very Somewhat A Little Not At All

7. How often have you stayed up late to finish a good book?
Never Seldom Sometimes Often Always

8. How often do you daydream?
Never Seldom Sometimes Often Always

9. Can you read in a moving car, boat, or plane?
Never Seldom Sometimes Often Always

10. Do you keep track of where you are when some one else is driving?
Never Seldom Sometimes Often Always

11. How important is it to be completely in control?
Totally Very Somewhat A Little Not At All

12. How often have people told you they called your name and you have no recollection of hearing them while watching TV, or reading a book?
Never Seldom Sometimes Often Always

Additional Questions:

13. How realistic and detailed are your usual visual images, such as of your car or of a good friend?
Totally Very Somewhat A Little Not At All

14. How much do you prefer improvisation and exploration over following directions?
Totally Very Somewhat A Little Not At All

15. How often as a child did you play pretend or make - believe?
Never Seldom Sometimes Often Always

POSTTEST: A questionnaire to try to measure the depth of someone's immersion into VR.
1. When you removed the Head mounted display, how surprised were you at the direction you were facing?
Totally Very Somewhat A Little Not At All

2. During the immersion how aware were you of the direction you faced in the real world?
Totally Very Somewhat A Little Not At All

3. How often did you think of the other person(s) in the real world with you?
Never Seldom Sometimes Often Always

4. How much more enjoyable would it have been to have the immersion experience with no one else in the room?
Totally Very Somewhat A Little Not At All

5. How restricted was your movement by the VR devices?
Totally Very Somewhat A Little Not At All

6. How completely did you believe you were part of the virtual environment?
Totally Very Somewhat A Little Not At All

7. When you turned your back on an object in the virtual environment, was it still there?
Never Seldom Sometimes Often Always

8. How flat and missing in depth did the VR world appear?
Totally Very Somewhat A Little Not At All

9. How woozy or nauseous did you feel after the experience?
Totally Very Somewhat A Little Not At All

10. How exhilarated did you feel after the experience?
Totally Very Somewhat A Little Not At All

11. How disoriented did you feel after the experience?
Totally Very Somewhat A Little Not At All
Additional Questions:

12. How disturbing was the lag or delay between your movements in the real world and the VR world?
Totally Very Somewhat A Little Not At All

13. How often did you feel your body image was in the wrong place in the VR world?
Never Seldom Sometimes Often Always

14. How responsive was the environment to your movements?
Totally Very Somewhat A Little Not At All

15. How natural and realistic was any object motion?
Totally Very Somewhat A Little Not At All

16. How much narrower was the field of view than normally?
Totally Very Somewhat A Little Not At All

17. How completely could you survey or search the environment visually?
Totally Very Somewhat A Little Not At All

18. How realistic was this experience visually?
PhotoReal LikeMovies LikeTV LikeComics Blurred

19. How completely did you adapt to the VR world and its special characteristics?
Totally Very Somewhat A Little Not At All

20. How completely were all your senses engaged by the VR world?
Totally Very Somewhat A Little Not At All

21. How often did images of the VR world intrude in your daily life or your dreams after the experience?
Never Seldom Sometimes Often Always

APPENDIX B

Summary of Susceptibility for Immersion Responses:

Question 1. If you tend to dream in color you believe the VR world will be more photorealistic, and you seldom feel claustrophobic.

Question 2. If you expect the VR world to be more cartoon - like and blurred you tend not to dream in color, and you tend not to be able to read in a moving car.

Question 3. If you tend to be claustrophobic you rarely dream in color.

Question 4. If you are self-conscious you find a good book engrossing and you seldom daydream.

Question 5. If you often cry at a good, sad movie you stay up late to finish a good book, you can read in a moving car, and you find it important to be in control.

Question 6. If you tend to find a good book engrossing you tend to stay up late to finish a good book, you cry more often at good, sad movies, and you find it important to be in control.

Question 7. If you tend to stay up late to finish a good book you tend to find a good book engrossing, you cry more often at good, sad movies, and you find it important to be in control.

Question 8. If you tend to daydream more often you are not self conscious and you find a good book engrossing.

Question 9. If you tend to be able to read in a moving car you tend to dream in color, you believe the VR world will be more photorealistic, and you seldom feel claustrophobic

Question 10. No significant correlations.

Question 11. The more important you feel it is to be completely in control. the more you tend to cry at good, sad movies, the more engrossing a good book is, and the more often you stay up late to finish a good book.

APPENDIX C

Summary of Depth of Immersion Responses:

Question 1. If you are more surprised by the direction you are facing when the HMD is removed, you become disoriented by the experience, and when object constancy is accepted in the virtual environment.

Question 2. If you tend to be more aware of the direction you are facing in the real world, while you are immersed, you think of other people in the real world, you tend to feel that the VR devices are restrictive, and you tend to become woozy or nauseous.

Question 3. If you tend to think about the other person(s) in the real world with you there you tend to be more aware of the direction you are facing in the real world, you tend to feel that the VR devices are restrictive, and the world tends to feel flat and missing in depth.

Question 4. No significant correlations.

Question 5. If you find the VR devices restrictive you tend to be more aware of the direction you are facing in the real world, you tend to think about the other person(s) in the real world with you there, and the world tends to be less flat and missing in depth

Question 6. If you tend to believe completely that you were a part of the VR world, your object constancy is strong in the VR environment, and the experience tends to be very exhilarating.

Question 7. If your object constancy is strong in the VR environment you are more surprised by the direction you are facing when the HMD is removed, you tend to believe completely that you were a part of the VR world, and the experience tends to be very exhilarating.

Question 8. If the world tends to be less flat and missing in depth you tend to be more aware of the direction you are facing in the real world, while you are immersed, you tend to think about the other person(s) in the real world with you there, and you find the VR devices restrictive.

Question 9. If you become more woozy and nauseous you tend to be more aware of the direction you are facing in the real world, while you are immersed.

Question 10. If the experience tends to be very exhilarating you tend to believe completely that you were a part of the VR world and your object constancy is strong in the VR environment.

Question 11. If you are more disoriented you are more surprised by the direction you are facing when the HMD is removed.

APPENDIX D

Summary of Depth of Immersion Related to Susceptibility for Immersion:

Question 1. You are more surprised by the direction you are facing when you remove the HMD if you tend to dream in color. This also happens when you become disoriented by the experience, and when object constancy is accepted in the virtual environment.

Question 2. You tend to be more aware of the direction you are facing in the real world , while you are immersed, if you are claustrophobic and tend to stay up late reading books. This also happens when you think of other people in the real world, if the VR devices are restrictive, and if you become woozy or nauseous.

Question 3. You tend to think about the other person(s) in the real world with you there if you are claustrophobic, if you daydream, and if you lose your sense of location when others drive you.

Question 4. You would find the VR experience more enjoyable alone if you are self-conscious, and feel an need for self-control.

Question 5. You find the VR devices restrictive if you are claustrophobic and self-conscious.

Question 6. Immersion is most complete if you dream in color and are not distractible.

Question 7. Your object constancy is strong in the VR environment when you are not claustrophobic, and you do not need to be in control.

Question 8. The VR world tends to be less flat and missing in depth when you expect it to be more cartoon like and blurred.

Question 9. You become more woozy and nauseous when you rarely dream in color, when you are more claustrophobic, and it is important to be in control.

Question 10. The experience is more exhilarating when you are less claustrophobic and less distractible.

Question 11. You are more disoriented when you often cry at a good, sad movie; and the more important it is to be completely in control.

Self-Attitude Awareness Training: An Aid To Effective Performance In Microgravity and Virtual Environments?

Donald E. Parker*, D. L. Harm** and Faith L. Florer*

*136 Benton, Miami University, Oxford, Ohio 45056

**SD5, Johnson Space Center, Houston, Texas, 77058

ABSTRACT

This paper describes ongoing development of training procedures to enhance self-attitude awareness in astronaut trainees. The procedures are based on observations regarding self-attitude (perceived self-orientation and self-motion) reported by astronauts. Self-attitude awareness training is implemented on a personal computer system and consists of lesson stacks programmed using Hypertalk with Macromind Director movie imports. Training evaluation will be accomplished by an active search task using the virtual Spacelab environment produced by the Device for Orientation and Motion Environments Preflight Adaptation Trainer (DOME-PAT) as well as by assessment of astronauts' performance and sense of well-being during orbital flight. The general purpose of self-attitude awareness training is to use as efficiently as possible the limited DOME-PAT training time available to astronauts prior to a space mission. We suggest that similar training procedures may enhance the performance of virtual environment operators.

INTRODUCTION

The self-attitude awareness training project described in this paper is designed to support the Preflight Adaptation Training (PAT) Project currently being pursued at NASA's Johnson Space Center. The overall goal of the PAT Project is to reduce hazards and discomfort associated with space motion sickness and to mitigate entry and post-landing self-orientation and locomotion disturbances. The specific goal of self-attitude awareness training is to enhance astronauts' ability to manipulate spatial mental representations and to perform mental simulations of movement through the unusual environment of microgravity. We hypothesize that these training procedures will enhance astronauts' abilities to determine their own orientation, motion and location. Successful training should facilitate adaptation to microgravity and transitions between microgravity and normal gravity on earth.

Training procedures are based on reports regarding self-attitude from astronauts in microgravity and in the Device for Orientation and Motion Environments Preflight Adaptation Trainer (DOME-PAT). Sensory information about spatial relationships in and movement through microgravity and virtual environments differs from that which is ordinarily detected in a normal terrestrial environment. Objects that ordinarily exhibit a constant visual polarity on earth, such as computer monitors and people, may be seen in unusual orientations. Relationships between visual scene polarity and the observer's internal body axes may differ from those ordinarily experienced. Changes in scene orientation and motion as a consequence of real or virtual locomotion may conflict with orientation and motion changes, or the absence thereof, detected by the astronaut's inertial receptors, including the vestibular and somatosensory receptors. Perceived self-orientation and self-motion with respect to the environment, a phenomenon analogous to what pilots call "attitude awareness," may be disturbed by discordant spatial information. These disturbances are likely to have negative effects on the performance and comfort of astronauts.

Astronauts' reports

During space Shuttle missions, astronauts enjoy looking though a window on the flight deck at the earth as it passes "under" the spacecraft. This is possible because the orbiter is usually oriented so that it flies "upside-

down;" that is, the roof of the orbiter's cabin is oriented toward the earth. Astronauts often "lie" on the cabin ceiling when they look though the window. After a few minutes, they report feeling as though they are lying prone with the earth "down." However, when they look back into the spacecraft, the cabin seems to be "upside down." Early in the mission, this is disturbing. After a few days in orbit, however, most astronauts readily shift between earth-referenced down and cabin-referenced down. This and several related observations suggests that for most astronauts self-attitude awareness improves as the mission progresses.

Recent observations suggest that astronauts develop enhanced ability to perform mental rotations during orbital space flight. Berthoz (1992) reported a study of mental rotation while subjects were exposed to the stimulus rearrangement of microgravity during a Russian space mission. Cosmonauts were trained to asymptotic mental rotation performance using a procedure described by Shepard and Metzler (1971) prior to flight. These cosmonauts exhibited significantly improved mental rotation performance during the mission relative to that observed preflight.

Prior to a recent mission an astronaut who had flown once eight years previously participated in a series of DOME-PAT training sessions. In this apparatus, graviceptor output is held constant by keeping the trainee stationary or permitting movement around an axis orthogonal to gravity. A visual scene produced by a computer-generated imagery (CGI) system and representing a complex enclosed space is presented to her/him via projectors and a dome/screen. Scene movement depends on attempted head movements by the trainee and/or inputs to hand controllers. However, gravity information transduced by the otolith and somatosensory receptors does not change, thereby achieving graviceptor stabilization. (Details regarding the rationale for and implementation of the PAT apparatus can be found in Parker, 1991, and Harm and Parker, in press.)

During training the astronaut practiced using the hand controller to move virtually through a simulated Spacelab. He practiced moving along the walls and ceiling, attempting to view those features as a floor. He practiced moving into the dark tunnel, which connects the Spacelab to the middeck, then turning around, re-emerging into the Spacelab, and identifying his orientation after re-emergence. Following the space mission, he returned to the DOME-PAT and attempted to perform the same activities.

This astronaut's performance in the simulator apparently was affected by his experience in microgravity. Preflight, he reported some orientation and motion difficulties in the virtual Spacelab environment. He found it difficult to view the Spacelab walls as a floor and he reported that the hand controller was difficult to use. One day after the mission, he remarked that it was easy to perceive the ceiling or walls as a floor and that "locomotion" through the virtual Spacelab using the hand controller was "intuitive." A week after landing, he reported greater difficulty in mentally rotating the virtual Spacelab; also, using the hand controller to move about in the Spacelab became less intuitive than it had been immediately post flight.

METHOD

Stacks

Self-attitude awareness training is being implemented using on a personal computer (Macintosh Quadra 700 4/230 with 20 MB RAM) and a color monitor (NEC 5FG). "Lessons" are being programmed using Hypercard stacks with Macromind Director movie imports. Four set of stacks are currently being developed: (1) overview – purpose and general procedures, (2) object orientation and motion, (3) self- (eye-point) orientation and motion, and (4) eye-point recognition

General procedure

"Clicking on" a TITLE CARD results in presentation of a randomly selected animation or scene (stimulus) from a subset of possible stimuli. The stimulus is followed by presentation of a RESPONSE CARD displaying four written descriptions of possible stimuli, only one of which is correct. The trainee selects his/her best guess regarding the correct stimulus description with a key stroke. Response accuracy (correct or incorrect) and reaction time are recorded. When a preset number of trials (stimuli) have been completed, a RESULTS CARD

summarizing response accuracy and latency is presented and the trainee may repeat the current lesson, proceed to the next lesson or quit.

Object orientation and motion

The purpose of these stacks is to familiarize the trainee with a 3-D coordinate system for describing orientation and motion. Stimuli consist of animations of a mannequin (nutcracker doll). The animations illustrate rotations (yaw, pitch, roll) and translations (X, Y, Z) from an initial "upright and facing forward" cardinal orientation. Yaw rotations (left or right), pitch rotations (forward or rearward) and roll rotations (left or right) are of 90^0, 180^0 or 270^0 magnitudes. Translations include X-axis (forward or rearward), Y-axis (left or right) and Z-axis (headward or footward).

Four lessons are included in the object orientation and motion stacks:
lesson 1-- object orientation and motion from a cardinal position; stimuli consist of rotations and translations from an initial "upright and facing forward" cardinal orientation;
lesson 2 -- object orientation and motion from a non-cardinal position; stimulus initial orientations are 90^o, 180^o, or 270^o rotated in pitch, roll and yaw from the cardinal orientation;
lesson 3 -- object orientation and motion, complex motion; stimuli include combinations of rotations and translations starting from the cardinal orientation;
lesson 4 -- object orientation and motion, complex motion starting from non-cardinal positions; stimuli include combinations of rotations and translations starting from non-cardinal orientations.

Eye-point orientation and motion

The purpose of these stacks is to familiarize the trainee with a 3-D coordinate system for describing eye-point orientation and motion. Stimuli consist of scene animations representing movement of the trainee's eye-point in the Spacelab. Animations include yaw scene rotations (left or right), pitch scene rotations (forward or rearward) and roll scene rotations (left or right) of 90^0, 180^0 or 270^0 magnitudes starting from a cardinal orientation (facing the tunnel entrance). X-axis scene translations (forward or rearward), Y-axis scene translations (left or right) and Z-axis scene translations (headward or footward) are of two magnitudes, 50 cm or 90 cm.

Four lessons are included in the eye-point orientation and motion stacks:
lesson 5 -- eye-point orientation and motion from a cardinal position; stimuli include scene rotations and translations starting from a cardinal orientation (facing the tunnel entrance);
lesson 6 -- eye-point orientation and motion from a non-cardinal position; stimuli include scene animations starting from non-cardinal orientations.
lesson 7 -- complex eye-point orientation and motion; stimuli include combinations of rotations and translations starting from the cardinal orientation.
lesson 8 -- complex eye-point orientation and motion from a non-cardinal position; stimuli include combinations of rotations and translations starting from non-cardinal orientations.

Eye-point recognition

The trainee will be presented with an object animation from the Object Orientation and Motion Lessons followed by four still images of the Spacelab interior. The trainee's task is to select the Spacelab image that corresponds to the object's current eye point.

Training evaluation

Training efficacy will be evaluated using a performance task in the DOME-PAT and by evaluating astronaut performance and sense of well-being during orbital flight. Specifications for the DOME-PAT evaluation task are incomplete but will include the following. The trainee will enter the virtual Spacelab from the tunnel in a randomly determined orientation (facing aft, facing forward, rolled left, and so on. The trainee's task is to move virtually, using the hand controller, through the Spacelab to a specified location and to report the status of a

display at that location. Evaluation during orbital flight will employ a variation of a currently approved Detailed Supplementary Objective (DSO 468).

EXPECTED RESULTS

We anticipate the following results: lesson performance, as assessed by percent correct response and response latency, will depend on (a) rotation complexity -- single axis rotation easier than multiple axis rotation, (b) relationship between axis of rotation and everyday activity -- yaw easier than pitch easier than roll, (c) rotation direction -- pitch back easier than pitch forward, and so on; subject's trained with the procedures described above will exhibit improved performance relative to untrained subjects in a DOME-PAT evaluation task; astronauts trained with the above procedures prior to flight will report more rapid development of self-attitude awareness inflight than will untrained astronauts, as assessed by the modified DSO 468 procedure noted previously.

DISCUSSION

Self-attitude awareness may be based on the activity of a perceptual system that receives inputs from visual, somatosensory and vestibular receptors (Gibson, 1966; Parker, 1991). Activity in this perceptual system permits the subject to answer the questions "where am I" (within an enclosed space) and "how am I moving?" Perception (P) of self-orientation and self-motion with respect to the environment may be dependent on the ability to detect and integrate information regarding three vectors: gravity/linear acceleration - G, visual scene polarity - VS, and internal vectors aligned with the trainee's head or trunk Z axis - IZ. A fourth determinant of perception is the observers' expectations - E - based on his/her own actions. As a first approximation, self-orientation perception may be considered as the weighted sum of these vectors:

$$\text{Equation 1. } P = W_1{*}G + W_2{*}VS + W_3{*}IZ + W_4{*}E$$

The major value of self-attitude awareness training may be to "loosen" the normal (1 G) congruence between these vectors and/or to alter vector weightings.

Mechanisms of self-attitude awareness

Research on self-attitude awareness training may elucidate processes and mechanisms examined in cognitive and perceptual psychology including mental rotation, cognitive maps, shape perception learning and affordance-effectivity.

Mental rotation. The face of a well-known person is not so easily recognized when his or her photograph is presented upside-down. This observation suggests the following question: How do people recognize familiar shapes when those shapes are in unfamiliar orientations? This and related questions have been reviewed by Howard (1982, Chap. 14) and by Fink and Shepard (1986).

One line of research suggests that in order to recognize shapes that are presented in unusual orientations people must perform mental rotations. Shepard and Metzler (1971) undertook a study in which observers were presented with pairs of figures representing three-dimensional objects. Some pairs represented the same object in different orientations while other pairs represented different objects. The observer's task was to determine whether the figures represented the same or different objects. As suggested by Howard and Templeton (1966), performance of this task may require observers to perform some internal operation equivalent to rotating the memory image of one figure in order to align it with the other figure.

As noted previously, Berthoz (1992) recently reported that cosmonauts exhibit enhanced mental rotation ability during orbital flight. We suggest that the following may be important for understanding Berthoz's finding. In microgravity, cosmonauts can be in orientations and move in ways that are ordinarily not possible on earth. Mental rotation is important for efficient goal-directed locomotion. In microgravity, as on earth, one must orient in order to locomote efficiently. The Shepard and Metzler procedure may be viewed as "passive;" (i.e., beyond pressing a response key whose location is known, no action on the part of the subject is required). It seems unlikely that this passive procedure would invoke the same neural operations as would active locomotion. Learning complex procedures often seems to elicit improved performance on the simpler, component procedures.

For example, learning calculus greatly improves the ability to perform algebraic operations. Similarly, learning to perform the complex mental rotations associated with locomotion starting from unusual initial orientations may facilitate performance on a passive mental rotation task. Regardless of the specific mechanisms underlying improvement in mental rotation ability during exposure to microgravity, it is likely that such enhanced ability would lead to increased self-attitude awareness.

Cognitive maps. The pioneering work of Tolman (1932) led to the recognition that animals and people are able to form "cognitive maps" -- internal representations of the spatial layout of an environment. Ordinarily cognitive maps are illustrated from a plan view, for example, a floor plan as seen from above. However, one has only to imagine a familiar building to recognize that a plan view "picture" may not describe adequately retrieval of information from the spatial representations in one's head. Rather, as one imagines the spatial relationships between landmarks, the process seems more like locomotion, like moving one's eye-point though the imagined environment, than like looking at a map or a picture.

The suggestion that internal map processing includes a mental simulation of locomotion/action is consistent with Boer's (1991) report that the time required to reorient with respect to locations on a memorized geographical map increases with the angular displacement of the required reorientation up to angles of 135^{o}. Similarly, Kosslyn (1983) reported that the time required for a subject to scan points on a memorized map increases as a function of the physical distance between the locations specified by the experimenter.

Consider the internal map problem confronting an astronaut in microgravity. The spacecraft middeck floor plan includes an airlock, a stairway and stowage lockers. For an "upright" astronaut looking toward the airlock from the lockers, a leftward yaw movement is necessary to "climb" the stairs to the flight deck. For an "inverted" astronaut in the same location, a rightward yaw movement is required to move toward the stairs. Due to their extensive experience during mission simulations prior to flight, astronauts acquire internal maps representing the shuttle middeck and flight deck and the processing of those internal maps may include a mental simulation of normal locomotion (walking). Locomotion modes and initial orientations prior to locomotion differ greatly in microgravity. Astronauts push and float rather than walk; locomotion may be initiated from an "inverted" orientation near a ceiling rather than "upright" on the floor. Consequently, new internal maps and mental simulations of locomotion may be required for rapid, accurate real locomotion in microgravity, and these may lead to improved self-attitude awareness.

Shape perception. Self-orientation depends on visual scene information including visual polarity, linear perspective, perspective transformations as a consequence of eye-point changes, shading, texture gradient and occlusion.

The accuracy and latency of self-attitude awareness responses may depend on the subject's ability to learn to detect "generic" surfaces (shapes). Nakayama and Shimojo (1992) note that visual shape stimuli are fundamentally ambiguous; i.e., a given physical shape may generate an infinite set of image shapes on the retina depending on the subject's eye-point with respect to the physical object. They suggest that shape recognition is determined by the "principle of generic image sampling." Some image shapes are generic: they have a high probability of occurring during normal locomotion as the subject changes his/her eye-point with respect to physical objects. Other images are labeled "accidental" because they have a lower probability of occurring.

Nakayama and Shimojo suggest further that shape recognition/perception is determined by generic images through a Bayesian-like conditional probability perceptual learning processes. They propose specifically that people learn conditional probabilities of the form p (Sn | Im), where Im is the generic image associated with a particular shape - Sn) and that perception is determined by learning which shape image that is most frequently associated with a particular physical shape, that shape's most generic image.

Nakayama and Shimojo's analysis assumes that locomotion normally takes place in a gravitational environment and ignores the fact that many objects are visually polarized (i.e., have a "normal" orientation with respect to gravity). The set of eye-points from which an subject ordinarily views objects as well as the orientation of the objects themselves is restricted. For a freely floating subject, a new set of "generic" images may be produced as a consequence of the fact that eye-points are no longer restricted by gravity. For example, a tree generates a generic

image on earth that may be labeled "lateral" (i.e., the image a child produces when asked to draw a tree). In microgravity, that same tree generates two generic images, one "lateral upright" and the other "lateral inverted."

The basic issue considered here differs from that addressed by Nakayama and Shimojo. They are concerned with object perception, whereas this paper is concerned with self-attitude awareness or what might be called eye-point perception. However, both classes of perception may be determined by an subject's learning of conditional probabilities for generic images. Following Nakayama and Shimojo's model, the goal of self-attitude awareness training may be described as enhancement of the ability to detect and respond appropriately to generic images produced by polarized shapes in environments where the subject's locomotion is not restricted by gravity.

Direct perception and affordance-effectivity. Based on the seminal work of Gibson (1966), ecological psychologists have proposed two new basic concepts: direct perception and affordance-effectivity. Direct perception theory (see Wertheim, 1990) implies that the visual world consists of a structured pattern of light, an *optic array* that includes fundamental structural features called *invariants*. According to this approach, motion perception is understood as the process of *picking up* invariants from the optic array. As suggested above, direct perception theory can readily be extended to self-orientation and self-motion detection by inertial receptors.

Affordance-effectivity has been described as the properties of the environment that support goal-directed activities such as sitting, walking, and so on (Shaw et al., 1990). We consider affordance-effectivity as a subclass of mental simulations which are among our most common conscious experiences. For example, we simulate social interactions with our children and colleagues in the form "if I say X, she is likely to reply Y, then I might suggest Z, etc." Mental simulations regarding action, including locomotion, are examples of affordance-effectivity. For example, on earth the Spacelab ceiling has no important affordance for action; however, in microgravity that same ceiling affords "pushing off" to float across the environment. Alterations of self-attitude awareness in microgravity may include development of new action/environment mental simulations (affordances) that are unique to that environment.

We are currently developing evaluation procedures designed to examine alternative predictions the foregoing mechanisms of self-attitude awareness.

ACKNOWLEDGMENTS

This project is supported by Grant NAG 9-446/Basic from the National Aeronautics and Space Administration to Miami University. We thank K. Davino, C. Haymaker, J. Jandi, N. Skinner and A, Wolff for their assistance.

REFERENCES

Berthoz, A. (1992). Mental rotation in mircogravity. Presentation at the XVIIth Barany Society Meeting, Dobris, Czechoslovakia, June, 1992.

Boer, L. C. (1991). Mental rotation in perspective problems. *Acta Psychologica, 76,* 1-9.

Finke, R. A. & Shepard, R. N. (1986). Visual functions of mental imagery. In K. R. Boff, L. Kaufman & J. P. Thomas, *Handbook of Perception and Human Performance,* Chap 37, New York: John Wiley & Sons.

Gibson, J.J. (1966). *The Senses Considered as Perceptual Systems.* Boston: Houghton Mifflin Co.

Harm, D. L. & Parker, D. E. (in press). Preflight adaptation training for spatial orientation and space motion sickness. *Journal of Clinical Pharmacology.*

Howard, I. P. (1982). *Human Visual Orientation,* New York: John Wiley and Sons.

Howard, I. P. & Templeton, W. B. (1966). *Human Spatial Orientation,* New York: John Wiley and Sons.

Kosslyn, S. M. (1983). *Ghosts in The Mind's Machine: Creating and Using Images in the Brain.* New York: Norton.

Nakayama K. & Shimojo S. (1992). Experiencing and perceiving visual surfaces. *Science; 257*:1357-1363.

Parker, D. E. (1991). Human vestibular function and weightlessness. *Journal of Clinical Pharmacology, 31*, 904-910.

Shaw, R. E., Kugler, P. N., Kinsella-Shaw, J. (1990). Reciprocities of intentional systems. In R. Warren and A.H. Wertheim (Eds.), *Perception and Control of Self- Motion*, Hillsdale, New Jersey: Lawrence Erlbaum Associates, pp. 171-218.

Shepard, R. N. & Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science*, 171, 701-703.

Tolman, E. C. (1932). *Purposive Behavior In Animals And Men.* New York: Appelton-Century-Crofts.

Wertheim, A. H. (1990). Visual, vestibular, and oculomotor interactions in the perception of object motion during egomotion. In R. Warren and A. H. Wertheim (Eds.) *Perception and Control of Self-Motion*, Hillsdale, New Jersey: Lawrence Erlbaum Associates, pp. 171-218.

Tutoring Methods and Strategies in LEAP

Frank Linton, Brigham Bell, Charles Bloom, and Edwin Norton
U S WEST Technologies, Suite 270
4001 Discovery Drive, Boulder CO 80303
flinton@advtech.uswest.com

ABSTRACT

The LEAP (Learn, Explore and Practice) intelligent tutoring system is a job task simulator with both domain and tutoring knowledge. Domain knowledge is organized by topics with declarative presentations and procedural exercises. Tutoring knowledge consists of numerous tutoring methods and a number of strategies for selecting the most appropriate domain knowledge and tutoring method for any given user model state.

One set of tutoring strategies selects topics to study and recommends one of several tutoring methods for studying them, based on the user's current knowledge and preferences, and on the topics' natural and prerequisite relationships.

Another set of tutoring strategies recommends which exercise to do next and how to do it. LEAP's flexibility here may be contrasted favorably with classroom instruction where all students are assigned to do the same exercises in the same sequence by the same method. LEAP selects exercises based on the user's current domain knowledge and desire for challenge, as well as on characteristics of the exercise. LEAP recommends one of four methods for studying an exercise, each emphasizing a different cognitive skill.

A third set of tutoring strategies determines within-exercise actions. Because exercises emphasize whole-task practice, within any given exercise a variety of actions may be required. LEAP keeps users practicing at the edge of their competence by selecting topic-related actions for practice, while skimming over familiar actions and scaffolding unfamiliar unrelated actions.

The Author mode has tools for easily modifying the topic selection strategy, the exercise selection strategy, the action selection strategy, and the user modeling facility. Modifications are likely to be needed whenever the characteristics of the intended users change or a different set of domain knowledge is put into the tutor.

INTRODUCTION

Most of us can recall having an instructor who was an expert in her field but who could not teach. In developing LEAP, we have made a conscientious effort to include both domain knowledge and teaching or tutoring skills. A side benefit of including tutoring skills in a tutor is generality. Once defined, the tutoring skills can teach other, similar, domains. To teach a new domain requires only inserting new domain knowledge into LEAP; the tutoring skills are already there

What kind of decisions are tutoring decisions and how does a tutor make them? The basic tutoring decisions are what to study and how to study. LEAP makes these decisions on three levels. LEAP recommends: 1) a topic of study and a method for studying it, 2) an exercise (a conversation) to practice and a method of practicing it, and 3) for each step within an exercise, LEAP determines whether the user or the system should perform it.

Training Task

The trainees are Customer Service Representatives, or reps, of a large telecommunications firm, a 'Baby Bell.' The telecommunications field is changing rapidly: challenging the company's basic service - voice conversations over copper wires - are the video cable companies' ability to provide better, cheaper broad band service via coaxial cable, and the cellular companies' ability to bypass wires altogether. New services such as voice messaging, caller ID, etc., and service enhancements are weekly occurrences.

Currently, the company has several thousand employees in customer service jobs or similar jobs, and hires or transfers several hundred employees to this area every year. The training need is to: 1) train the several hundred 'new' employees annually and 2) update the knowledge and skills of the several thousand existing employees as service changes are made.

The service rep's job is complex: 1) service representatives deal with a vast amount of frequently-updated information about the products and services available, 2) the company is highly regulated at both the federal and state level and the reps must be careful to comply with all regulations, and 3) the rep must simultaneously construct a conversation with a customer while inputting and retrieving information from various databases.

Domain representation

LEAP simulates the trainees' work environment. Simulated customers call with requests of various sorts and the rep must respond appropriately. Instead of simulating the database, LEAP monitors trainees' interaction with a training version of the actual database, permitting only appropriate actions. The simulated work environment is built using the domain representation, whose elements are described below.

The core item in LEAP's representation is the *Situation-Action* pair (SA pair). The situation side of the SA pair presents a situation to the service rep trainee, or user, and the action side of the SA pair contains the appropriate action a rep would take; for example:

Situation:
Customer: "I'd like to add voice messaging to my service."
Action:
Rep: "I can help you with that; may I have your name and phone number please?"

There are several types of Situation possible: 1) a customer statement or question, 2) a database field providing or requiring information, or 2) a mental recollection by the rep. Likewise, there are several types of Action possible: 1) talk with the customer, 2) enter information into the database, or 3) conclude something about the situation.[1]

A *conversation* is made up of a sequence of SA pairs. The key task that reps perform is the conversation, and all exercises in LEAP are conversations.

A *grammar* is a set of conversations in an AND/OR tree, where SA pairs are the nodes, and the branches are the different possibilities based on the customer's situation; e.g., whether she has a private line or a party line, whether she accepts the rep's suggestion or rejects it, etc. Every path through the grammar is a valid conversation. in LEAP's Table of Contents. Each type of conversation (e.g., Order Voice Messaging,

A *topic* is a portion or portions of the grammar that are conceptually related. A topic has a name that appears Remove Voice Messaging, etc.) is a major topic. Each part of a conversation (e.g., Check Voice Messaging Availability, Provide Set-Up Information, etc.) is a sub-topic. Each conversation passes through (addresses) several sub-topics.

[1] Compound situations and actions also exist, for example, when the situation is the Due Date database field, the rep's actions are 1) to tell the customer when the service can be provided, and 2) to type the due date into the field.

LEAP user model
LEAP's user model is pattered after Newell & Rosenbloom's (1981) ubiquitous law of practice. LEAP's user model is an improvement over the law of practice in that it considers not only the user's average score on each SA pair, but also the strength of the user's knowledge (or ignorance) as measured by the number of consecutively correct (or incorrect) responses for the SA pair. The user model output is a number (from 0 to 1) that predicts, roughly, the likelihood that the user will do the SA pair correctly the next time it appears.

Each SA pair in the domain knowledge base has this predictive value attached. The values of SA pairs are aggregated to provide ratings for each conversation and each topic. LEAP uses these numbers, along with other information, to make several tutoring decisions, including SA pair presentation, exercise selection, and topic recommendation.

HOW LEAP MAKES TUTORING DECISIONS

LEAP's tutoring decisions consist of determining what the user should study next and how the user should study it. These decisions are based on characteristics both of the study material and of the user. Study material is divided into topics, with presentations and exercises for each topic LEAP recommends specific topics, presentations and exercises. LEAP also contains a wide variety of learning methods for studying the material in the exercises. LEAP recommends the most appropriate learning method for the user at any given moment (Collins, Brown, & Newman, 1989; Reigeluth, 1983). The recommendation mechanisms are described in detail below. In general, LEAP makes tutoring decisions but does not force them on the user, instead they are put forth as recommendations. Responsibility and control for choosing what and how to study remain in the hands of the user.

How LEAP selects topics
LEAP recommends both a topic to study and a method for studying it. Topic selection is based on a consideration of three factors: the topic sequence in the Table of Contents[2], the topic last studied, and the user's proficiency on each topic. The selection algorithm is based on the goal of having the user first attain minimum proficiency in all topics (so as to become productive as soon as possible) and later acquire expertise in all topics (for improved effectiveness).

The topic selection algorithm is: begin at the first topic and practice it to a predetermined proficiency rating (currently set at 'good'). In the process, some other topics will have been encountered and studied to some extent. For the next topic, pick the one which is nearest to a 'good' proficiency rating and study it. Continue in this way until all topics have a proficiency rating of 'good.' After the user has learned all topics to 'good,' revisit each topic in the same manner, practicing until the user reaches the 'excellent' rating. If the user overrides LEAP's recommendation regarding the topic to study, stay with the user-selected topic until the user reaches the target rating, chooses another topic, or asks for a recommendation.

Topic ratings are: untried, needs practice, almost!, good, and excellent. These ratings are determined by the percentage of SA pairs in the topic the user has tried and her average score on those SA pairs. Rating names are created and their values assigned by the domain (content) author.

How LEAP selects a study method
After selecting a Topic, LEAP selects the method of studying the topic. There are three top-level study methods: Study Topic, Examine Conversation Structure, and Rehearse Conversation.

The algorithm for selecting a study method is straightforward and mirrors conventional classroom and textbook instruction: In brief, LEAP recommends that users first see a demonstration of the topic in use, then study knowledge related to the topic,, then practice applying the knowledge in simulated conversations. Users may

[2] As in textbooks, some topics are prerequisite to others; the instructional designer must place them earlier in the sequence.

also elect to drill and practice factual material, or examine the structure of a conversation or of the entire knowledge base. Each of the tree top-level study methods has two or more sub methods; these will now be described in more detail.

Study Topic

In Study Topic the user can learn by either of two methods: by Viewing conventional multimedia presentations or by using Drill and Practice mode. LEAP presents information using multiple media as appropriate: text, speech, graphics, photographs, animations, and videos. Currently animations are created in Macintosh Quicktime files, saved in PACO format, and then ported and run under UNIX using PacoPlayer. Video sequences are captured using a Parallax Video board, and saved in compressed JPEG format. The compressed JPEG files can then displayed under UNIX using Uniflix software.

The Drill and Practice mode (not implemented 04/31/93) is envisioned to be multiple choice questions based on the facts in the media presentations. Because the Drill and Practice mode will be similar to one phase of conversation practice *(described below) LEAP will be able to refer to the user model when selecting drill and practice items and to update the user model based on the user's performance during drill and practice sessions.

Examine Conversation Structure

The user can Examine Conversation Structure in either of two modes: Explore or Browse. In both modes, users examine the conversation structure (the AND/OR tree describing the conversation space) by traversing it, choosing among multiple situations, and moving forward or backward through the conversation space at will. In Browse mode, users see the expert response to each situation they select. Explore mode is a little more adventurous; in Explore mode, users attempt to respond to each situation themselves before seeing expert responses.

Rehearse Conversation

Conversations correspond to the exercise in conventional textbook instruction. Rehearse Conversation is the most complex part of LEAP's tutoring process. The complexity arises from the desire to have users work at the edge of their competence in the context of whole conversations. Working at the edge of one's competence means practicing mainly the task that is the current focus of attention while not redoing those tasks one already knows how to do, nor doing those tasks one is not yet ready to do, even though these tasks arise naturally during the course of a conversation.

How LEAP selects conversations

LEAP recommends both a conversation to study and a method for studying it. LEAP selects a conversation for study by ranking each conversation according to several factors and selecting the top-ranked one. Some of the factors LEAP considers are characteristics of the conversation itself: its relation to the current topic, its complexity, and its overall importance ranking; other factors are related to the user's current situation:: the number of times she has already practice the conversation, her current skill level. and her preference for hard or easy exercises.

How LEAP selects conversation study methods

After selecting a conversation, LEAP must select a study method for the conversational There are four ways to study an individual conversation. Sequenced by increasing user involvement, they are: See a Demonstration, Critique a conversation, Accompany a conversation, and Practice a conversation. Each of these is described below.

The study method selection algorithm is straightforward, in general, it progresses from less to more user involvement:

1. If the user has not yet seen any conversation addressing the topic then 'See a Demonstration.'
2. If the conversation has been labeled by the author as type critique then 'Critique a Conversation.'
3. If the user's performance rating on the keyboarding skills for the conversation is less than 'Good' then Accompany a Conversation.'
4. If none of the above, then 'Practice a Conversation,' which is the default.

See a Demonstration

When the user elects to See a Demonstration, LEAP demonstrates both sides of the entire conversation, presenting the customer's verbal actions and the rep's verbal actions, cognitive actions and database I/O actions. The user observes the demonstration and builds a conceptual model of the task.

Critique a Conversation

When the user elects Critique (not implemented 04/31/93), LEAP presents a conversation as in Demonstrate, however some of the 'expert' responses are in error. The user must note the errors, on a checklist, as they occur and indicate why the response is in error, thus articulating her knowledge.

Accompany a Conversation

When the user elects Accompany, LEAP presents both customer and expert sides of the conversation, omitting, however, the expert's keyboarding tasks. Thus the user must accompany the simulated expert conversation, by interacting with the database as appropriate, using keyboard and mouse, while listening to the conversation between the customer and the expert rep.

After each database command or entry the user makes, LEAP provides Situation-Action Feedback.[3] In general, if the user's action is correct, the conversation simply continues. If the action is incorrect, LEAP will give a hint as to the expected action. The user may then retry the action. If the user still fails to take the expected action, LEAP supplies it and the conversation continues.

Accompany is a form of scaffolding that reduces the cognitive load of multitasking (the reps call it overlapping). Instead of both constructing a conversation and interacting with the database, the user listens to a conversation between a customer and an expert rep, and simply keyboards along as appropriate. Fading is the absence of scaffolding. Thus when a rep's keyboarding skills are adequate for holistic practice, LEAP will fade by no longer recommending Accompany.

Practice a Conversation

Practice is the predominant learning event in LEAP. When the user elects Practice, LEAP presents a conversation step by step. Each step is an SA pair. During the conversation LEAP presents a situation, then observes and evaluates the action the user takes and provides feedback.

LEAP presents three types of Situation to the user: the first is a customer statement or question, the second is a database I/O requirement, the third is a customer contact standard or legal requirement. The user must respond to each situation with one of three types of action: a verbal statement or question, a database command or input, or an observation or conclusion. There is no correlation between situation type and action type.

Situation:
Customer: "I'd like to add Voice Messaging to my service."
and
Database ScreenO1: Name Number ( ) _
Action:
Rep replies: "I can help you with that, what is your name and number?"

Situation:
Customer: "My name is John Smith and my number is 448-3214."
Action:
Rep enters: John 448-3214

[3] Situation-Action Feedback is contrasted with Conversation Feedback, which occurs after the conversation has been completed. Conversation Feedback is described below. Usually both types of feedback are simply referred to as 'feedback,' since the type of feedback is clear from the context.

Situation:

Database screen: Services: ..., voice messaging, ...

Action:

Rep concludes: Voice Messaging is available in the customer's area so it's OK to provide the service he's requested.

When the user takes action in the database (as in Accompany), LEAP can directly observe, evaluate, and respond to the user's action. However, when the user is conversing with a simulated customer, LEAP cannot understand the user's action. Natural language understanding, especially the sort of speech understanding needed to interpret service representative trainee responses, is simply not on the horizon, nor has it proven realistic to expect users to type their responses. Thus, LEAP has the user speak her response (for the purpose of creating, articulating and practicing it) and indicate when she is done. LEAP then presents the user a list of plausible responses, from which the user selects the nearest equivalent to her own. The distracters in the response set are selected from a list based on misconceptions, missing conceptions, near misses, blunders, and other empirically determined wrong answers trainees often make. This design compromise allows the tutor to observe, evaluate, and respond to the user's 'spoken' responses without using speech recognition or natural language processing.

After each action the user takes, LEAP provides SA pair feedback. If the user's action is correct, the conversation simply continues. If the action is incorrect, LEAP provides a hint of some sort and allows the user a retry before supplying the correct action and continuing. Detailed feedback for incorrect actions depends on the type of action the user is expected to take. Feedback for database actions was described above in Accompany. As mentioned, if the user's action is verbal, the user makes a verbal statement or asks a question aloud. The verbalization is recorded for possible use later during Conversation Feedback. LEAP then presents the list of plausible responses. If the user selects an incorrect action, LEAP informs the user that her answer was not the best, gives her a hint, and allows the user to retry. If the user still fails to select the expected action, LEAP supplies it and continues. The feedback for an observation or conclusion is like that for verbal actions.

Other study methods employed during practice

While the user is practicing a conversation, the tutor may decide to overrule the practice format described above and skim material already known to the user, or to scaffold material that is unknown to the user and irrelevant to the current focus of attention.

The algorithm for selecting these methods is:

1. If the user already knows the action for the situation presented in the SA pair, then Skim.
2. If the user doesn't know the action for the situation and the situation is not part of the current topic, then Scaffold.
3. Otherwise Practice, which is the default.

The need for skimming and scaffolding arises from two design decisions: the first decision is to have the user practice in the context of whole tasks. Users should practice specific skills in the context of whole tasks in order to avoid the problem of learning 'disembodied knowledge' (Suchman, 1987; Lave & Wenger, 1991). In this case the whole task is a conversation, and in realistic conversations almost anything could occur, including situations with which the user is already well-practiced, situations the user is practicing, and situations the user is unprepared to deal with (i.e., situations the user has studied, is studying, and has not yet studied, respectively). The second design decision is to have the user work at the edge of her competence, neither wasting time re-doing tasks already-known, nor floundering with (and probably foundering on) tasks which she is not yet capable of dealing with. Skimming and scaffolding are ways to avoid these two situations while practicing in a whole-task context.

Skim

In general, if the user knows what action to take in a given SA pair, then LEAP will skim it. Skimming means that LEAP presents both the situation and its appropriate action, and the user need only click to make the conversation continue. On any given SA pair, however, LEAP may or may not skim. The better the user knows the SA pair, the more likely LEAP is to skim it. When the task is new to the user, LEAP will always have her perform it, and when the task is known to the user, LEAP will hardly ever ask her to perform it. LEAP's decision

to skim an SA pair is made by chance; and the odds of skimming are determined by the user's knowledge of the particular SA pair.

There are three pedagogical justifications for this approach to skimming. First, there is no theoretical way of determining exactly how many times a user must try something in order to learn it (Anderson, Boyle, Corbett, & Lewis 1990). Thus there is no way to determine when to stop questioning. Second, there is no theoretical way of determining how the passage of time and the performance of other tasks (related and unrelated) influence the retention and use of the knowledge. Thus it is necessary to practice or rehearse the material over time in order to ensure the retention and correct usage of the knowledge. Third, a certain element of unpredictability or surprise contributes to maintaining the user's attention (Lesgold, et al., 1992).

Scaffold

When the user has no experience with the situation and the SA pair is not in the current topic of instruction, LEAP will scaffold. Scaffolding, like skimming and demonstrating, means that LEAP presents both the situation and the action; and the user need only click to make the conversation continue. The value of scaffolding is that the user can practice in a realistic context, that is, she can practice the tasks she is learning within a conversation she would otherwise be unable to complete on her own.

While LEAP is scaffolding, the user has the opportunity to observe a skilled performance and to begin forming a mental model of the task.

Fade

Besides skimming and scaffolding there is a third mode that LEAP must occasionally employ during conversation practice. LEAP occasionally decides to fade a certain type of situation (not implemented 04/31/93). The purpose of fading is to increase the fidelity of the simulation. Recall that conversation knowledge is represented in an and/or tree of SA pairs. Occasionally the Situation side of an SA pair has no explicit characteristics or properties.

For example, when taking an order for Voice Messaging the rep must remember to tell the customer that Voice Messaging is a competitive service available from other sources. In such cases, there is no explicit Situation to stimulate the Action; the rep must remember to make the disclosure on her own. The first few times the user visits this SA pair, LEAP scaffolds by presenting an explicit message describing the situation: "Full Disclosure requires 'competitive service' disclosure." Once the user knows how to perform the action with an explicit situation, LEAP fades the situation, removing the explicit situation material and requiring the user to remember to perform the task, just she would have to in actual conversations.

The representation chosen for LEAP makes it difficult to fade situations completely. The SA pair representation has a built-in constraint: the only time a user can input information to LEAP is when there is a situation to respond to. Situations can, however, be partially faded; a partially faded situation message is: "You need to recall what to do here."

Conversation feedback

When the user reaches the end of a conversation in Explore, Critique, Accompany, or Practice modes, LEAP provides conversation feedback[4] (not implemented 04/31/93). In conversation feedback the user may perform several activities: review customers. measures of overall performance, review the conversation from her own and from the customer's perspective, ask and answer questions, and retry key parts of the conversation.

---

4 Many instructional design theories espouse the value of feedback; three are found in Reigeluth (1983): Aronson and Briggs, page 92, state: "Providing feedback is a crucial instructional event" Merrill, page 322, states: "Feedback should always accompany practice at every performance level" and Keller, page 426, states: "To maintain intrinsic satisfaction with instruction, use verbal praise and informative feedback rather than threats, surveillance, or external performance evaluation."

There are two measures of overall conversation performance: 1) an overall score based on the number of correct actions the user took, and 2) a rough measure of productivity based on the amount of time the user took to complete selected portions of the exercise. There are two perspectives from which to review the conversation: the rep's and the customer's. Working from the rep's perspective the user can review the conversation segments containing her errors, comparing her answers to the expert's. Errors based on known misconceptions have explanations presented at this time. The user can also reflect on the experience by attempting to perceive the conversation from the customer's perspective. One way to do this is by rating the customer's satisfaction for the call using relevant portions of the customer satisfaction survey form, the survey form used to acquire satisfaction data from real

As in the other learning modes, if the user has a question during feedback, she can look up information in the rep's Desk Reference and in the multimedia Study Topic side of LEAP. If the user believes LEAP to be in error regarding any point, or has any other suggestions to make, she can leave a note to the developers. Finally, the user may elect to retry the conversation, skimming everything except the SA pairs she erred on.

Test

When the user reaches a certain level of performance, LEAP recommends Test. Test mode, (not implemented 04/31/93) as the name implies, is for measuring the user's skill under test conditions. In Test mode the user must perform the whole conversation under conditions as realistic as possible. All Skimming and Scaffolding are dropped. Implicit situations are Faded. Access to multimedia presentations of lesson material is blocked. User performance data in test mode is recorded separately, with a time & date stamp. During LEAP's development cycle, users will be required to take pre- and post tests in order to measure LEAP's efficacy.

Whether and how Test mode is used during training depends on the training philosophy of the organization where LEAP is fielded. Having separate specific tests is something users, instructors, and administrators are used to. A single test, however, is not as good a measure of the user's skill as the User Model LEAP has built up over time, if for no other reason than the time limit of the test situation, which limits testing to a sample of the content material and a single-shot performance. The User Model is a superior measure of the user's performance because the user has do to well consistently to have a 'good' or 'excellent' proficiency rating.

Revising LEAP's decision-making strategies

LEAP's decision-making strategies have been described in detail above. These strategies are not hard-coded into LEAP, but are easily modified. Tutoring decisions are made by taking into consideration a number of factors. LEAP's authoring interface has been designed so that the weight given each factor is easily varied. The contribution of each factor to the decision can vary from zero to one hundred percent. That is, the factor can be removed from LEAP's decision-making altogether, or it can be the sole consideration in the decision, or be anywhere in between. The influence of each of these factors on the topic selection decision can be varied by simply moving a slider.

In this section we will describe how to vary LEAP's decision-making process for selecting topics, selecting conversations, and for skimming an SA pair.

<u>Selecting topics</u>

Topic selection is based on a consideration of three factors: the topic sequence in the Table of Contents, the topic last studied, and the user's proficiency on each topic. Moving the sliders adjusts the weight of the factor's contribution to the decision from 0% to 100%. Within the proficiency factor, the current design allows the author to select one of three predetermined sequences of priorities. A more flexible design for assigning and sequencing topic ratings is planned.

When creating the course materials, the author must take care to sequence the topics according to accepted instructional design principles, as one of the topic selection factors considers this author-defined sequence. As the user progresses through the course, LEAP will note her proficiency and preferences and re-sequence the topics accordingly. The current defaults are set so that user proficiency is weighted more heavily than topic sequence, with the effect that a topic, once begun, is mastered before going on to another. If a new, different, domain or

content area, one with more prerequisite relations among the topics, were to be put into LEAP, the topic sequence factor would have to be given more weight.

Selecting Conversations

As mentioned, LEAP selects a conversation for study by ranking each conversation according to several factors and selecting the best one. Some factors LEAP considers are characteristics of the conversation itself: its relation to the current topic, its complexity, and its overall importance ranking; other factors are related to the user's current situation: the number of times she has already practiced the conversation, her current skill level, and her preference for hard or easy exercises. As with the topic selection factors, the relative weight given to each of these conversation selection factors is adjustable. Weights are adjusted by manipulating sliders accessible to the author. Currently, weights are set by the developer's experienced intuition. Empirically determined weight settings are preferred; the requisite studies are planned.

Skimming an SA pair

The decision to skim or not to skim is made by chance, with the odds of skimming determined by two factors: 1) whether the SA pair is in the current topic; ceteris paribus, SA pairs in the current topic are more likely to be asked, and 2) how well the user 'knows' the SA pair; the better she knows it, the less likely she is to be asked. The author revises the skimming decision odds by, as usual, manipulating slider bars.

SUMMARY

At the push of the Recommend button LEAP will recommend a topic, a conversation, and a set of SA pairs for the user to study. LEAP has a selection method for each level of domain knowledge, Topics, Conversations, and SA pairs. LEAP's selection method for each level of knowledge is based on a number of factors, some characteristic of the knowledge, others characteristic of the user's state. Each of the factors considered by LEAP's knowledge selection mechanism is accessible to the Author. Each factor has a slider the Author can use to increase or decrease the factor's weight in the knowledge selection process. Thus, the method by which LEAP selects content for study can be radically, yet easily, altered

The same push of the Recommend button directs LEAP to recommend a method to study the materials it selects. LEAP has a set of study methods for each level and a method selection mechanism for each level. In general, the methods range from the cognitively simple to the cognitively complex and LEAP's recommendation mechanism recommends them in sequence from simple to complex.

For each Topic, knowledge is represented:
- Declaratively in presentations
- Procedurally in conversation exercises
- Abstractly in conversation space

There are two methods for studying declarative presentations: view multimedia presentations and practice recalling related facts. There are five increasingly complex methods for studying procedural conversations: See a demonstration, Critique a conversation, Accompany a conversation, Practice a conversation, and Self-Test. Within a conversation LEAP may present SA pairs in four ways progressing from one to the next as the user's skill develops: Scaffold, Practice, Fade, and Skim. Users are fluent and their training is complete when they can skim all SA pairs. Finally there are two methods the user may traverse the abstract conversation space: Browse and Explore.

REFERENCES

Anderson, J. R. (Ed.). (1981). Cognitive skills and their acquisition. Hillsdale NJ: Erlbaum.

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. In W. J. Clancey & E. Soloway (Eds.), Artificial intelligence and learning environments (pp. 7-50). Cambridge MA: MIT Press.

Aronson, D. T., & Briggs, L. J. (1983). Contributions of Gagne and Briggs to a prescriptive model of instruction. In C. M. Reigeluth (Ed.), Instructional design theories and models: An overview of their current status. (pp. 75-100). Hillsdale NJ: Erlbaum.

Clancey, W. J. & Soloway, E. (Eds.). (1990). Artificial intelligence and learning environments. Cambridge MA: MIT Press.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser (pp. 393-452). Hillsdale NJ: Erlbaum.

Keller, J. M. (1983). Motivational design of instruction. In C. M. Reigeluth (Ed.), Instructional design theories and models: An overview of their current status. (pp. 383-436). Hillsdale NJ: Erlbaum.

Lave, J., and Wenger, E. (1991). Situated learning: Legitimate peripheral participation. Cambridge: Cambridge University Press.

Lesgold, A., Eggan, G., Katz, S. and Rao, G. (1992). Possibilities for Assessment Using Computer-Based Apprenticeship Environments. In J. W. Regian and V. J. Shute (Eds.), Cognitive approaches to automated instruction (pp. 49-80). Hillsdale NJ: Erlbaum.

Merrill, D. M. (1983). Component Display Theory. In C. M. Reigeluth (Ed.), Instructional design theories and models: An overview of their current status. (pp. 279-334). Hillsdale NJ: Erlbaum.

Newell A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), Cognitive skills and their acquisition (pp. 1-55). Hillsdale NJ: Erlbaum.

Regian J. W., & Shute V. J. (Eds.). (1992). Cognitive approaches to automated instruction. Hillsdale NJ: Erlbaum.

Reigeluth, C. M. (Ed.). (1983). Instructional design theories and models: An overview of their current status. Hillsdale NJ: Erlbaum.

Resnick L. B. (Ed.). (1989). Knowing, learning, and instruction: Essays in honor of Robert Glaser. Hillsdale NJ: Erlbaum.

Suchman, L. A. (1987). Plans and situated actions: The problem of human-machine communication. Cambridge: Cambridge University Press.

An Intelligent Foreign Language Tutor Incorporating Natural Language Processing

Michelle R. Sams, Ph.D.

Army Research Institute
5001 Eisenhower Ave. (ATTN: PERI-II)
Alexandria, VA USA 22333
(703) 274-5540; sams@alexandria-emh2.army.mil

ABSTRACT

The Army Research Institute has a program of research on technology applications for language learning. Phase I of the project has produced an interactive, intelligent PC-based tutor to assist military linguists in maintaining and improving their German language proficiency. The most powerful aspect of the tutor lies in the use of a natural language processor (NLP). The NLP can analyze students' freely typed input, provide descriptive feedback regarding grammatical errors, build a student model, and use that information to drive lesson progression and remediation. Phase II efforts are currently underway in Arabic and Spanish. Advances will include limited speech recognition, a dialog module, and an animated graphics microworld responsive to student's linguistic input.

INTRODUCTION

Foreign language skills which are needed for global conflict decay rapidly without adequate practice. The optimal language sustainment and enhancement environments are those in which the student is an active participant using the language, practices job relevant skills, and receives individual tutoring guidance. The Army Research Institute recognized that various emerging technologies could be innovatively combined to create such an environment in a cost-effective and portable computer tutor[8].

Fluency in a language incorporates comprehension (listening and reading) and production (speaking and writing). Many forms of passive materials support language comprehension, such as audio-cassettes and textbooks. More active environments exist in the form of computer-based instruction, especially in the use of interactive video disc (IVD) and hypermedia applications[4,6]. However, many of these programs don't adapt to the individual student and are based on the student's recognition of correct responses (e.g., selecting multiple choice items). Some computer programs do allow students to produce their responses via written input, however these inputs are unnaturally constrained due to a reliance on either a "string or keyword" matching approach to determine correct input.

The ultimate goal is for the computer and the human to exchange information much like two humans. Computational linguists, computer scientists, linguistic and cognitive psychologists have all contributed to this goal through the field of natural language processing (NLP). NLP has supported foreign language applications in machine translation programs (i.e., scanning and translating documents) and message understanding (i.e., extracting pertinent information from text messages)[7]. However for these applications, the NLP operates on high-powered machines (e.g., SUN workstations) and are designed with the assumption that the material is in fairly correct grammatical form, which often requires pre- and post-editing.

The potential power of NLP in a language tutor is that it can "understand" and analyze student's natural use of the foreign language. Although some development efforts are underway in applying NLP to foreign language learning [3,9,10], none encompass the breadth and depth of the requirements in the current project. The challenge of ARI's tutor was to devise a way for the NLP to analyze student input which is not in correct form, identify multiple grammatical errors within the same sentence, provide descriptive feedback to the student, build a model of the student's strengths and weaknesses, and then use that information to drive lesson progression and

remediation. And for the tutor to be effective in usability and cost to the Army community, it needed to provide an easy authoring system for instructors, maximize use of commercially available software and run at a reasonable speed on an 80386/486 platform.

NATURAL LANGUAGE PROCESSOR

There are many grammatical theories and hence, different approaches to representing a grammar computationally [1]. The challenge of the language tutor was to identify a NLP methodology which would be accurate, tolerant of student errors, compact (for PC application) and extendible across languages (for developing a family of tutors for various languages). The framework chosen was Chomsky's [5] government-binding (GB) theory which posits that universal abstract principles form the basic structure of all languages. The GB approach generates the underlying structures of a language through a small set of interacting modules and constraints, instead of developing an extensive list of specific rules which characterize all the syntactic possibilities of the language. Therefore, the GB modular approach, with parameters that are initialized for different languages, promotes both compactness within a language and extendibility across languages.

The GB NLP works by parsing a sentence in a bottom-up process by identifying the individual words and then determining the relationships among them. The main components of the parser are a preprocessor, a morphological analyzer, a lexicon, a syntactic parser, an error handling facility and a semantic interpreter. First, word strings are submitted to an interactive preprocessor which identifies spelling mistakes to the student. The morphological analyzer then decomposes the words into subparts (i.e., roots and affixes) based on information in the lexicon (similar to a dictionary). Specific information is drawn from the lexicon about the word subparts (e.g., whether it is a verb or noun, singular or plural). The word subparts are unified back into their original state and passed along with the descriptive information from the lexicon to the syntactic parser.

Based on this information, the parser tries to build a structure, called a parse tree, which reflects the appropriate relationship among the words. The parser moves words around trying to find all possible constructions which satisfy a minimal set of basic phrase structure rules (i.e., language universal abstract principles). Also operating are two interacting modules. The required constraints component establishes those barest essentials needed to comprise somewhat intellible parts of a sentence. The second component, the preferred constraints, contains the categories of grammatical errors to flag once the sentence or phrase has been successfully parsed. In other words, the parser at first overgenerates the number of possible structures and then imposes constraints ("weeds them out") based on language specific rules. This approach provides for robustness (i.e., parser does not fail as it encounters each error) and allows specification of the grammatical error types of importance to the instructional objectives.

Let us illustrate (see Figure 1) with a simplified example in English, "The girl write the story." All the words and subparts are recognized and passed on to the parser. "Write" has been identified as a verb which often has an inanimate object. The parser looks for an inanimate object in the appropriate position (following the verb) to complete the verb phrase branch of the tree, "write story". It continues on in this manner until it completes the parse tree and the error handler conveys to the tutor that a subject-verb agreement error ("girl-write") was made. The tutor reads the error data and uses that information to drive specific feedback to the student and to build the student model.

The parser can even handle some reasonable sentence fragments, so that if the tutor asked where the castle is located, the student could appropriately reply, "to the north of Berlin". However, every possible sentence construction and grammatical error could not be captured by the NLP. Therefore, determination of the range of input and error types was made through analysis of language usage by the military linguists in job-specific environments. The most common, problematic, and critical areas identified were those which were included in the NLP development. The NLP in the German tutor [2] can parse many but not every type of sentence construction, can identify five classes of syntactic errors and has a lexicon of 5000 words (common and military).

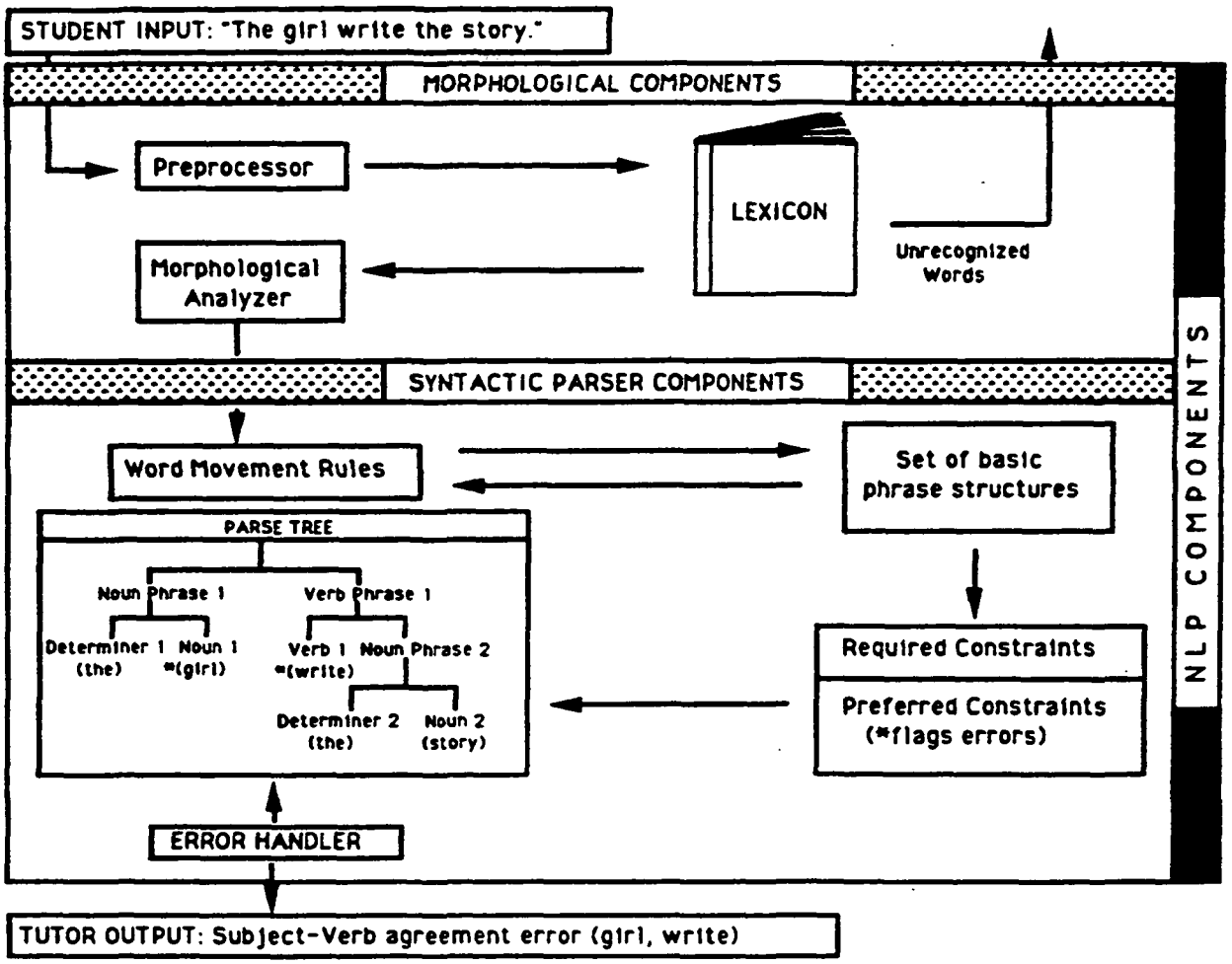


Figure 1. A simplified functional diagram of the natural language processor (NLP).

FEATURES OF THE FIRST GENERATION TUTOR

The tutor is a modular, open architecture design which supports the development of different lessons and instructional approaches by authors (i.e., instructors and researchers). A variety of exercise types are available as building blocks for lessons. These include multiple-choice, point-to-a-location on a graphic (using the cursor), fill-in-the-blanks, type in responses (phrases or full sentences) to tutor queries, and sorting text items under appropriate columns. All of these exercises, except the sorting, can be either text or oral queries from the tutor. The fill-in-the-blank and typed responses are automatically sent to the NLP component which analyzes the student's input. Exercises can be presented in a fixed sequence or drawn randomly from a pool of exercises within a set.

Students receive feedback as to the correctness of their responses for any of the exercise types. The most specific and diagnostic feedback is provided as a result of those exercises sent to the natural language processor. The NLP analyzes the student's input and identifies the specific grammatical errors the student made. The tutor also can classify the grammatical mistakes into primary and secondary errors. Students are automatically presented with details of their primary errors, but have the option whether to view information about their secondary errors. This classification schema was developed to promote language use and not constantly interrupt the intermediate

level student. Therefore, the instructor can determine which types of errors to bring to the immediate attention of the student and which errors are less important for overall lesson objectives.

Progression from one exercise set to another can be in a fixed sequence or based on the student's performance. For example, if the student demonstrates proficiency (e.g., answers 80% of the exercise set correctly), then the student can be moved on from that set and presented with the next, more challenging exercise set. More sophisticated lesson branching can be accomplished using the student model, which is built on output from the NLP analyses. The student model is developed from error counts within specified grammatical categories which can accumulate across exercise sets. If it is found that a student is making an unsatisfactory number of errors in a particular grammatical category (e.g., noun-verb agreement), then the student can be automatically branched to an exercise set providing additional assistance and practice in that problem area. After remediation, the student can return to the regular lesson. At any time, students can opt to view a graphic depicting their grammatical problem areas and general progress throughout the lesson.

A variety of on-line help is available for the student through pull-down menu items. Explanations are provided about the tutor's capabilities and exercise procedures. Students may request assistance in the form of oral or written hints as they attempt to respond to each exercise. For vocabulary help, there is a German-English dictionary which can be accessed either alphabetically or through search terms.

One of the goals of the tutor was to design a lesson authoring interface which requires no programming skills and little training. This capability allows instructors and researchers to create new lessons and test alternative instructional strategies. The author only needs to access a template of the desired exercise type and fill in the blanks. Changing a "point to-a-location on a graphic" exercise is accomplished by simply "clicking and dragging" an existing "hot button" (target square) to the desired location. Creation of exercise sets, setting parameters for lesson progression, defining feedback and hints are all modified through easy-to-use interfaces.

The tutor runs on an 80386/486 class machine requiring 8Mb of RAM or better. It needs a color VGA, Microsoft-compatible mouse, Windows 3.1 or better running in 386 enhanced mode. It uses a Pro Audio Spectrum speech board. The user interface is in Toolbook, the NLP is written in Arity PROLOG, and runtime versions are provided of each.

ADVANCES FOR THE SECOND GENERATION TUTOR

The German tutor is the first product of a multi-phase research and development effort. The feasibility of the GB NLP approach and its integration into an intelligent language tutor was demonstrated in the first phase, particularly with regards to real-time diagnosis of language form (grammar). However, it is the goal of ARI's language tutor project to develop a more communicative approach to language training. Consequences of correct language usage in the real world usually result in one of two responses, either a conversational reply or an appropriate action (e.g., execution of a command). Therefore, to support a more realistic language training environment, some major advances will need to be accomplished in the Phase II effort (supporting Arabic and Spanish).

It is ARI's plan to include a dialog module in the second generation tutor. The scenario will include a modelled prisoner of war (POW) with an domain specific knowledge base, which the military linguist can freely query to practice interrogation skills in the target language. Providing the capability to converse with the computer poses challenges in understanding student input and generating appropriate responses. It requires a component for semantic interpretation (to determine meaning). Our approach is to transform the language specific input into a language independent representation, called a lexical conceptual structure (LCS). The LCS is the cornerstone for interaction with the knowledge base and discourse planner to support dialog (see Figure 2).

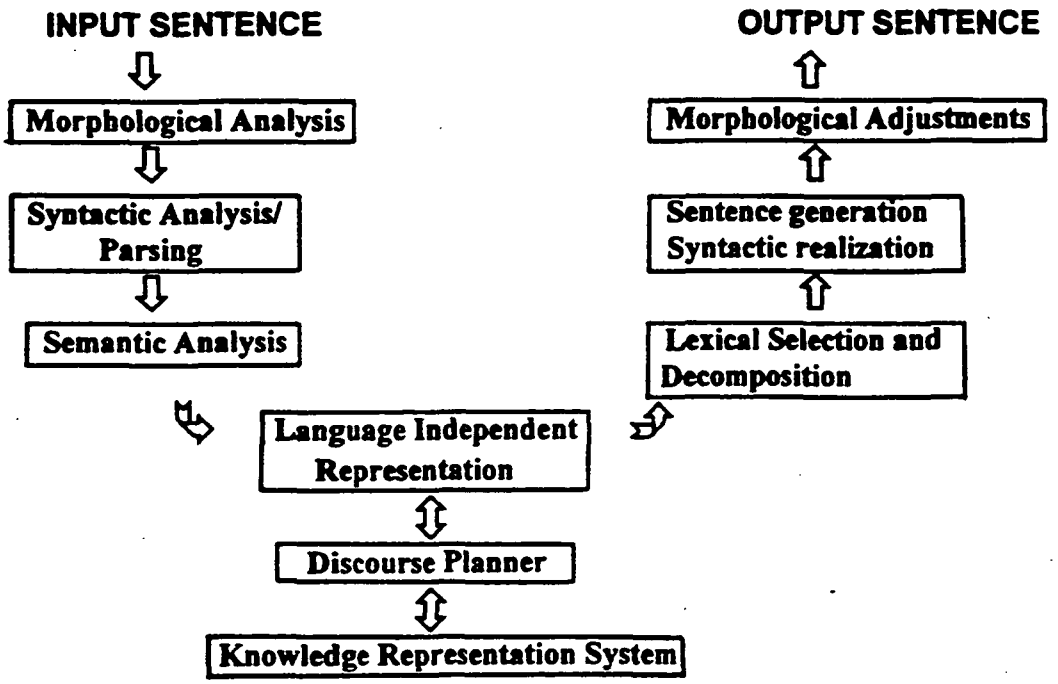


Figure 2. System design to support dialog.

A separate module is planned which will provide an animated graphics microworld in which students can explore the consequences of their language input. It will contain contain agents and objects with certain attributes. Our strawman idea is to provide a scenario containing allies and enemy agents with weapons and documents. The goal of the interaction is to prepare for an interrogation. The military linguist can enact the surrender (e.g., command that the enemy surrender their weapons) and search for and examine valuable documents. The on-screen action will occur as a result of the linguist typing in the appropriate commands. These will be analyzed by the NLP and transformed to the LCS which will be the "hook" to the code that generates the appropriate animated graphic.

Other features of the second generation effort will include speech recognition exercises (words and short phrases). These exercises will include selecting multiple choice alternatives and pronunciation drills with real-time feedback. Ease of authoring new lessons will again be a requirement and this will extend to the new developments. Particular challenges will lie in easy authoring of the NLP lexicon, animated graphics, and speech components.

In summary, the general approach is to provide an open architecture system to provide flexibility and extendability. The flexibility applies to instructors who wish to develop new lesson material within a variety of instructional approaches (e.g., feedback, branching) and to researchers who wish to investigate language acquisition and training issues. The extendability applies to the NLP framework so that additional development efforts are minimized for additional languages (even widely disparite languages, such as Arabic and Spanish). This is accomplished with the parsimonious representation of language into interactive modules and transformation into language independent structures.

ACKNOWLEDGEMENTS

ARI's prime contractor for the Phase I German Tutor was the Science Applications International Corporation and Phase II efforts are being developed by Micro Analysis & Design of Boulder, CO. The natural language processing components for both phases are developed by the University of Maryland.

The views, opinions, and findings contained in this paper are those of the authors and should not be construed as an Official Department of the Army position, policy, or decision.

REFERENCES

[1] Allen, J. (1987). *Natural language processing*. Benjamin/Cummings: Menlo Park, CA.

[2] Azadegan, S., Martin, J., Merlo, P., & Weinberg, A. (submitted). A government-binding parser for foreign language training. *Journal of Computational Linguistics*.

[3] Bailin, A. (Ed.) (1991). ICALI research [Special Issue]. *CALICO Journal, 9*(1).

[4] Bickes, G. & Scott, A. (1989). On the computer as a medium for language teaching. *CALICO Journal, 6*(3), 21-31.

[5] Chomsky, N. (1981). *Lectures on government and binding*. Foris, Dordrecht.

[6] Rubin, J., Ediger, A., Coffin, E., Van Handle, D. & Whiskeyman, A. (1991). Survey of interactive language video discs. *Calico Journal, 8*(4), 65-94.

[7] Slocum, J. (1985). A survey of machine translation: Its history, current status, and future prospects. *Computational Linguistics, 11*(1), 1-17.

[8] Swartz, M.L. & Psotka, J. (1989). *AI tools for foreign language training* (ARI Research Report No. 1533). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences

[9] Swartz, M.L & Yazdani, M. (Eds.). (1992). *Intelligent tutoring systems for foreign language learning*. New York: Springer-Verlag.

[10] Underwood, J.H. (1987). Artificial intelligence and CALL. In W.F. Smith (Ed.), *Modern media in foreign language education: Theory and implementation*. Lincolnwood, IL: National Textbook Company.

Knowledge Acquisition and Interface Design for Learning On Demand Systems

Wayne A. Nelson
Department of Educational Leadership
Southern Illinois University at Edwardsville

The rapid changes in our world precipitated by technology have created new problems and new challenges for education and training. A knowledge "explosion" is occurring as our society moves toward a service-oriented economy that relies on information as the major resource. Complex computer systems are beginning to dominate the workplace, causing alarming growth and change in many fields. The rapidly changing nature of the workplace, especially in fields related to information technology, requires that our knowledge be updated constantly. This characteristic of modern society poses seemingly unsolvable instructional problems involving coverage and obsolescence. The sheer amount of information to be learned is rapidly increasing, while at the same time some information becomes obsolete in light of new information. Education, therefore, must become a lifelong process that features learning of new material and skills as needed in relation to the job to be done.

Because of the problems cited above, the current model of learning in advance may no longer be feasible in our high-technology world. In many cases, learning in advance is impossible because there are simply too many things to learn. In addition, learning in advance can be time consuming, and often results in decontextualized knowledge that does not readily transfer to the work environment. The large and growing discrepancy between the amount of potentially relevant knowledge available and the amount a person can know and remember makes learning on demand an important alternative to current instructional practices. Learning on demand takes place whenever an individual must learn something new in order to perform a task or make a decision. Learning on demand is a promising approach for addressing the problems of coverage and obsolescence because learning is contextualized and integrated into the task environment rather than being relegated to a separate phase that precedes work. Learning on demand allows learners to see for themselves the usefulness of new knowledge for actual problem situations, thereby increasing the motivation for learning new information. Finally, learning on demand makes new information relevant to the task at hand, leading to more informed decision making, better quality products, and improved performance.

Successful models of learning on demand have existed where a learner could afford the luxury of a personal coach and critic who lends support to the learner's individual problem-solving activities. For a variety of reasons, this relationship is difficult to achieve in modern society between humans, but advances in computer technology now make it possible for the computer to assume the role of coach and critic in many situations where the machine is already used as a tool to complete work. With a computer-based learning on demand system, learning does not take place in a separate phase and in a separate place, but instead is integrated with working and contextualized by real problem situations. In fact, learning on demand carries the potential to truly set computer-based learning environments apart from other instructional media, and therefore may be a unique research direction where technology can make a significant difference in achieving new educational objectives and overcoming some of the basic limitations of our current instructional approaches.

CONCEPTUAL FRAMEWORK

Learning-on-demand systems are based on recent findings in cognitive science that suggest the learning process involves knowledge construction, not knowledge absorption (Papert, 1980). Successful learners elaborate and develop self-explanations that extend the information in texts or other instructional materials. Learners use current knowledge to construct new knowledge and to restructure existing knowledge (Kintsch, 1988). Learning is highly related to the situation in which it takes place (Greeno, 1989; Lave, 1988; Suchman, 1987). No amount of knowledge of general principles accounts for or guarantees the success of action in real-world problem situations. In other words, general information needs to be elaborated and experienced in various contexts in order to devise specific courses of action that can be applied in specific problem situations. There is also overwhelming evidence that self-directed (Brown & Palinscar, 1989) and intentional learning (Bereiter & Scardmalia, 1989) are some of the most effective components of successful learning experiences.

Learning on demand systems also utilize the reflection-in-action model that describes how many practitioners proceed in their work (Schoen, 1983). Action is governed by a nonreflective thought process that proceeds until a breakdown occurs, when the practitioner realizes that nonreflective action has resulted in unanticipated consequences (positive or negative). In order for the practitioner to notice the breakdown, the situation has to "talk back" to the practitioner, so that reflection can be used to repair the breakdown and nonreflective, situated action can resume. Reflection-in-action takes place within the time period during which the decision to act has been made, but the final decision about how to act has not. Mechanisms in support of learning on demand must take advantage of these breakdown situations in order to provide information when it is most needed.

TYPES OF LEARNING ON DEMAND SYSTEMS

Learning on demand has been described by Fischer and his colleagues (Fischer, 1991; Fischer, Lemke, Mastaglio, & Morch, 1990; Fischer, Lemke, & Schwab, 1985), who have developed two types of systems: active help systems and critiquing systems. Active help systems, also known as context-sensitive help systems, are typically embedded in a software application, interrupting the user to "volunteer" information based on an analysis of the task the user is attempting to complete. Critiquing systems can be integrated within a variety of work environments to present a reasoned opinion about a user's product or action using knowledge of domain principles to detect and critique suboptimal solutions or actions. Other researchers working in the area of electronic performance support systems (EPSS) are studying learning on demand as a form of decision support (e. g., Gery, 1991). Such systems may be designed as expert systems that identify problematic decisions and suggest alternatives, or as on-line documentation for learning about aspects of the job that are infrequently performed. Finally, "intelligent applications" are beginning to emerge that facilitate production tasks by automating certain aspects of the tasks, or by suggesting ways that the user may use the software to accomplish various goals.

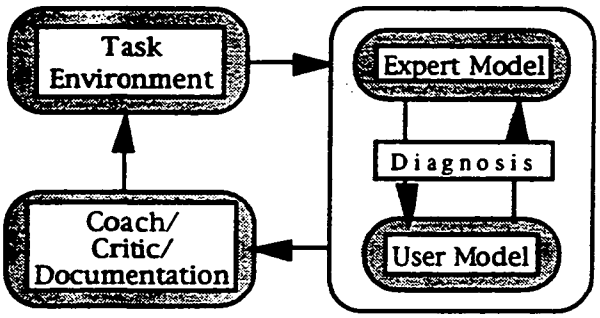
Regardless of the type of learning on demand system under development, issues of system architecture and knowledge representation still remain at the forefront. Essentially, learning on demand environments are similar to expert systems and intelligent tutoring systems. In fact, a general architecture for a learning on demand system, as depicted in Figure 1, is similar to those used for intelligent tutoring systems. A task environment (interface) where the user completes work in the domain is designed to communicate with an expert model that represents the various types of activities that can be accomplished with the system. The actions taken by the user are represented in a user model, and diagnosis of the user's actions can occur through comparison with the expert model. When errors are made (or help is requested), a coach or critic is invoked to communicate with the user within the task environment.

The types of knowledge required for learning on demand systems, and the subsequent knowledge representations within the computer, will also be similar to intelligent tutoring systems and expert systems. Considerable debate among cognitive scientists about the nature of declarative and procedural knowledge has resulted in the recognition that the appropriate knowledge representation scheme depends on the problem being addressed (Gardner, 1987). Certainly all representations can be reduced to bits in computer memory, but just as certain high-level computer programming languages are more effective for particular types of problems, the features of various knowledge representation schemes can make one type of representation more appropriate than others.

In general, high-level representations can be categorized according to representational characteristics and system functions. Semantic networks, augmented transition networks, bites, frames, scripts, and grammars have been used to construct systems which hierarchically organize information for tutoring, simulation, and parsing of natural language. These schemes use highly structured databases that allow reasoning based on links between concepts, inheritance of values from superordinate concepts, and procedures for dealing with specific objects. Modeling physical processes, problem-solving processes, and decision behavior has been achieved with behavior trees, dependency graphs, planning nets, probability matrices, and procedural nets. These networks have nodes representing various modules of a process, and links representing relations and interactions between real-world structures and processes. Rule-based production systems utilizing GAO graphs, overlays, and goal-directed productions have been employed to implement skill acquisition, tutoring and diagnosis. These techniques make comparisons between models of expert behavior and the actual student behavior in order to prescribe appropriate

intelligent tutoring. These higher-level representations have been developed for many different applications (see Wenger, 1987).

Figure 1. A general system architecture for learning on demand.



KNOWLEDGE ACQUISITION FOR LEARNING ON DEMAND SYSTEMS

To develop expert systems or intelligent tutoring systems, the knowledge structures and decision-making processes experts utilize must be elicited. Decisions must be made about which knowledge representation scheme is most appropriate, and how the expert's reasoning might best be implemented by the computer. Knowledge engineers typically accomplish this by working in several stages where they analyze the problem domain, determine the key concepts for representation in the system, design an informal structure for the knowledge to guide subsequent knowledge acquisition activities, structure the knowledge in a formal computer-based representation, and then validate the knowledge and system performance (Nelson, 1989). Since learning on demand systems closely resemble intelligent tutoring systems, the knowledge acquisition strategies already developed for building intelligent tutors may also be appropriate for learning on demand systems.

One of the major problems in knowledge acquisition results from trying to fit the knowledge of an expert into a representational scheme which was chosen too early in the process. It is important that the content and organization of the system be matched to that of the expert, not vis versa. Consulting written references can be an effective way to form an initial knowledge base, and may be the only technique necessary for learning on demand systems that function as job aids for consulting little-used information before making decisions. If the goal is to construct a learning on demand system that monitors user actions, it will be necessary to consult with domain experts to elicit procedural knowledge and problem-solving strategies. This process should begin by having the expert tutor the knowledge engineer on the domain and terminology. During consultations with the expert, it is also important to pay attention to the expert's non-verbal cues of absorption in the task. Finally, test cases can be presented to the expert in order to elicit and refine the representation of the expert's knowledge (Evanson, 1988; Prerau, 1987).

Common methods for eliciting expert knowledge typically fall into one of two categories: interviews and protocol analyses (Cooke & McDonald, 1986). Both techniques are relatively ill-defined, time-consuming, and susceptible to communication barriers that can inhibit effectiveness, since the methods require experts to engage in introspection (Evanson, 1988). Interviews tend to be unstructured and center on spontaneous questions asked by the knowledge engineer as the expert performs the task (Hoffman, 1987). Success depends on the expert's ability to verbalize while performing, and on the knowledge engineer's ability to ask pertinent questions. Analyzing the data collected can be very complex, and may require using audio or video tapes (Ericsson & Simon, 1984). The expert's non-verbal behaviors may indicate the degree of mental effort required to solve certain problems and provide additional cues about what the expert is actually thinking.

Task analysis methods can be employed to provide information about what experts do during problem solving. As the expert performs a task, the knowledge engineer can observe the problem-solving goals and information used by the expert. Many of the tasks performed by experts fall into the general categories of interpretation, diagnosis, monitoring, prediction, planning and design (Stefik et al., 1983). These activities generally require

experts to perceive patterns, analyze similarities and differences in data, form concepts based on the relationships between data categories, make inferences, and test hypotheses. Task analysis methods can provide information about the basic knowledge and skills necessary for problem solving, but since these methods do not indicate why actions are undertaken, it is often necessary to employ cognitive task analysis to reveal more subtle aspects of the expert's reasoning (Means & Gott, 1988). Scenarios or plans derived from past experience, which experts may recall to make judgments when information is incomplete, can also be identified with such methods.

Several methods have been developed to generate specific representations of the conceptual knowledge structures of experts. Statistical procedures such as multi-dimensional scaling, cluster analysis and link-weighted networks can be used to analyze the proximity of pairwise sets of concepts in order to generate geometric representations of knowledge structures (Schvaneveldt, 1990). A less complex, easier to implement method involves the analysis of pattern notes (Buzan, 1974; Fields, 1980). The expert constructs a "map" of a subject area by free association of ideas related to the main topic. Secondary ideas are indicated by labeled lines extending from the main subject, and subordinate ideas are indicated by other extensions from these. Research has shown that knowledge structures elicited by the pattern note technique produces maps of cognitive structures very similar to those produced by statistical procedures mentioned above (Jonassen, 1987).

Another approach based on Kelly's (1955) personal construct theory utilizes repertory grid techniques to elicit and analyze personal beliefs from individuals (Boose, 1986; Hart, 1986). Constructs (attributes) and elements (instances) about a particular problem domain are accumulated through interviews (face-to-face or computer-based). Experts rate the relationships among the elements for each construct of the problem, and the ratings are assembled in a grid. The grid can be statistically analyzed to determine clusters of elements, and can be reorganized (focused) to better represent the expert's knowledge.

INTERFACE DESIGN FOR LEARNING ON DEMAND SYSTEMS

The design of an interface for a learning on demand system is highly dependent on the type of task environment required and the features being developed for the application. For example, a learning on demand system for an "intelligent" word processing application will differ significantly from a control system for a nuclear reactor. The former might be designed to simply monitor the activities of the user in order to infer what the user might be trying to do when help is requested. The latter might be designed to demonstrate various operations to the user before allowing the user full control of the system, and would monitor the user at all times, regardless of expertise, to prevent fatal errors. These examples illustrate the various levels of abstraction in learning on demand environments (from real representations of the work environment to abstract representations) that Burton (1988) has described elsewhere.

Related to the level of abstraction of the interface is the notion of fidelity. It has been suggested that initial learning requires an interface that is high in fidelity, that is, the system feels, looks, acts, and seems like the actual task environment (Burton, 1988). Since learning on demand systems are in fact the actual work environments, fidelity may be less an issue than for intelligent tutoring systems that rely on simulation of the work environment. However, it is important to maintain task fidelity in many applications, both for the comfort of the user (would you like to draw with a graphics program that required you to type coordinates for every line?), and the possibilities for user modeling afforded by a high fidelity, direct manipulation interface.

There are a number of strategies borrowed from intelligent tutoring systems that can be employed in learning on demand environments to improve interaction and learning. Burton (1988) has described several forms of assistance to learners that may be adapted for learning on demand systems. Besides direct help available on request (and tailored to the current task the learner is completing), strategies might include assistance (where the system does part of the task for the user), empowering tools that record a user's actions and provide browsing of an action sequence for user reflection, modeling of the task (where the system performs the task while the user observes), coaching and critiquing interactions that are initiated by the system when a user makes an error, and direct online tutoring controlled by the system.

Finally, practical considerations for interface design also need to be incorporated into learning on demand systems. A balance between user control and system intervention must be maintained, especially in learning on

demand systems that employ coaching or critiquing. Considerations of the user's work tasks are also important, so that aspects of the learning on demand system do not interfere with the work required by the application. For example, in a decision support system, the information entered to allow the system to help in making a decision should also be available for the work requirements. It would be ridiculous to gather information on a prospective loan applicant, submit that information to an expert system for verification of the decision, then have to fill out addition forms with the same information for purposes of record keeping. Many applications have been designed without these simple, practical considerations, and not surprisingly, such applications are not well received by users.

CONCLUSIONS

While learning on demand offers many possibilities, there are still a number of limitations. Certain essential skills may need to be acquired before work can begin, and since learning on demand is task driven, it may expose learners to isolated pieces of knowledge that require further integration and elaboration. Learners may also encounter difficulties in decontextualizing knowledge so that it can be used in new settings, and learning on demand may not support substantial restructuring of knowledge, since the additional information learned occurs only in relationship to what the learner already knows.

Nevertheless, learning on demand represents a new and relatively untested solution to the problems of learning and instruction cited earlier, and raises many questions and challenges for psychology, education and computer science. It has not been established whether learning on demand is more efficient than conventional forms of learning. Little is known about what classes of users (e. g., novices, intermediates, experts) benefit most from using an integrated learning on demand work environment. What intervention strategies should the system use for providing information without disrupting the work process? When will users suspend the work process and access relevant information? Additional research that addresses these and other questions is needed before learning on demand systems can assume the position of primary instructional role that has been predicted by advocates of these systems.

REFERENCES

Bereiter, C., & Scardmalia, M. (1989). Intentional learning as a goal of instruction. In L. B. Resnick (Ed.), *Knowing, learning and instruction: Essays in honor of Robert Glaser* (pp. 361-392). Hillsdale, NJ: Erlbaum.

Boose, J. H. (1986). *Expertise transfer for expert system design.* New York: Elsevier.

Brown, A. L., & Palinscar, A. S. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. B. Resnick (Ed.), *Knowing, learning and instruction: Essays in honor of Robert Glaser* (pp. 393-451). Hillsdale, NJ: Erlbaum.

Burton, R. R. (1988). The environment module of intelligent tutoring systems. In M. C. Polson & J. R. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 109-142). Hillsdale, NJ: Lawrence Erlbaum.

Buzan, T. (1974). *Use both sides of your brain.* New York: Dutton.

Cooke, N. M., & McDonald, J. E. (1986). A formal methodology for acquiring and representing expert knowledge. *Proceedings of the IEEE, 74*(10), 1422-1430.

Ericsson, K. A., & Simon, H. A. (1984). *Protocol analysis.* Cambridge, MA: M. I. T. Press.

Evanson, S. E. (1988). How to talk to an expert. *AI Expert, 3*(2), 36-41.

Fields, A. (1980). Pattern notes. *NSPI Journal, 19*(8), 12-17, 43.

Fischer, G. (1991). Supporting learning on demand with design environments. In L. Birnbaum (Ed.), *Proceedings of the 1991 International Conference on the Learning Sciences* (pp. 165-172). Charlottesville, VA: Association for the Advancement of Computing in Education.

Fischer, G., Lemke, A. C., Mastaglio, T., & Morch, A. (1990). Using critics to empower users. *Human Factors in Computing Systems: CHI '90 Conference Proceedings* (pp. 337-347). New York: Association for Computing Machinery.

Fischer, G., Lemke, A. C., & Schwab, T. (1985). Knowledge-based help systems. *Human Factors in Computing Systems: CHI '85 Conference Proceedings* (pp. 161-167). New York: Association for Computing Machinery.

Gardner, H. (1985). *The Mind's New Science.* NY: Basic Books.

Gery, G. (1991). *Electronic performance support systems.* Boston: Wiengarten.

Greeno, J. G. (1989). Situations, mental models, and generative knowledge. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing* (pp. 285-318). Hillsdale, NJ: Erlbaum.

Hart, H. (1986). *Knowledge acquisition for expert systems.* New York: McGraw-Hill.

Hoffman, R. R. (1987). The problem of extracting the knowledge of experts from the perspective of experimental psychology. *AI Magazine, 8*(2), 53-67.

Jonassen, D. H. (1987). Assessing cognitive structure: Verifying a method using pattern notes. *Journal of Research and Development in Education, 20*(3), 1-13.

Kelly, G. A. (1955). *The psychology of personal constructs.* New York: W. W. Norton.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. *Psychological Review, 95,* 163-182.

Lave, J. (1988). *Cognition in practice.* Cambridge, UK: Cambridge University Press.

Means, B., & Gott, S. P. (1988). Cognitive task analysis as a basis for tutor development: Articulating abstract knowledge representations. In J. Psotka, L. D., Massey, & S. A. Mutter (Eds.), *Intelligent tutoring systems: Lessons learned* (pp. 35-58). Hillsdale, NJ: Erlbaum.

Nelson, W. A. (1989). Artificial intelligence knowledge acquisition techniques for instructional development. *Educational Technology Research and Development, 37* (3), 81-94.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* New York: Basic Books.

Prerau, D. S. (1987). Knowledge acquisition in the development of a large expert system. *AI Magazine, 8*(2), 43-51.

Schoen, D. A. (1983). *The reflective practitioner.* New York: Basic Books.

Schvaneveldt, R. W. (Ed.). (1990). *Pathfinder associative networks: Studies in knowledge organization.* Norwood, NJ: Ablex.

Stefik, M., Aikins, J., Balzer, R., Benoit, J., Birnbaum, L., Hayes-Roth, F., & Sacerdoti, E. (1983). Basic concepts for building expert systems. In F. Hayes-Roth, D. A. Waterman, & D. B. Lenat (Eds.), *Building expert systems.* Reading, MA: Addison-Wesley.

Suchman, L. A. (1987). *Plans and situated actions: The problem of human-machine communication.* Cambridge, UK: Cambridge University Press.

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Los Altos, CA: Morgan Kaufmann.

Improving the Explanation Capabilities of Advisory Systems[1]

Bruce Porter
Art Souther
Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712

Abstract

A major limitation of current advisory systems (e.g., intelligent tutoring systems and expert systems) is their restricted ability to give explanations. The goal of our research is to develop and evaluate a *flexible* explanation facility, one that can dynamically generate responses to questions not anticipated by the system's designers and that can tailor these responses to individual users. To achieve this flexibility, we are developing a large knowledge base, a viewpoint construction facility, and a modeling facility.

In the long term we plan to build and evaluate advisory systems with flexible explanation facilities for scientists in numerous domains. In the short term, we are focusing on a single complex domain in biological science, and we are working toward two important milestones: 1) building and evaluating an advisory system with a flexible explanation facility for freshman-level students studying biology, and 2) developing general methods and tools for building similar explanation facilities in other domains.

1 Research Objectives

The goal of our research is to develop and evaluate a *flexible* explanation facility that can dynamically generate responses to questions not anticipated by the system's designers and that can tailor these responses to individual users. Previous advisory systems have lacked these capabilities for a variety of reasons. In this section we will describe the problems of current advisory systems, the solutions to these problems that we propose, and our research activities for achieving those solutions.

Problems. The explanation facilities of current advisory systems are inflexible for two reasons:

- Inadequate domain knowledge: At least two factors limit the adequacy of the knowledge base as a source of "raw materials" for flexibly generating explanations: small size and task specificity. Although small size is an obvious limitation, few research projects have built a large-scale knowledge base as their "starting point" for research on explanation. Furthermore, because the knowledge for most advisory systems supports only a single task, most research on explanation has overlooked issues outside the task requirements, such as answering a range of questions, explaining terminology, and customizing explanations for specific users [22]. (For notable exceptions see work by Moore and Swartout [33, 24].)• Inability to reorganize knowledge: Little work has been done to develop methods to select coherent packets of knowledge from a knowledge base, and even less on the reorganization of portions of the knowledge base to improve specific explanations. These issues have been avoided by "hardwiring" knowledge

[1]Support for this research was provided by the Air Force Office of Scientific Research (contract number F49620-93-1-0239) and the National Science Foundation (grant number IRI-9120310)

structures that are suitable for the limited explanations required by a particular advisory system. (For notable exceptions see work by McKeown [21] and Suthers [32].)

Solutions. We are developing a five-part solution to the problems of current advisory systems. Our solution comprises: (1) constructing a knowledge base which is large-scale and contains very fine-grained representations, (2) selecting and organizing knowledge with viewpoints and models, (3) generating new viewpoints on demand, (4) constructing and simulating models and using them to explain the behavior of mechanisms, and (5) generating explanations which relate new information to what the user already knows. We discuss each of these in turn.

First, we have built an extensive knowledge base for one area of biology—college-level anatomy and physiology of plants [26]. Although it is under constant development, it is already one of the largest knowledge bases in existence. (Our knowledge base currently contains about 3,000 frames and over 28,000 facts.) Unlike knowledge bases built with instructional frames [14] or hypertext [10], our knowledge base consists of "atomic facts" that our explanation facility can combine in different ways to produce different explanations.

Second, we are developing methods for selecting information from the knowledge base and organizing it into a coherent bundle appropriate to the situation at hand. One organizing structure is that of *viewpoints*, which provide coherent descriptions of objects or processes. For instance, the viewpoint "photosynthesis as a production process" selects and organizes facts to explain how photosynthesis produces glucose from carbon dioxide and water. Another organizing structure is that of *models*, which are built from viewpoints and support computer simulation. For example, an energy flow model of the plant includes the viewpoints "photosynthesis as an energy transduction process" and "respiration as an energy transfer process," and it allows an advisory system to predict and explain the effects of changes in light wavelength on a plant's photosynthetic or respiratory rate under a variety of specific circumstances .

Third, we are developing methods to automatically generate *new* viewpoints. This ability is important because, as system designers, we cannot anticipate all the viewpoints necessary for effective explanations. For example, Table 1 lists several viewpoints on photosynthesis and the situations in which they might arise. Our question answering facility will be able to construct these viewpoints by selecting and reorganizing the individual facts comprising existing viewpoints in the knowledge base (see [1]).

Fourth, we are developing methods for automatically constructing and simulating models and interpreting the consequences of simulations. These methods use existing methods of qualitative reasoning, but add two new capabilities: constructing models from large knowledge bases and generating explanations from these models. This will allow our explanation facility to answer "what-if" questions that were unanticipated when the knowledge base was built (see [28]).

Finally, we are developing methods to automatically generate *integrative explanations*, which explicitly relate new information to what the user already knows. This is important to advisory systems because the coherence of an explanation depends upon the particular situation. Our system will record the discourse with each user and will explain new topics in ways that relate to that user's knowledge and interests (see [18]).

2 The Design of Our Advisory System

An advisory system that simply provides facts to a user fails to take advantage of established techniques for effective communication. These techniques include treating the user as an active learner, grounding new information within a relevant context, and conveying information in appropriate ways through an interface which is intuitively easy to use. If the advisory system is to be used in a learning situation, it also needs to motivate the user with an appealing environment.

To provide these capabilities, our advisory system is designed with layers of software between the knowledge base and the user, each providing an essential capability for a flexible, reactive advisory system (see Figure 1). The outermost layer is the *discourse generator*, which interacts with the user by presenting focused information and encouraging the user to ask questions and to explore additional issues germane to the topic. To generate the relevant knowledge within an appropriate context and provide alternate modes of presentation, the discourse

Viewpoint on Photosynthesis	*Contextual Situation*
as a destructive process	To explain the effects of the first oxygen producing plants on other organisms during evolution.
as an essential process in ecosystem energy flow	To explain how almost all living things depend on photosynthesis for deriving energy from an abiotic source.
as a magnesium-utilizing process	To explain the effects of magnesium deficiency on the plant.
as an enabling process	To explain how photosynthesis is important for any processes which use glucose or oxygen.
as a constructive process	To explain how photosynthesis is vitally important to plant growth and reproduction.

Table 1: A few of the viewpoints on photosynthesis and the teaching situations in which they might be appropriate.

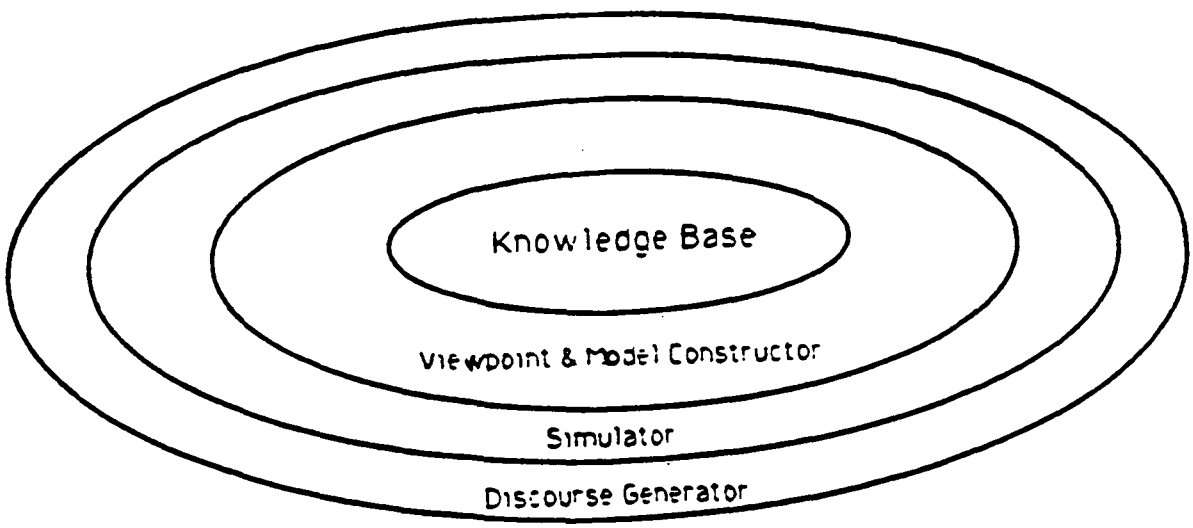


Figure 1: The layered design of our proposed advisory system. Each layer of software can access and use any layers within it

generator uses information from the inner layers. The *modeler and simulator* predict and explain the behavior of biological systems by using computational models to answer "what if" and "why" questions; they permit the user to directly investigate the predictions of a model by manipulating its parameters. The *viewpoint constructor* selects and organizes domain information into coherent explanations. Many of these viewpoints may be directly encoded in the *knowledge base*. Others will be constructed by reorganizing the facts comprising existing structures.

This section describes the capabilities of each layer of software, and our current prototypes, beginning with the knowledge base.

2.1 A Knowledge Base for Biology

At the core of any advisory system is a knowledge base. It contains both the information to be communicated to the user and the information required for effective communication, such as the background knowledge required to understand particular concepts.

For many domains, building a knowledge base is difficult and time consuming. To avoid this difficulty, most system designers have built advisory systems in subject areas for which a small knowledge base will suffice [35, 34, 4, 7, 6, 29, 16, 27, 25]. These subjects fall into two categories. The first is task-specific subjects that focus on a single application of knowledge. For example, the Guidon system [9] teaches diagnosis of infectious blood diseases. Teaching other tasks, such as how to determine a patient's prognosis, would require substantial changes to the system because Guidon is specialized for its single task. The second category of subjects is formally characterizable subjects that require only a small set of logical rules or axioms. For example, the GEOMETRY system [2] requires only a few rules of introductory geometry. However. the fundamental knowledge in a field like biology is neither committed to performing a single task nor formally characterizable with a small set of axioms. We believe that we con overcome the inherent difficulty in building a large knowledge base for two reasons: 1) we have developed sophisticated software that assists us in viewing and editing large, fine-grained knowledge bases; 2) we have used this software to build a large knowledge base and applied our prototype systems for explanation generation to it.

2.2 The Viewpoint Constructor

A knowledge base for basic science must represent multiple *viewpoints* of each concept. For example, encoded in the Biology knowledge Base are many different viewpoints of photo-synthesis. Two of these, which we mentioned earlier, are "photosynthesis as a production process" and "photosynthesis as an energy transduction process." The knowledge base also contains more focused viewpoints that are appropriate in certain situations, such as "photosynthesis as a *glucose* production process" and "photosynthesis as an *oxygen* production process ."

Figure 2 suggests why viewpoints are useful and even essential. The figure shows just part of the knowledge about photosynthesis that is encoded in our Biology Knowledge Base. Taken altogether, the totality of knowledge about photosynthesis is incoherent—there are so many facts about photosynthesis that some focus is necessary. Viewpoints provide this focus. The figure shows the two viewpoints of "photosynthesis as production" and "photosynthesis as energy transduction," highlighted with solid and dashed bold lines, respectively. Each collects and organizes facts about the basic process of photosynthesis that are relevant to that particular point of view and omits the large number of other facts that are irrelevant from that point of view. For example, "photosynthesis as production" focuses on the compounds, oxygen and glucose, that are produced by photosynthesis and on the compounds, carbon dioxide and water, that are its raw materials, and omits intermediate compounds, such as ATP that participate in photosynthesis but are, overall, neither produced nor consumed. This viewpoint also omits much other information about photosynthesis that is irrelevant to viewing photosynthesis as production.

A viewpoint then, is a collection of facts about a particular concept that are all relevant within a particular context. The focus that viewpoints provide is critical because an arbitrary collection of facts is usually incoherent, even when the facts all pertain to the same topic. For example, describing photosynthesis as "a process that converts light energy into glucose and oxygen" is not patently incorrect but is confused or incoherent in that it intermixes facts from the viewpoints of energy flow and material flow. It is better to say that photosynthesis converts light energy into chemical bond energy (the energy transduction viewpoint), or that it converts carbon dioxide and

water into glucose and oxygen (the production viewpoint). The *viewpoint constructor is* the part of our system that processes requests for viewpoints and produces the appropriate collection of facts selected from all facts in the knowledge base.

Many researchers acknowledge that viewpoints are a useful way of organizing knowledge. However, most methods for retrieving viewpoints from a knowledge base assume that each viewpoint is explicitly encoded [33, 23, 20]. Unfortunately, the difficulty of explicitly encoding viewpoints increases combinatorially with the number of concepts in the knowledge base. In addition, relying on pre-encoded viewpoints is inflexible because new viewpoints cannot be created as needed.

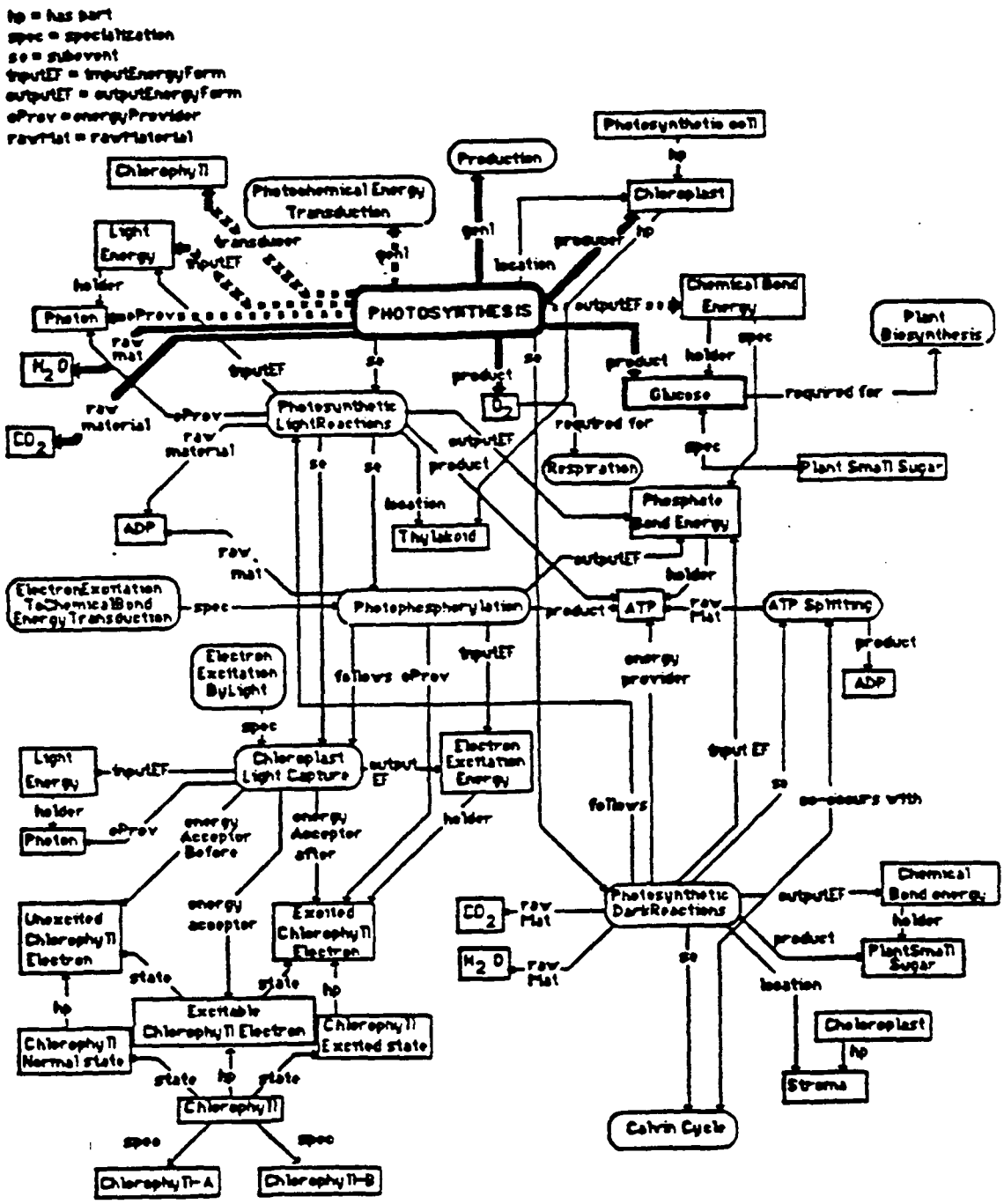


Figure 2: A small portion of the knowledge about photosynthesis represented in the Biology Knowledge Base.

These labeled graphs, or "semantic networks", are widely used in artificial intelligence. Each fact is a relation (depicted as a labeled arc or line) between two concepts (depicted as labeled boxes). Solid bold lines represent information that is part of the viewpoint "photosynthesis as production", while the dotted bold lines represent the viewpoint of "photosynthesis as energy transduction".

Our solution to this problem is to enable the advisory system to dynamically generate viewpoints when they are needed. We have experimented with methods for doing this using abstract specifications for points of view, called *view types*. For example, the *structural* view type specifies methods for constructing viewpoints concerning an object's parts and their interconnections, such as the viewpoint "endosperm as part of a seed." Similarly, the *functional* view type specifies methods for constructing viewpoints concerning the role of an object in a process, such as "chloroplast as the producer in photosynthesis."

View types can also be combined. The *structural-functional* view type specifies how the individual parts of an object participate in the subevents of some process. For example, a structural-functional description of angiosperm sexual reproduction would discuss how each part of the flower (sepals, petals, stamen, and carpels) participates in some event of the reproductive process (e.g., pollinator attraction. pollen formation, and pollination).

We believe that a relatively small number of such view types is sufficient to characterize and produce many viewpoints within the natural sciences. Support for this conjecture is preliminary but encouraging. First, we found that our view types and their combinations are sufficient to characterize over fifty definitions chosen at random from the glossary of a biology textbook. Second, we have successfully used view types in a prototype system for generating viewpoints [1, 30]. These viewpoints constitute answers to a wide range of definational questions (e.g., "What is C3-photosynthesis?") and comparative questions (e.g., "What is the difference between mitosis and meiosis?").

2.3 The Modeler and Simulator

Our advisory system will use computational models to predict and explain the behavior of complex biological systems. This capability is very important because it can tie together otherwise disparate and uninteresting facts into an explanation of how something works.

Most computational models in biology are *quantitative* models, which interrelate a system's parameters using differential equations. Although these models are precise, they can also be intractable? especially if some of the equations are nonlinear. Moreover, because quantitative models require complete numeric data, model builders must assume precise values for parameters for which little precise data may be known. Finally, the quantitative details often obscure the more important qualitative principles.

During the past ten years, research on *qualitative* models has addressed these problems [15, 13, 11]. Instead of using exact relationships and values, qualitative models employ qualitative relationships, such as "water potential increases with turgor," and qualitative values, such as "cell turgor is positive and decreasing." Approximations like these are frequently sufficient to express essential information about a system when complete knowledge is unavailable or unnecessary. They also enable a qualitative simulator to characterize the behavior of a system, much as a human reasoned could, without knowing or needing exact relationships or values. For example, a qualitative simulator with a model of a plant's water flow could predict that "excessive transpiration from a plant caused by increasing temperatures will be countered by closing of the stomata" without knowing the original concentration of water in the plant or the exact rate of transpiration. Qualitative models have been used in advisory systems for steam-plant operation [31], weather prediction [5], circuit diagnosis [3, 35], and many other domains.

We are extending the research on qualitative reasoning in two ways. First, while previous research assumes that a model is given *a priori*, we are developing methods for constructing models as needed. In order to support a wide range of questions, our knowledge base must provide a vast array of viewpoints and levels of detail. However, overly detailed models, while perhaps capable of answering many questions, can be inefficient or even intractable, and excess detail would make their predictions opaque. Our program uses each question to decide

which perspectives and abstractions are needed, constructs a model from these pieces, and simulates this model to answer the question (see [28]). Such a model not only answers the question, but also highlights the knowledge supporting the answer and provides transparent, explainable answers.

Second, we are developing methods to generate in-depth explanations of qualitative reasoning. A major shortcoming of current simulators (both qualitative and quantitative) is that they generate extensive details about a model's behaviors but little overview or explanation. Our system will provide concise and focused textual answers to a range of questions about a model and its behaviors. For example, we expect to provide multilevel overviews of both a model and its behaviors which highlight their most important features and compare and contrast different behaviors (if there is more than one). We also expect to provide an explanation of the mechanisms by which a model causes its behaviors, grounded in familiar physical principles, and how a model would respond to changed circumstances (see [19]).

2.4 The Explanation Generator

Our overriding goal is to develop and evaluate a *flexible* explanation facility that can dynamically generate responses to questions not anticipated by the system's designers and that can tailor these responses to individual users. We are building an explanation generator that will achieve flexibility in three ways. First, it will produce *integrative explanations* that relate new information to the user's existing knowledge. In producing an integrative explanation, we can define three networks of relevant concepts and relations. The *target* network is the set of concepts and relations that a system seeks to communicate to the user. The *base* network is the set of concepts and relations that model what the user already understands and is relevant in some way to the target. The *linking* network is the set of concepts and relations that relate the target to the base. To produce an integrative explanation, our system will determine the relevant target, linking, and base networks, and it will organize the knowledge in the linking and target networks in a manner that facilitates their integration into the base network.

Opportunism is the second way that our explanation generator will achieve flexibility. The system will actively seek opportunities to include important information in the domain that is closely related to the topic being explained but is unknown to the user. For example, suppose the system were explaining embryo sac formation to a user, and noticed that two participants in this process, a megaspore and a megaspore mother cell, are both kinds of botanical cells. It can recognize this as an opportunity to discuss the difference between haploid and diploid cells, an important distinction in biology. Moreover, rather than interjecting this discussion in the middle of another topic, the system can relocate it to an appropriate place in its explanation.

Finally, our explanation generator will achieve *organizational flexibility.* Such flexibility is desirable for two reasons. First, a generator should be able to introduce prerequisite material and elaborations at appropriate positions in the explanation. Second, it should be able to place material that is familiar to the user earlier in the explanation and material that is new to the user later. To achieve organizational flexibility, the generator takes a *delayed-commitment* approach: it delays organizational commitments as long as possible. Initially, the propositions of the explanation are organized very loosely. As the explanation develops, the generator adds new propositions and gradually arranges them in an order that is most suitable for the user.

We are aided in our efforts to construct an explanation generator by previous research results on user modeling and natural language generation. An *overlay* model [8] represents what the user knows as a subset of the concepts in the knowledge base. The explanation generator initializes the user model with basic concepts covered in previous courses and lessons, and updates the model based upon explanations that it generates and questions the user asks. Also, we are using the FUF system [12] for converting explanation structures into English. FUF, which has been in development at Columbia for the past seven years, employs one of the largest machine grammars ever constructed and provides wide linguistic coverage.

We have constructed a prototype system, which provides integrative explanations, opportunism, and organization flexibility [17, 18]. We have used this system to produce multi-paragraph explanations from portions of the Biology Knowledge Base. Because the system is not restricted to schemas, it generates different explanations for different users. The system's output was favorably evaluated by a domain expert, who found the explanations both accurate and clear.

3 Evaluating and Generalizing Our Results

Our long-term objective is to build advisory systems for complex domains that compete well with human advisors. Although we cannot meet this objective soon, we believe We can build and evaluate the core components of an advisory system that competes well with textbooks for an important portion of a course, and that meeting this short-term objective is a critical milestone for achieving our long-term objective.

We plan to evaluate our advisory system by using it to help teach an introductory biology course at the University of Texas at Austin. In addition to introductory material, the system will explain advanced material that has not been covered in the classroom or assigned readings.

The evaluation will be based on data from the following experiment. Users will be paid to spend extra time in the course studying the advanced material with the help of the advisory system. When the users are comfortable using the system, we will give them several assignments. Each assignment will require answers and explanations for a range of technical questions on both the introductory and advanced material. (These questions will be formulated by a biologist who is not affiliated with our project. Our research team will not know the questions beforehand.) To complete their assignments, the users will be randomly assigned to three groups. Users in the "traditional" group will be permitted to use any standard (non-human) resources, such as textbooks and laboratory equipment. The "advisory" group will be allowed to use only the advisory system, and the "eclectic" group will be allowed to use both traditional sources and the system.

We will compare the performance of the three groups of users on correctness and completeness of answers and on efficiency of task completion. The users' answers and explanations will be judged by the teaching staff for the biology course, who will not be apprised of the users' identity or group. If a benefit for the advisory system is found, we will separately analyze user performance on the introductory material to see if a benefit exists even when the material has been covered in the classroom. Including the eclectic group will further allow us to ascertain whether there is a synergistic effect among the three sources of information—classroom, textbook, and advisory system. The users' proficiency in terms of the amount of time used to complete the assignment will be measured, controlling for the correctness of the users' responses. For each of the three groups, we will also measure the users' interest in the advanced materials taught. This assessment will be based on questions from standard course evaluations.

Based on the results of our evaluation, we will generalize our research results to help others build advisory systems in a range of domains. This will involve removing dependencies on the domain of biology that our experience will no doubt reveal and re-implementing those parts of our system that contributed most to its success, to improve its portability and ease of reuse.

4 Summary

The primary results of this research will be the following: (1) an explanation facility for college-level biology, (2) a critical evaluation of the explanation facility based upon its use in an introductory biology course at the University of Texas, and (3) general methods and tools for building similar explanation facilities in other domains.

During the last six years, we have built a very large knowledge base for one area of biology and we have developed prototype systems for each component of our proposed explanation facility. From this experience, we have learned how to structure large knowledge bases using viewpoints and models, and We have created a foundation on which to build a flexible explanation facility.

Our proposed explanation facility will dynamically generate responses to unanticipated questions and tailor these responses to individual users. This flexibility will encourage a user to ask questions and request clarification or detail. In the future we expect this functionality to be the foundation for a wide range of computer-based advisory and research tools, such as intelligent databases, electronic libraries, and simulated laboratories.

References

[1] L. Acker and B. Porter. Access methods for large knowledge bases. *Artificial Intelligence*, (submitted), 1992.

[2] J. Anderson, C. Boyle, and G. Yost. The geometry tutor. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1-7, 1985.

[3] J. Brown, R. Burton, and A. Bell. Sophie: A step towards a reactive learning environment. *International Journal of Man-Machine Studies*, 7:675-696, 1975.

[4] J. Brown, R. Burton, and J. de Kleer. Pedagogical, natural language, and knowledge engineering techniques in Sophie I, II, and III. In D. Sleeman and J. Brown, editors, *Intelligent Tutoring Systems*. Cambridge, MA: Academic Press, 1982.

[5] J. Brown, R. Burton, and F. Zdydel. A model-driven question-answering system for mixed-initiative computer-assisted instruction. *IEEE Transactions on Systems, Man, and Cybernetics*, 3:248-257, 1973.

[6] R. Burton. Diagnosing bugs in a simple procedural skill. In *Intelligent Tutoring Systems*. London: Academic Press, 1982.

[7] R. Burton and J. Brown. An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, 11:5-24, 1979.

[8] B. Carr and I. Goldstein. Overlays: A theory of modeling for computer aided instruction. Technical Report AI Lab Memo 406 (Logo Memo 40), Massachusetts Institute of Technology, 1977.

[9] W. Clancey and R. Letsinger. Neomycin: Reconfiguring a rule-based expert s-stem for application to teaching. In *Seventh International Joint Conference on Artificial Intelligence*, pages 829-836, 1981.

[10] J. Conklin. Hypertext: An introduction and survey. *IEEE Computer*, 20(9):17-41, 1987.

[11] J. DeKleer and J. Brown. A qualitative physics based on confluences. *Artificial Intelligence*, 24:7-83, 1984.

[12] M. Elhadad. *Using Argumentation to Control Lexical Choice: A Functional Unification Implementation*. PhD thesis, Department of Computer Science, Columbia University, 1992.

[13] K. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85-168, 1984.

[14] G. Kearsley. Intelligent computer-aided instruction. In Stuart C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 154-159. John Wiley and Sons, New York, 1987.

[15] B. Kuipers. Qualitative reasoning: Modeling and simulation with incomplete knowledge. *Automatica*, 25(4):571-585, 1989.

[16] P. Langley, J. Wogulis, and S. Ohlsonn. Rules and principles in cognitive diagnosis. In N. Frederiksen, editor, *Diagnostic Monitoring of Skill and Knowledge Acquisition*. Hillsdale, New Jersey: Lawrence Earlbaum Associates, 1987.

[17] J. Lester and B. Porter. Generating integrative explanations. In *Proceedings of the AAAI Workshop on Explanation*, pages 80-89, 1990.

[18] J. Lester and B. Porter. A revision-based model of instructional multi-paragraph discourse production. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, August 1991.

[19] R. Mallory. Generating structured, causal explanations of qualitative simulations: A research proposal. Technical Report (forthcoming), Department of Computer Science, University of Texas at Austin, 1993.

[20] K. McCoy. The role of perspective in responding to property misconceptions. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence. pages* 791-793, 1985.

[21] K. McKeown. Discourse strategies for generating natural language text. *Artificial Intelligence,* 27:142, 1985.

[22] K. McKeown and W. Swartout. Language generation and explanation. *Annual Review of Computing Science,* 2:401-409, 1987.

[23] K. McKeown, M. Wish, and K. Matthews. Tailoring explanations for the user. In *Proceedings of the International Joint Conference on Artificial Intelligence,* pages 794798. Palo Alto, California: Morgan Kaufmann, 1985.

[24] J. Moore and W. Swartout. A reactive approach to explanation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* pages 1504-1510, 1989.

[25] B. Porter, R. Bareiss, and R. Holte. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence Journal,* 45(2):229-263, 1990.

[26] B. Porter, J. Lester, K. Murray, K. Pittman, A. Souther, L. Acker, and T. Jones. AI research in the context of a multifunctional knowledge base: The Botany Knowledge Base project. Technical Report AI88-88, University of Texas at Austin, 1988.

[27] B. Reiser, J. Anderson, and R. Farrell. Dynamic student modeling in an intelligent tutor designed for LISP programming. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* pages 8-14, 1985.

[28] J. Rickel and B. Porter. Using interaction paths for compositional modeling. In *Proceedings of the Sixth International Workshop on Qualitative Reasoning about Physical Systems,* pages 82-95, 1992.

[29] D. Sleeman. Inferring student models for intelligent computer-aided instruction. In *Machine Learning: An Artificial Intelligence Approach,* pages 483-510. Palo Alto, California: Morgan Kaufmann, 1983.

[30] A. Souther, L. Acker, J. Lester, and B. Porter. Using view types to generate explanations in intelligent tutoring systems. In *Proceedings of the Eleventh Cognitive Science Conference,* pages 123-130, 1989.

[31] A. Stevens and B. Roberts. Quantitative and qualitative simulation in computer-based training. *Journal of Computer-Based Instruction,* 10(1):16-19, 1983.

[32] D. Suthers. Providing multiple views of reasoning for explanation. In *Proceedings of the International Conference on Intelligent Tutoring Systems,* pages 435-442, 1988.

[33] W. Swartout. Xplain: A system for creating and explaining expert consulting programs. *Artificial Intelligence,* 21(3):285-325, 1983.

[34] K. vanLehn and J. Brown. Planning nets: a representation for formalizing analogies and semantic models of procedural skills. In R. Snow, P. Frederico, and W. Montague, editors, *Aptitude, learning, and instruction: Cognitive process analyses.* Hillsdale NJ: Lawrence Earlbaum Associates, 1980.

[35] B. White and J. Frederiksen. Qualitative models and intelligent learning environments. In R. Lawler and M. Yazdani, editors, *AI and Education.* New York: Ablex Publishing Co., 1987.

VIS/ACT : The Next Episode'

Tucker Maney & Henry Hamburger
Naval Center for Applied Research in Artificial Intelligence

ABSTRACT

VIS/ACT is a multi-media educational system for aircrew coordination training (ACT). Students view video segments, answer questions that are adjusted to individual performance, and engage in related activities. Although the system puts the student in a reactive critiquing role, it has proved effective in improving performance on active targeted ACT skills, in group simulation tasks. VIS/ACT itself is the product of coordination among three Navy agencies.

OVERVIEW

The importance of aircrew coordination is widely accepted. A 1987 investigation "found that human error was the cause of between 60-80% of all aircraft incidents and accidents" and another recent study "concluded that these incidents and accidents are not a result of the inability to fly, but rather ineffective aircrew coordination." These quotes are from course materials prepared for the Navy Training Systems Center (NTSC) by MacCuish and Morgan (1991). In response to this problem, we have designed and implemented VIS/ACT, a video-oriented instructional system for ACT. The system itself is the product of coordination and close collaboration among instructional designers at NTSC, pilot-educators at the Atlantic Fleet Helicopter Operations School (AFHOS) and artificial intelligence specialists at the Naval Research Laboratory. Our respective roles are shown in Figure 1, a diagram of the development process.

VIS/ACT is a multi-media educational system that shows a video episode while the student categorizes and critiques the performance of the participants. Initially built with domain independent features, it has been specialized to aircrew coordination training (ACT). The system as specialized is intended as part of an NTSC curriculum in which the classification system is introduced earlier, in the classroom. The system permits flexible use of NTSC videos of aircrew episodes under the guidance of our computational tutor. The role of the tutor is to assure that the student absorbs the specific and general points of crew coordination that the video episodes are intended to convey. These points have been substantially elaborated by AFHOS personnel, who have also helped us to redesign the system to meet the actual needs of training, as well as the needs of these experts during the knowledge acquisition phase. The system runs under Hypercard and Lisp on a Mac-II. A PC-VCR provides computer control of on-screen videos.

SYSTEM

The specific purpose of VIS/ACT is to enhance pilots' aircrew coordination skills by increasing their ability to recognize and classify these skills. The NTSC course identifies 44 concrete, observable behaviors, each of which is associated with one of seven defined categories of crew coordination: mission analysis, leadership, decision making, assertiveness, situation awareness, communication and flexibility. The system (Figure 2) provides a framework in which students can watch on-screen videos of crews in action, classify the observed behaviors, and get immediate commentary on the accuracy of their observations. The current system evolved from a more broadly applicable implementation that allowed for an expert to enter domain information and specify various aspects of pedagogical strategy.

In order to provide appropriate feedback to the student, the system requires extensive knowledge about the contents of the video being viewed. A knowledge acquisition interface provides an environment in which an

ACT expert can mark the time intervals of major tape segments and of individual episodes within these segments. Because more than one crew coordination skill may be displayed in a short video clip, the expert enters a question that will be posed to the student in order to focus him or her on a specific crew activity. The expert then classifies the activity by specifying which crew member's actions are being addressed, whether or not good aircrew coordination skills are being displayed, and the specific skills and subskills that are involved in the targeted activity. He also provides a comment addressing the important aspects of the crew interchange, and an explanation of his reasoning for his classification choices.

In preparing a video tape for use by the student, the expert tries to mark all possible ACT activities, not all of which can or need to be viewed by a student in a single session. The system allows the expert to choose what percentage of the marked events should be shown at any given time, and what proportion of these, if any, should be biased toward the student's crew position or error record within the session. Based on these numbers and the information acquired about each video episode, the system, prior to each major tape segment, selects the video clips that the student will view within that segment.

A student interface allows students to view a video of a mission and to answer questions about each of its episodes. Required student entries can be adjusted to any of three levels of difficulty. Novices are only asked to judge whether or not good ACT skills are being employed. Intermediate students additionally identify the particular aircrew coordination skills involved in the episode. Potential ACT instructors must also select the relevant subskills for each skill they select (Figure 3). At each student level, the system commentary, generated using appropriately selected language templates, contains three kinds of information. First it presents the expert's narrative comment about the episode and the reasoning behind the expert's classification choices. The response then includes a comparison of the student's answer to the appropriate part of the expert's evaluation. In order to begin preparing students to master the next level of difficulty, the answer to both beginning and intermediate students incorporates the skill classification information that will be asked for at the next higher level. Upon completion, the system tells the student how well he or she did overall and in comparison to others of the same crew position. After informing the student of both the strong and the weak areas of performance, VIS/ACT recommends a next course of action.

As the student progresses through the session, the system updates two databases. One contains information about how successful students are at correctly answering individual questions and identifying the skills and subskills involved in the related video episode. The other keeps track of the ability of students in the same crew position to recognize a particular skill. The expert frequently names multiple skills and subskills for a single video episode. Since the student is expected to do likewise, partial credit must be allocated, in both the skill and subskill identification tasks, for selecting parts of the correct answer; additionally, credit is reduced for errors of commission. The credit formula is quite complex because the system needs equitable ratings at the level of subskill, skill and overall student performance.

RESULTS

Initial testing of VIS/ACT took place in February, 1993, at AFHOS. Testing objectives included validation of expert entries, determination of system refinement specifications, and evaluation of teaching effectiveness. Over 50 naval helicopter trainees were tested, with strongly favorable reactions, on the whole. Beyond mere acceptance, indications are that the system was effective in improving performance on targeted skills.

In part of the aircrew coordination training program, pairs of students fly a mission using a low-level simulation of a cockpit. During the mission, an instructor guides the crew through a prescribed scenario, and observes their use of aircrew coordination skills in completing the mission. When one of four training sessions that are part of the regular course was replaced by a VIS/ACT session, students showed a marked improvement in their use of crew coordination skills during the simulated mission, as judged on an objective observational scale by the AFHOS training instructors. Early indications are that despite placing the student in the role of a critic, outside the situation, use of the system improves performance on active skills, specifically the spontaneous use of appropriate communication behaviors in simulated missions. This result is consistent with others' findings that

"situated training in virtual contexts similar to the environments in which learner's skills will be used helps their knowledge to transfer." (Dede, 1992)

Initially the major value of VIS/ACT was thought to be in retraining as opposed to initial training. Used in either context, the system is designed for individual instruction. However, in order to expose more students to the system and thus get a more meaningful evaluation, the educators, in the later stages of testing, began using it in the classroom. By projecting the computer screen image to a larger screen, they were able to use the system to generate discussion in the classroom. In their view, using the system in this way increased the interaction among students and improved the effectiveness of the group session.

FUTURE

Some enhancements to the ACT specialization need to be made, including being able to track a student's progress over a series of sessions. Making the shift from VCR to CD-ROM would make the system more user-friendly by reducing delays. More importantly, it could provide immediate access to a library of mission videos, permitting the system to give the student a greater variety of opportunities to focus on his weak areas. Another objective is to provide avenues for increased student interaction. A proposed system would combine video with student voice communication by stopping a video at crucial points in a crew coordination activity, and having the student voice his response. The speech recognition and natural language processing that would be essential to such a system appear to be almost within the state of the art, as represented by Bates (1993). Not only syntactic and semantic analysis, but even discourse phenomena, would need to be handled in determining whether the student's communication conveyed the information known to be needed in the particular situation and evaluating other aspects of its appropriateness. An important notion here is that of a "target" communication, currently under experimental development at the NTSC. A target is a particular kind of communication behavior that is anticipated as the correct response to a particular event sequence. The target strategy has the potential of reducing the system's task to a manageable level.

PERSPECTIVE

It seems worthwhile to place VIS/ACT in the context of a variety of approaches to multi-media educational systems. With the advent of video on the computer screen comes the opportunity to turn computer-based tutoring and learning systems into .. television! In the light of television's dubious effects on students, we may wonder: will multi-media be the solution or part of the problem? Clearly, video on the computer screen is not an educational panacea. The exciting challenge is to place it under flexible control, in the service of educational goals. We see six ways to do this, one of which is the VIS/ACT strategy of providing and supervising relevant tasks, in order to give the student an active role in the proceedings, to assure that s/he is still receiving input, and to adjust the demands on the student as appropriate. Our approach, like several of the others touched on next, makes use of short clips of video material.

Approaches to avoiding the eyes-glaze-over behavior sometimes observed in response to straight television are to mingle it with: (i) a human teacher, (ii) itself, (iii) knowledge, (iv) structure, (v) manipulation and (vi) a relevant task, as in VIS/ACT. We elaborate on each of the first five just a bit, in turn. First, the teacher may mingle segments of video with question-and-answer, discussion and the like. For mixing video with itself, we have in mind the recombinant segments in the Athena's Project stories for language learning, which let the student have some control over the situation and which make a literary virtue out of computational shortcomings of not being quite sure what the student is up to (Murray, 1987). Mixing video with knowledge is represented by the strategy of Parkes and Self (1990), who ground tiny clips of video in a complex artificial intelligence-style knowledge representation. Mixing video with structure is the hyperization approach that Lynn Fontana (1991) is pursuing by making the not-so tiny clips of the Kenneth Burns Civil War tapes available at various nodes of a hypermedia semantic network. Finally, the manipulation approach of graphical user interfaces with learning environments, pioneered in Steamer (Holland, Hutchins, and Weitzman, 1984), is really an alternative to video, but can perhaps be usefully wedded to it.

ACKNOWLEDGMENTS

We thank our collaborators, Tom Franz of the Naval Training Systems Center and Rob Lowry and Stephen Byers of the Atlantic Fleet Helicopter Operations School. They were absolutely crucial to the success of the application. We also acknowledge the support of the Office of Naval Technology.

References

Bates, M. (1993) Proceedings of the ARPA Human Language Technology Workshop, Plainsboro, NJ, March 21-24, 1993.

Dede, C.J. (1992) The Future of Multimedia: Bridging to Virtual Worlds. Educational Technology, May, 1992, 54-60.

Fontana, L.A. (1991) The Civil War Interactive. Instruction Delivery Systems, November/December 1991, 5(6), 5-9.

Holland, J. H., Hutchins, E.L., and Weitzman, L. (1984) Steamer: An interactive inspectable simulation-based training system. AI Magazine,5,15-27.

MacCuish, D. A. and Morgan, B. B. (1991) Aircrew Coordination Observer Training Program for Instructor Pilots. Orlando, FL: Naval Training Systems Center.

Murray, J.H. (1987) Humanists in an institute of technology: How foreign languages are reshaping workstation computing at MIT. Academic Computing.

Parkes, A. P. and Self, J.A. (1990) Towards "interactive video": A video-based intelligent tutoring environment. In Frasson, D. and Gauthier, G. (Eds.), Intelligent Tutoring Systems : At the Crossroads of Artificial Intelligence and Education. Norwood, NJ: Ablex Publ. Corp.

Figure 1 ACT: 3-way Navy Collaborative Effort

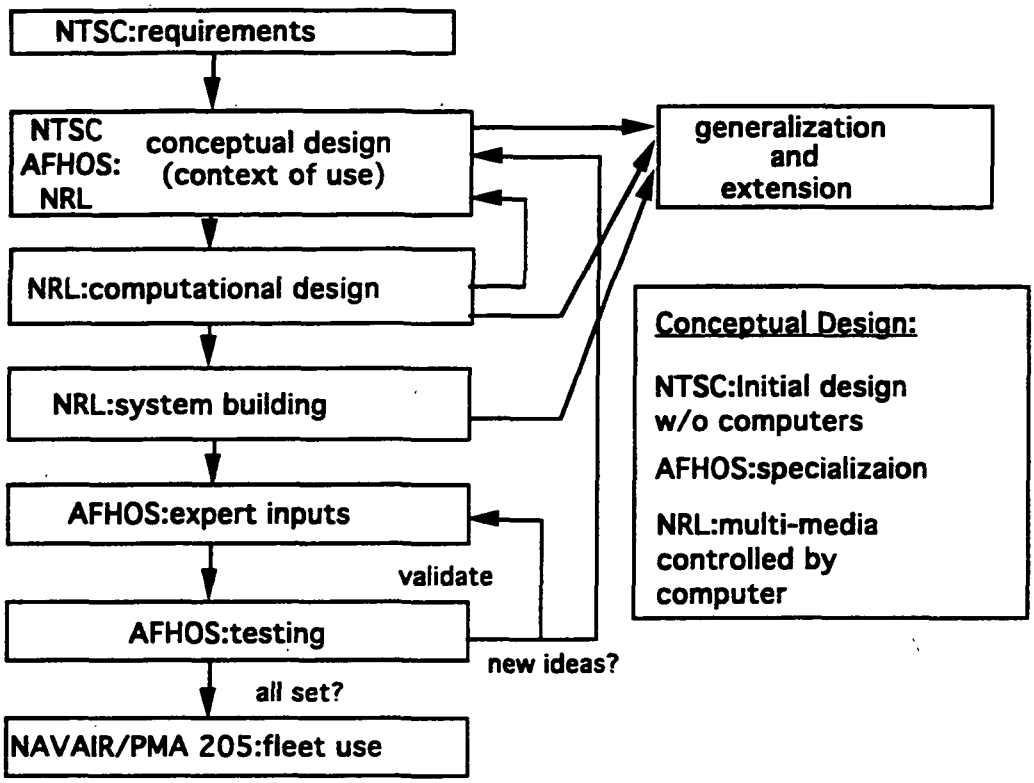


Figure 2 VIS/ACT: System Architecture, showing information flow

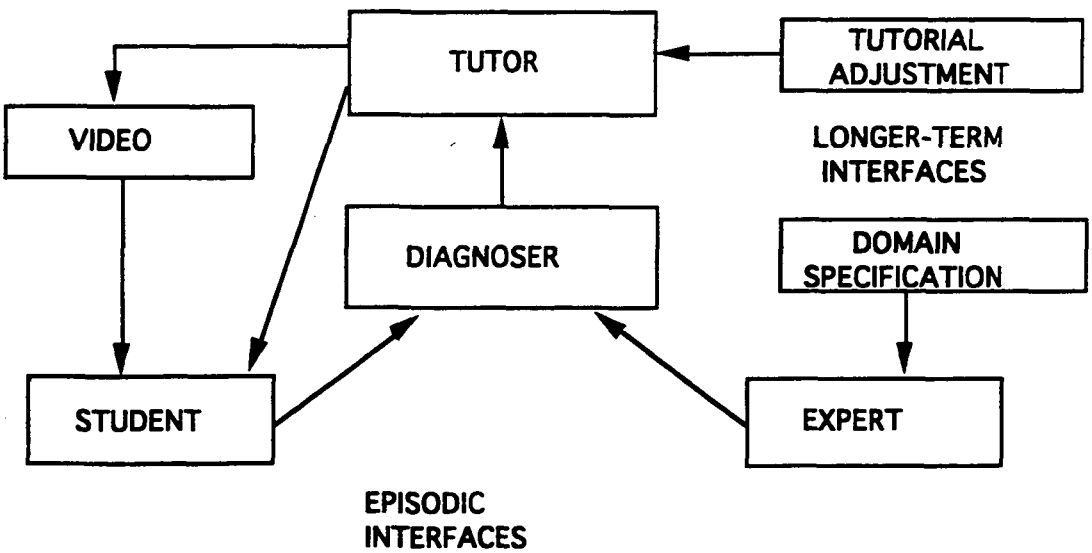
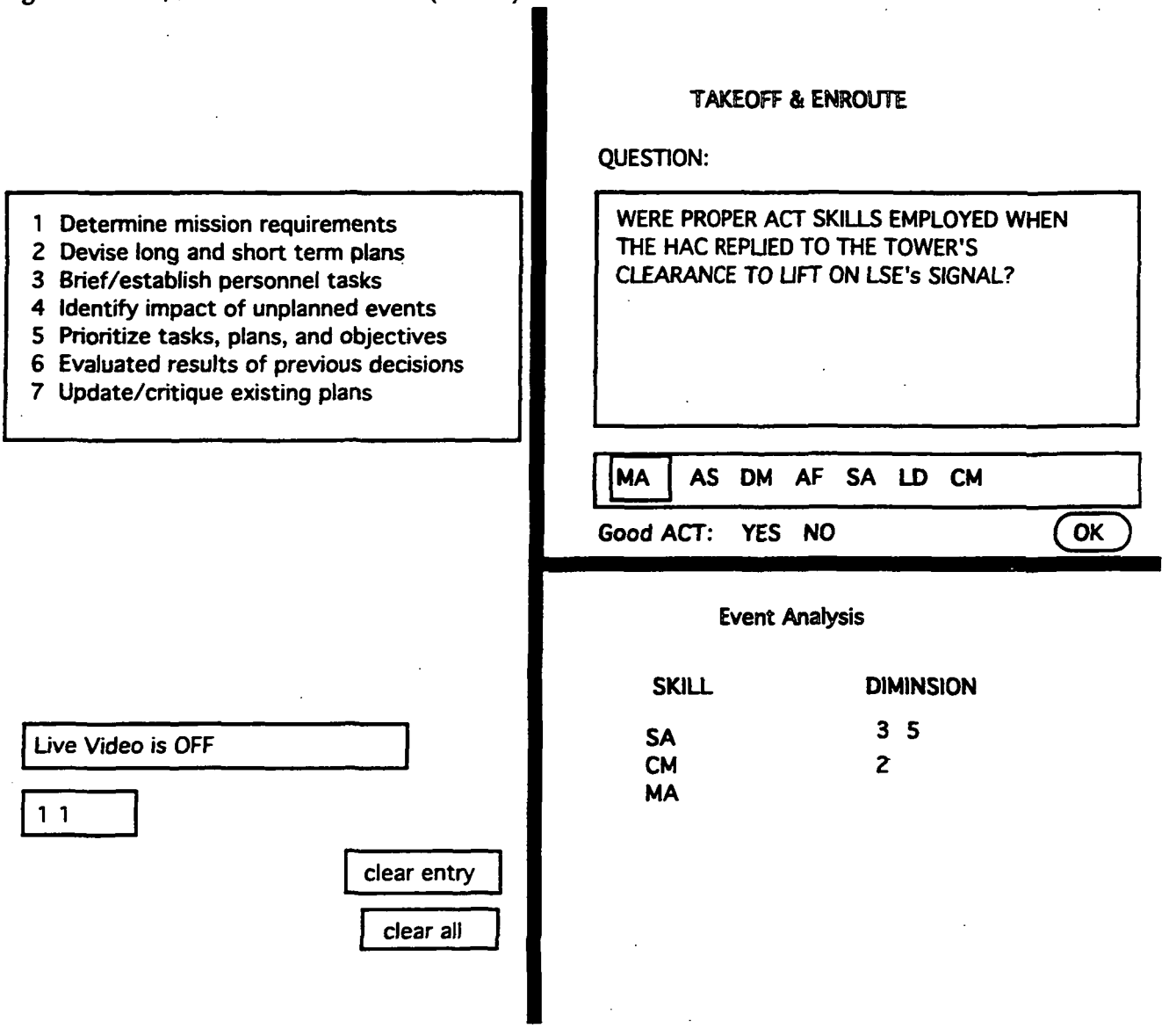


Figure 3 *VIS/ACT: Student Interface (Level 3)*



MACH III: Past and Future Approaches to Intelligent Tutoring'

Sylvia Acchione-Noel
U.S. Army TRADOC Analysis Command -
White Sands Missile Range

Joseph Psotka
U. S. Army Research Institute

Abstract

In 1986, the U.S. Army Research Institute created an intelligent tutoring system as a proof-of-concept for artificial intelligence applications in Army training. The Maintenance Aid Computer HAWK Intelligent Institutional Instructor (MACH III) taught student mechanics to maintain and troubleshoot the AN/MPQ-57 High Power Illuminator Radar (HPIR) of the HAWK Air Defense Missile System. In 1989, TRADOC Analysis Command compared the effectiveness of MACH III to traditional paper-based troubleshooting drills. For the study, all students received lecture and hands-on training as usual. However, during troubleshooting drills, students traced faults using either MACH III or the traditional paper-based method. Class records showed that the MACH III group completed significantly more troubleshooting tasks and progressed through tasks of greater difficulty than the paper-based group. Upon completion of training, students took written, practical, and oral essay tests. Mean test scores showed that students performed similarly regardless of the drill method used. However, significantly different standard deviations showed that the MACH III group performed more consistently than the paper-based group. Furthermore, significantly different time measures showed that the MACH III group reached faster troubleshooting solutions on the actual radar transmitter than the paper-based group. We will present the study results and discuss how updating the design of MACH III can include desktop computing in a virtual environment.

Introduction

In 1986, the U. S. Army Research Institute created an intelligent tutoring system (ITS) as a proof-of-concept for artificial intelligence applications in Army training. Called the Maintenance Aid Computer HAWK Intelligent Institutional Instructor (MACH III), it supported transmitter and receiver instruction in a radar maintenance course. The MACH III taught student mechanics to maintain and troubleshoot the AN/MPQ-57 High Power Illuminator Radar (HPIR) of the HAWK Air Defense Missile System.

The design of MACH III continued the philosophy behind STEAMER (Hollan, Hutchins, and Weitzman, 1984; Psotka, Massey, and Mutter, 1988). MACH III designers focused on user mental models; they emphasized conceptual rather than physical fidelity; and they aimed to construct generic tools (e.g., the conduit program) as much as possible. Also, in the STEAMER tradition, MACH III provided students with a graphical interface to interact with an inspectable simulation and a troubleshooting expert system. Just as STEAMER graphics displayed the movement of steam through pipes, MACH III graphics displayed the movement of electric current through radar components.

Students interacted with MACH III using its keyboard, mouse, and two monitors. Interaction occurred through three different modes: magic mode, real-life mode, and demonstration mode. The magic mode permitted the students to test and replace a variety of radar components through direct interaction with the model-based

simulation and view the results. It's "magic" nature enabled students to explore components that they ordinarily could not explore on the radar itself. In the real-life mode, interaction with the model-based simulation was mediated through the troubleshooting expert system. Students could test and replace components only as they would on the radar. However, they could receive one of three types of feedback: advice, critique, or both advice and critique. Also, in the demonstration mode, students could request the troubleshooting expert system to perform the next appropriate step(s) of a task.

MACH III was designed to help student mechanics develop the appropriate mental models for troubleshooting a radar. If successful, it would enable students to conceptualize radar signal loops-within-loops and to apply this knowledge to obtain faster, more efficient solutions. In 1989, TRADOC Analysis Command compared the effectiveness of MACH III to traditional paper-based troubleshooting drills (Acchione-Noel, 1991a; Acchione-Noel, 1991b; Acchione-Noel, Saia, Williams, and Sarli, 1990). The first half of this paper reports that quantitative evaluation of MACH III in a real-world setting under controlled conditions.

Method

The evaluation compared the training effectiveness of two courses. The traditional course contained lectures, training on the radar itself, and paper-based troubleshooting drills. In the paper-based drills, students traced symptoms through the manuals and schematics. The other course also contained lectures and training on the radar, but students performed troubleshooting drills with MACH III. In this case, students traced a symptom and tested components simulated by MACH III software. Manuals served as references. Lectures and radar training remained identical for the two groups. Therefore, any differences in performance could be attributed to the supplemental instruction. The evaluation focused on which method provided effective and efficient training--troubleshooting on paper or troubleshooting on MACH III.

Twenty-nine students with American citizenship and no previous radar maintenance skills participated in the data collection between December 1989 and May 1990. Previous exam scores and other demographic data were used to achieve stratified random assignment of students to the two courses.

Instructors of the MACH III and paper-based groups recorded each student's progress through a troubleshooting task list created for the study. The troubleshooting task list contained a lengthy list of easy, medium, and difficult radar symptoms for the students to troubleshoot. Students worked with partners to accomplish tasks on the radar, and then, rotated to MACH III or to the paper-based method to perform additional tasks. The instructors recorded the task name, the date, the training method used (MACH III, paper, or radar), and the start time and end time.

Approximately 4 days were spent troubleshooting transmitter malfunctions and 3 days were spent troubleshooting the receiver. Instructors encouraged students to accomplish as many tasks on the troubleshooting list as possible. Also, instructors determined which tasks should be done and in what order, based on their teaching experience and the constraints of the rotation scheme. Students worked with "easy" tasks at first, but eventually progressed to the "medium" and "difficult" tasks of each circuit.

Standard practical examinations for the course required students to identify transmitter and receiver parts and describe their function. They also required students to perform check procedures, symptom recognition, troubleshooting, and signal tracing. Students could use all pertinent manuals and schematics, but were limited to 45 minutes for the total examination. To control for ceiling effects, additional practical examinations contained troubleshooting problems of greater difficulty and specific malfunctions that the students had not seen before. The problems did not coincide step-for-step with the manual. Students were required to think on their own and fill in logical steps where the manual left a gap.

The standard paper-and-pencil examination covered 20 multiple-choice transmitter questions randomly selected from a test question databank. The examination prompted students about where to find answers in the manuals and schematics; however, students had one hour to complete the test. To control for ceiling effects, additional paper-and-pencil examinations covered 20 transmitter and 20 receiver questions hand-selected from the databank

for greater difficulty. These latter examinations gave no prompts as to where answers could be found and lasted one hour each. All examination measures included percent correct and completion times.

Results

Examination scores. Twenty-nine students participated in the training and standard examinations. However, only 27 students were present for the additional examinations. Table 1 presents the mean scores ($\bar{X}$) of the practical and the paper-and-pencil examinations for the two groups. Recall that the additional examinations were designed to be more challenging than the standard examinations. Generally, students scored lower on the additional examinations than on the standard examinations, regardless of group.

Three out of seven F tests showed that the variances (s^2) of the MACH III group distributions were significantly smaller than those of the paper-based group. Significance differences in variance meant that the standard deviations (s) differed significantly as well. The differences made important implications for the effectiveness of MACH III. They indicated that students of the MACH III group performed more consistently than students of the paper-based method of instruction.

Table 1 Mean scores and standard deviations of examinations.

Examinations	Paper-Based Group			MACH III Group			Test for s^2 Differences
TRANSMITTER	n	$\bar{X}$%	s	n	$\bar{X}$%	s	F
Std. Practical	14	93	18.6	15	95	5.1	13.49*
Added Practical	13	71	32.9	14	75	24.9	1.75
Std. Paper & Pencil	14	85	8.8	15	88	8.2	1.15
Added Paper & Pencil	13	79	10.0	14	79	5.1	1.0
RECEIVER	n	$\bar{X}$%	s	n	$\bar{X}$%	s	F
Std. Practical	14	93	18.1	15	99	2.1	71.42*
Added Practical	13	95	11.8	14	98	3.6	10.69*
Added Paper & Pencil	13	66	13.0	14	70	10.8	1.45

*Significance, $p < .05$.

The seven sets of examination scores were converted to z scores to meet the assumption of equal variance prior to performing analyses of variance. When comparing the groups, the MACH III group tended to have higher examination scores than the paper-based group, but univariate analyses of variance performed on the z scores showed that these differences were not statistically significant. A multivariate analysis of variance was also performed on the z scores of all seven examinations, but Pillai's Trace was not significant. Also, Mann-Whitney U tests on the ranked times showed no significant group differences based on completion times.

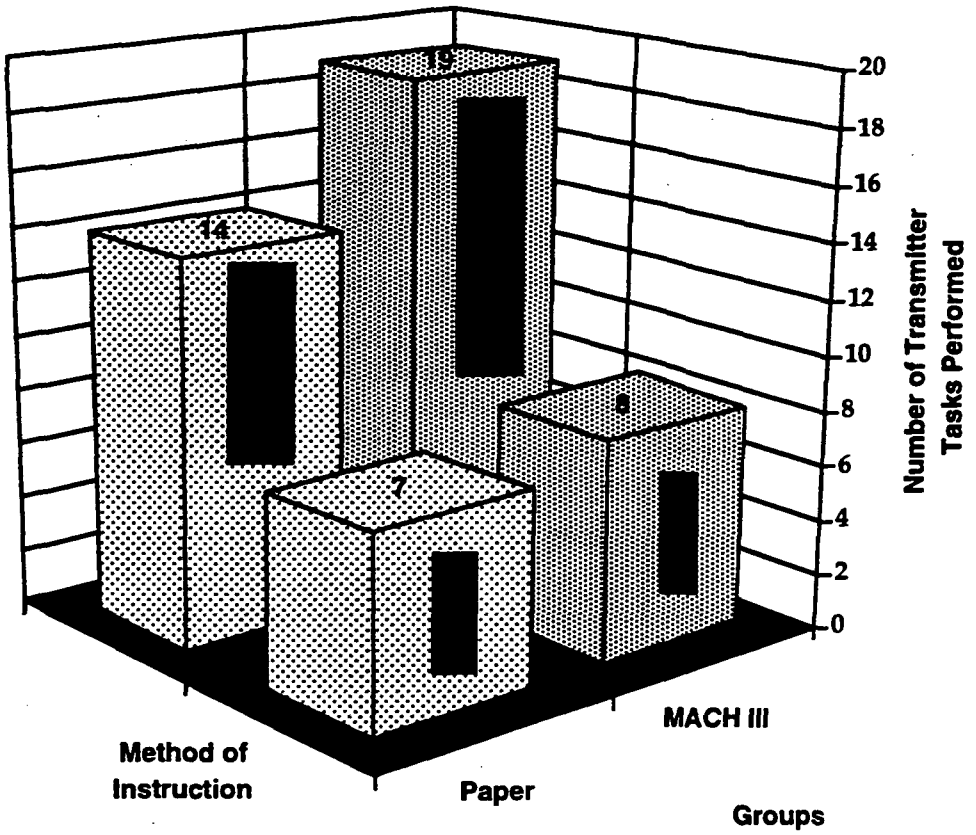


Figure 1. Mean number of transmitter tasks performed.

Number of tasks performed. The troubleshooting task list contained 25 transmitter tasks and 26 receiver tasks. In many cases, students completed the tasks for that day's subject area early, so additional tasks were assigned. Progress on these additional tasks was still recorded by when and how they were performed, but the tasks themselves were not identified. The following analysis is based on the number of tasks completed in drills by 14 paper-based students and 15 MACH III students.

The troubleshooting task list contained easy, medium, and difficult radar symptoms for the students to troubleshoot. During transmitter troubleshooting, the MACH III group averaged 13 easy and 13 medium/difficult tasks. By comparison, the paper-based group averaged 11 easy tasks and 9 medium/difficult tasks. Figure 1 shows that the MACH III group performed 1.4 times as many transmitter tasks using their supplemental method of instruction as the paper-based group did, $t(27) = 4.15$, $p < .05$.

During receiver troubleshooting, the MACH III group averaged 11 easy and 6 medium/difficult tasks. The paper-based group averaged only 8 easy tasks and 3 medium/difficult tasks. Figure 2 shows that the MACH III group performed 2.4 times as many receiver tasks as the paper-based group did using their supplemental instruction. The difference in number of tasks completed was significant, $t(27) = 8.05$, $p < .05$. These results indicate that the two methods of instruction were vastly different in terms of training efficiency. Not only did the MACH III group receive more troubleshooting practice, they received more challenging practice because of the efficiency of MACH III.

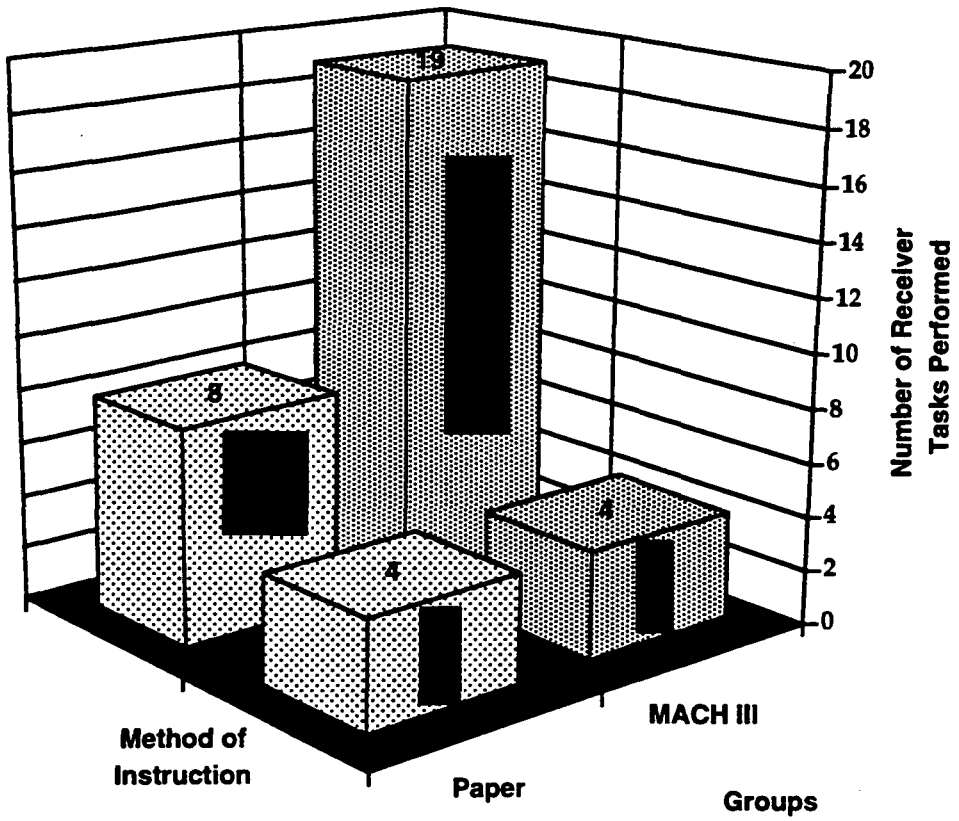


Figure 2. Mean number of receiver tasks performed.

Time spent between tasks. Because the two groups differed so much in amount of practice, the length of time between tasks was examined. Analysis indicated the median time between tasks was 8 minutes for the MACH III group and 30 minutes for the paper-based group. The time delay resulted from the paper-based group's dependence on outside sources for task information. The students who sat at the desk depended on their peers at the radar for information about meter readings, lamp lights, etc. Until students at the radar had progressed far enough in a task to determine certain information, students who troubleshooted on paper could not proceed. Once the students on the radar relayed the information, the students at the desk tended to complete the task very quickly. In fact, the students who troubleshooted on paper often sat idle for several minutes while the students on the radar finished the actual troubleshooting procedures.

By contrast, the students on MACH III worked independently from their peers on the radar. All the information needed to complete the tasks was contained in MACH III's simulation of the radar. Because of the dependency of the paper-based method on hands-on performance, the paper-based group progressed more slowly than the MACH III group did.

Time to perform tasks. Next, the time taken to complete troubleshooting tasks was examined. Recall that instructors determined which tasks should be done and in what order. Since instructors skipped around in the task list, the two groups did not always perform the same tasks. The following analysis reports the median time based on only those specific tasks which both groups performed. The analysis of common tasks included 14 transmitter tasks and 10 receiver tasks.

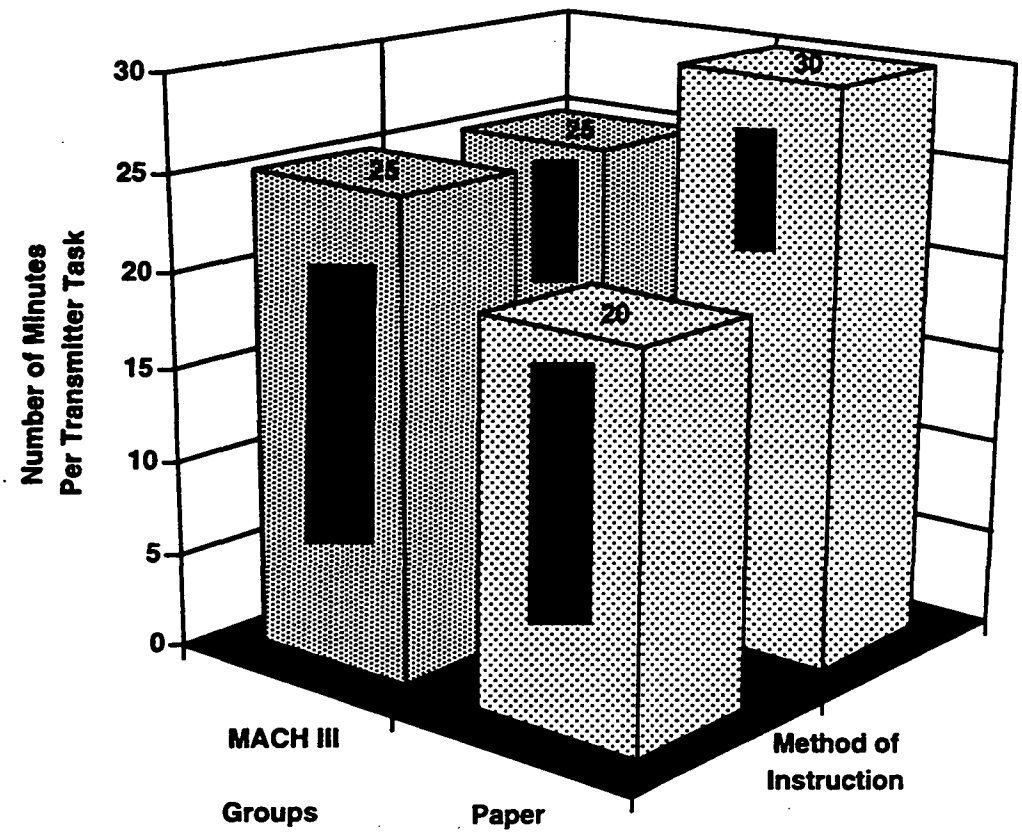


Figure 3. Median time to perform transmitter tasks.

Figure 3 shows that the paper-based group performed transmitter tasks on paper 5 minutes faster than the MACH III group did on MACH III. A Mann-Whitney U Test of the ranked times was significant ($p < .05$). This savings occurred partly because paper-based students often took shortcuts in procedures. According to the instructors, once the students at the desk had received essential information from the radar, they were free to skip steps they had seen before.

In MACH III's software, however, the fault isolation procedures had to be followed, much like the actual procedures performed on the radar. Little, if any, opportunity existed for shortcuts or glossing over procedures. Each step in the procedures required an active response from the student. Furthermore, the troubleshooting times reflected the level of task difficulty. Medium and difficult tasks took longer to perform than easy tasks. Because MACH III students performed more medium and difficult tasks than paper-based students, their average times were longer. Together, the difficulty level and the lack of shortcuts slowed performance on MACH III.

Performance time on the radar's transmitter was a different story. The MACH III students performed transmitter tasks 5 minutes faster than the paper-based group. A Mann-Whitney U Test of the ranked times was significant ($p < .05$). The MACH III students had performed so many tasks on the MACH III they were likely to perform similar tasks on the radar itself. The resulting practice effect may have helped MACH III students isolate faults more quickly than the paper-based students. Further, the students may have applied new, more efficient troubleshooting strategies they had learned through MACH III.

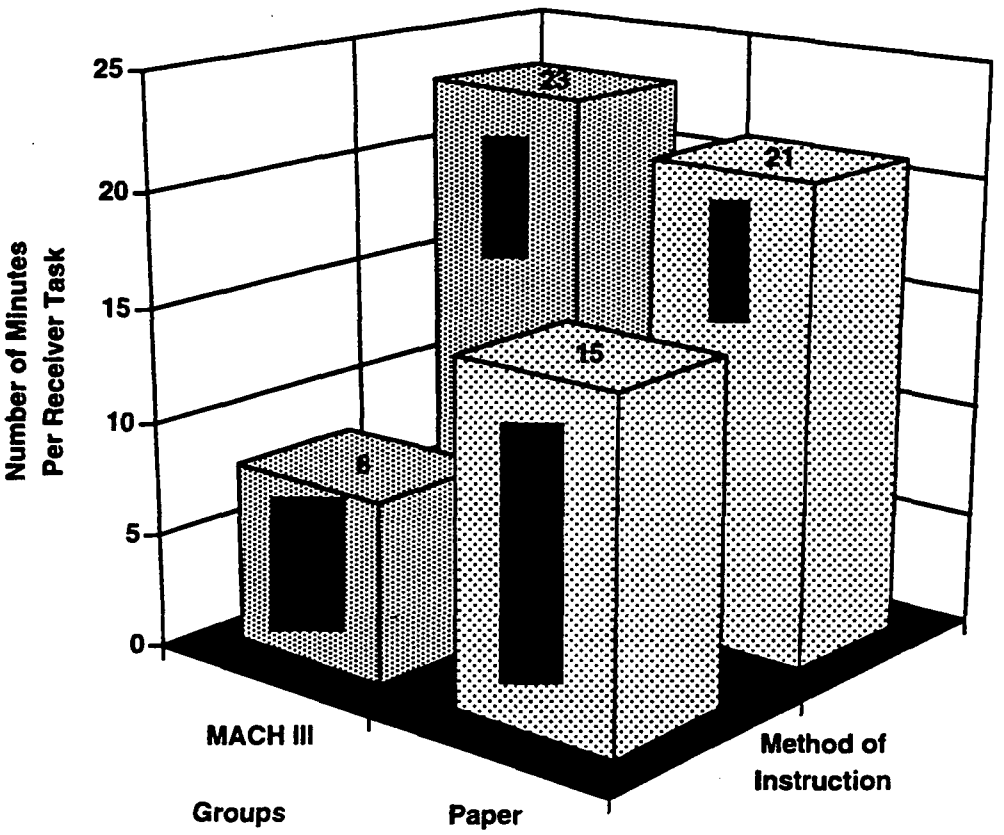


Figure 4. Median time to perform receiver tasks.

Figure 4 shows that the MACH III group performed receiver tasks on MACH III 7 minutes faster than the paper-based group did on paper. A Mann-Whitney U test of the ranked times was significant ($p < .05$). Instructors reported that once MACH III students had performed one lamp test on the receiver simulation, they could perform all remaining tasks without consulting the manuals. Also, the MACH III students probably had grown accustomed to using MACH III before receiver training began. As a result, they lost no time due to lack of familiarity with the training device. Figure 4 also shows that the paper-based group performed receiver tasks on the radar 2 minutes faster than the MACH III group did. This difference was not significant.

EVALUATION CONCLUSIONS

Overall, MACH III provided a more structured and time efficient method of instruction than the paper-based method. Given the same course length, more troubleshooting tasks were accomplished using MACH III. Although, more tasks did not produce higher examination scores, MACH III instruction resulted in greater consistency of performance overall and faster performance on the transmitter tasks.

Arguably, the same case can be made for the use of intelligent tutoring systems (ITS) in general. That is, an intelligent tutoring system can help to eliminate low scoring performances. Also, the greater task coverage afforded by an ITS can help students develop accurate and efficient mental models. When students attempt actual troubleshooting for the first time, their ITS practice can lead to a time savings on the real task.

Evaluation postscript. During the evaluation, instructors provided some insight into how MACH III might be revised. They noted that fault isolation checks required mechanics to walk to different sides of the radar to

monitor the status of lamps. However, the MACH III did not allow students to "view" the radar from all sides. As a result, an instructor suggested that future improvements to MACH III should more closely represent the dynamic and visual aspects of such procedures. Unknowingly, the instructor had voiced the demand for a virtual training environment.

VIRTUAL REVISIONS

REFERENCES

Acchione-Noel, S. C. (1991a) The Benefits of Training Radar Mechanics with an Intelligent Tutoring System. Rapallo (Genova), Italy: Proceedings of the Sixth International PEG Conference: Knowledge Based Environments for Teaching and Learning.

Acchione-Noel, S. C. (1991b) Intelligent tutoring: Effective and efficient training for radar mechanics. Fort Lee, VA: Proceedings of the Army Operations Research Symposium.

Acchione-Noel, S. C., Saia, F. E., Williams, L. A., & Sarli, G. G. (1990) Maintenance Aid Computer HAWK Intelligent Institutional Instructor Training Developments Study, TRAC-WSMR-TEA-90-051. White Sands Missile Range, NM: U.S. Army TRADOC Analysis Command-WSMR.

Psotka, J., Massey, L. D., Mutter, S. A. (Eds.) (1988) Intelligent Tutoring Systems: Lessons Learned. Hillsdale, N. J.: Lawrence Erlbaum Associates Publishers, pp. 451-478.

Hollan, J. D., Hutchins, E. L., & Weitzman, L. (Summer 1984) STEAMER: An Interactive Inspectable Simulation-Based Training System. The AI Magazine, pp. 15-27.

BIOGRAPHY

Sylvia Acchione-Noel has conducted training effectiveness analyses for the US Army since 1981. She can be reached at US Army TRAC-WSMR, ATTN: ATRC-WGB, White Sands Missile Range, NM 88002-5502. DSN 258-5065. Commercial phone: (505) 678-5065. E-mail address: acchions@wsmr-emh91.army.mil.

Training Mix Model

Philipp A. Djang, Walter G. Butler Richard R. Laferriere

"That was not the first time that I fought a battle. I was able to fight battles at the NTC, I had taken platoons down tank tables, CALFEXs, simulations and on the ground....but, if you training realistically, then my first battle was fought on a simulator or a range versus a battlefield like Iraq"

Platoon Leader, Ghost Troop, Second Armored Calvary Regiment, during a DARPA debriefing following the Battle of 73 Easting.

Introduction

The U.S. Army is faced with the requirement to develop training strategies that will produce highly trained soldiers within certain resource restrictions. The extensive use of training aids, training devices, simulators and simulations (TADSS) promises to be a means of developing innovative training strategies that will allow the U.S. Army to achieve its training goal in an environment of reduced resources. However, it is often the case that a specific TADSS is proposed, developed and fielded individually, without systematic consideration of how the TADSS will fit into the existing suite of training devices.

Training developers need a tool to assist them in determining how to mix existing, new and proposed training systems into cost and effective training strategies. The tool would indicate which strategies fall within budget constraints, what the tradeoffs are between training effectiveness and cost, especially for new TADSS, and suggest new strategies in light of reduced force levels or TADSS availability. This need is especially relevant today, as the Army and other services can expect continued budget and personnel reductions.

Background

The Commanding General, U.S. Army Training and Doctrine Command (TRADOC) chartered the Simulator/Simulation-based (SIM2) study[1]. The motivation for the study is based upon the increasing costs associated with Army training, and in particular, the cost of training ammunition and system use life cycle costs (OPTEMPO). To counteract these costs, the Army has made a substantial investment in TADSS. The interest in developing a tradeoff methodology was summarized in a high level Department of Army memorandum: "This [training] expense, coupled with the fielding of sophisticated and expensive training devices, simulators, and simulations and recent funding constraints, has elevated the interest in justifying these expenses and defining tradeoffs now and in the future. Specifically, Congress and Secretary of Defense continually ask Army leadership: 'What is the tradeoff or payback in live ammunition for your investment in simulators' and 'How can the Army reduce high dollar training expenditures without incurring additional expenses or degrading readiness?' To date, our answers have been based on experience and without specific quantifiable analysis to adequately defend our program, as will be necessary in the future."[2] In his tasking memorandum, the Commanding General, TRADOC defined the focus of the SIM2 study effort as: "The study will focus upon brigade-level and below training simulators and simulations used to train crewman, crews, and command staffs. It will determine quantities needed, their application, resource impact, training transfer, tradeoffs and alternative approaches to meet the Army's total integrated training and education needs. The alternatives will be analyzed on the basis of cost and effectiveness."[3] Furthermore, the CG, TRADOC recognized that the Training Development (TD) community did not have the tools to evaluate training device tradeoffs and directed the

---

[1]Commander, Headquarters, U.S. Army Training and Doctrine Command, unpublished memorandum entitled, '*Simulator/Simulation-based Training Study*', Fort Monroe, Virginia, August 1990.

[2]DA-ODCSOPS Memorandum, April 1989. Official unpublished correspondence. Department of the Army, Office of the Deputy Chief of Staff for Operations, The Pentagon, Washington, D.C., p.1.

[3]Commander, Headquarters, U.S. Army Training and Doctrine Command, op. cite.

development of a methodology that could assess a training strategy and determine the cost and training effectiveness of a mix of training devices and methods. The U.S. Army TRADOC Analysis Command - White Sands (TRAC-WSMR) was directed to plan, develop and execute the SIM2 study.

Problem

The Army must maintain an adequately trained force in a time of severely diminished funding. TADSS have been proliferated throughout the various branches of the Army, but it is not clear how best to use them to meet the overall training need. There is a requirement for an analysis of the current state of device-based training and a determination of how to proceed with the integration of current TADSS and the development and fielding of future TADSS.

Without an analytic methodology, the Army's training strategies and their supporting TADSS will continue to be a disconnected process. The Army cannot be certain that the most cost and effective training strategies are being implemented. A comprehensive analysis that determines how to integrate TADSS into a cost and training effective strategy will provide the Army leadership with the basis for budget requests and resource decisions.

Objectives

The SIM2 study has three study objectives: 1) to develop an analytical system that examines the relationships between the training effectiveness of a suite of TADSS and their associated costs, 2) to determine a cost and training effective mix of devices that will train soldiers to standard on all tasks and 3) to identify the benefits and costs associated with incorporating specific devices or methods into a training strategy.

Approach

The methodology is a synthesis of mathematical programming and military task training. A mathematical programming model was developed to determine appropriate mix of training devices and incorporates cost and effectiveness data. The title that has been adopted for the mathematical programming model is the Training Mix Model. Given a set of training devices and simulators, a mission essential task list, the effectiveness of the devices with respect to the task list and the operational and procurement cost of the devices, a mixed integer linear programming model selects the least expensive mix of devices such that all soldiers are trained on all tasks to military standard. All of the mission critical tasks and their associated resources that must be trained to standard are represented in the model.

The effectiveness of the training devices are determined through a knowledge representation process that elicits expert judgement at a basic level; these data are integrated with respect to each task. Expected proficiency levels are calculated based upon a generalized exponential decay function (learning curve) and a probability distribution that models training frequency.

Cost data combine the program acquisition unit cost and a training session cost. The program acquisition unit cost is the battalion's share of the cost of procuring the training device. The training session cost is a composite of the OPTEMPO, ammunition and device usage costs that are associated with one session of training with a particular device.

The mixed integer linear program was formulated and incorporates these data. The program contains constraints that insure that all soldiers are trained to military standard on all tasks and that certain real-world training requirements (such as field and live fire training) can be assimilated into the training strategy.

Based upon the data provided, the model produces an optimal training strategy. The training strategy identifies which training devices were selected, how much training on each device is needed, what the strategy will cost, and how many concurrent training days will be required to complete the strategy for each level. The training strategy shows the contribution of each training mode to overall effectiveness and cost. In addition, the tasks that influence the solution at each level can be identified. These tasks, referred to as forcing tasks, are those which

require specific training modes to be included in the solution. By developing alternative or innovative devices or methods for these tasks, the training developer could develop less expensive training strategies.

Training Effectiveness Data

The meaningfulness of any model is directly dependent upon the quality of its data. The results of a combat simulation are based upon the accuracy of input data. For many weapon systems, operational data or data obtained from rigorous field testing is simply not available or cannot be collected (e.g, a year 2005 threat armor vehicle equipped with a kinetic energy weapon system). Probability of hit and kill for these futuristic weapon systems must be approximated. Expert judgement is used to estimate these data. Analogously, the Training Mix Model depends upon data derived from expert judgement.

The U.S. Army Armor school was asked to identify candidate training devices and methods that could be used to develop a training strategy. Traditional techniques such as field training and live fire exercises, as well as the currently and soon-to-be available devices, such as the Close Combat Tactical Trainer (CCTT) were selected. Mission critical tasks were identified in the Army Training and Evaluation Program (ARTEP) Mission Training Plan (MTP) and gunnery manuals. These tasks form the backbone of the model and the data efforts.

After selecting the tasks, devices and methods, Subject Matter Experts (SMEs) specified the *training mode* for the device. A training mode is defined as, "How the device will be used to train soldiers". The training mode concept links the device with its use. In specifying a training mode, the SMEs considered the unit level, the mission trained (offense, defense, etc.) and the duration of one training session. The incremental contribution of one repetition of a training mode represents the atomic building block of the training effectiveness data. The model translates the selected training modes into training devices and number of required repetitions.

For each task, SMEs evaluated the efficacy of each training mode with respect to the number of iterations of training. For each task, the model assumes a continuum of possible proficiency levels ranging from 0 percent (no proficiency) to 100 percent (maximum proficiency). Seventy percent is defined to be standard proficiency.

For a given task, proficiency would be expected to decay until the task is trained by some TADSS. After training, the proficiency level rises to some level and begins to decay again until additional training is received. The rate of skill decay is a function of the characteristics of the task. The gain in proficiency after training is a function of both the task and the type of training. The training effectiveness data estimate the level of task proficiency that results from a repetition of a training mode.

Training Costs

A cost analysis for each method of training was conducted and these results were incorporated into the model. The purpose of the cost analyses was to determine TADSS Life Cycle Costs (LCC). The analysis was based on a variety of simulator cost data. The data consisted of Baseline Cost Estimates (BCE) and rough order-of-magnitude estimates which were developed at TRAC-WSMR with data furnished by the simulator Program Manager. In addition, TRADOC school personnel were consulted concerning key operational parameters. All cost data that were used in the analysis were converted to an FY94 dollar base year and adjusted for sunk costs. Military Pay and Allowances were removed from the LCC data. The most recent OPTEMPO cost inputs were obtained from the Army Cost and Economic Analysis Center's Force Cost Database. Current unit training budgeting levels are accounted for the payback analyses.

Based upon the LCC analysis, the model uses the cost data as coefficients in the objective function. The cost data represent one battalion's share of the cost of procuring and using the TADSS. The cost data consist of the program acquisition unit cost (PAUC), which includes both the procurement cost and the portion of the remaining R&D cost, and the training event cost, or operating cost, which is composed of device usage cost, ammunition expenditures (if any) and OPTEMPO miles (if any) that are consumed during one session of training.

Model Description

The Training Mix Model is formulated as a mixed integer linear programming model. The fundamental problem that the model solves is the selection of the training modes that satisfy all of the constraints at minimum cost. In particular, the selected training modes must meet or exceed the training standard value (70%) for all tasks. This means that all of the soldiers are trained to standard or better. The cost and training effectiveness data for the devices and tasks are consolidated in a series of equations. The selected training modes are translated into the training strategy. The objective function of the model is to choose the minimal cost TADSS that satisfy all of the constraints. The objective function consists of two terms; the first describes the operating expense of a device and the second term describes the procurement expense. Formally, the objective function is

$$Min\ Z = \sum_D \sum_L\ OperatingCost(D,L) * Sessions(L,D) * 2 + \sum_D\ \frac{PAUC(D) + RD(D)}{LCP}\ *\ Devices(D)$$

where D is the index for the device dimension and L is the index for the organizational level. Sessions(L,D) are decision variables that account for the number of training sessions for each device at each level. It is multiplied by the operating cost and by two to obtain an annual cost. The second term describes the unit's share of the procurement/research and development costs prorated over the life cycle period (LCP) of a device and is multiplied by the number of devices that are required.

Decision Variables & Constraints

In addition to limiting certain decision variables to integer cardinality, the Training Mix Model requires that certain decision variables be restricted to either a value of zero or one. These variables are used to select the training modes and the devices at each level. The majority of the decision variables in model fall in this class. In essence, the model selects the modes from a large binary space. Furthermore, the training mode variables form a specially ordered set of unit magnitude. For one of the dimensions, one and only one variable can assume a unit value, while all other values in the dimension are forced to zero. Once the training modes are chosen, they are translated into sessions and devices. These decision variables are subject to a series of upper and lower bound constraints. One of the two key constraints in the model is:

$$\forall L\ \forall T\ \sum_M \sum_R\ TE(L,T,M,R)\ *\ TM(L,M,R)\ \geq\ 0.70$$

where L is the level (battalion, company, platoon, crew) dimension, T is the task dimension, M is the mode dimension and R is the repetition dimension. These data are multiplied by the binary training mode TM(L,M,R) decision variables. The product of these variables must exceed 70% on each task at each level. The other key constraint is:

$$\forall L\ \forall M\ \sum_R\ TM(L,M,R) \leq 1$$

This constraint restricts the TM decision variable to a value of no more than one along the repetition dimension. This constraint models the special ordering of the set of binary variables. With these two constraints, the model must choose one of the 10 repetition modes such that the sum of the modes, when multiplied by the appropriate training effectiveness estimate meet or exceed the training standard (70%) for all tasks at all levels.

Model Environment

The General Algebraic Modeling System[4] (GAMS) software was used to construct the mixed integer linear programming model. GAMS writes the initial basis matrix and passes it to a solver. The integer programming solver that was used is the Linear and Mathematical Programming System (LAMPS)[5]. When LAMPS has produced a solution that is within user-specified tolerances, the basis matrix is returned to GAMS and postprocessed. GAMS has the facility to postprocess primal and dual decision variable values and has sophisticated report generating facilities. Currently, a SUN4 computing machine running UNIX SunOs 4.1.3, is used to solve the Training Mix Model. The example model consists of 677 equations and 567 decision variables (513 are discrete variables). The solution time for mixed integer linear programming problems are known to vary with the type and number of decision variables and associated data. The example model, due to the number of binary variables and model size requires approximately four hours of real clock computer time.

Example Results

The purpose of this section is to present the results of an application of the Training Mix Model. The focus of the model is a 1995 armor M1A1 platoon and crew sustainment training strategy. *These results do not reflect the actual training effectiveness values for the TADSS and are only provided to illustrate the capabilities of the model.* The training effectiveness data were generated by armor SMEs with the SIM2 methodology solely for the purpose of model development. The training methods that were considered in the model are given in tables 1 and 2. These training methods are a combination of present day training methods and proposed training methods. The SMEs described how the methods would be used to train maneuver and gunnery tasks. The SMEs rated the training effectiveness of each method for each task with the device effectiveness methodology.

For each training device, one or modes were defined to describe how the device would be used in the training strategy. For example, 12 different platoon CCTT training modes were defined. The tasks that the modes could train were based on perceived capabilities, logical groupings of related tasks, and how many tasks could be trained during a training session. Because the modes refer to way the device is used to train, the frame of reference for the use of the device was taken from the type of operations that the platoon and crew would need to perform. For example, some of the operational frameworks were defined as offense, defense, retrograde operations.

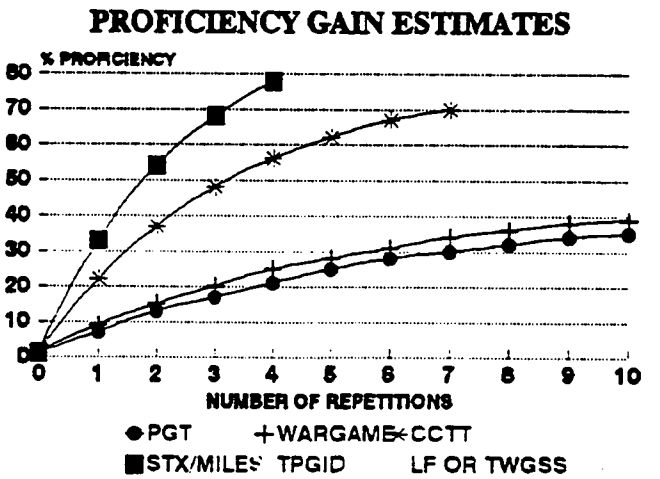


Figure 1 Proficiency Gain Estimates for a Platoon Task

The use of a device, say CCTT, would be combined with an operation, for example, the offense operation, to form a training mode. In this example, the training mode would consist of CCTT device being used to train offensive operations. A total of 34 platoon training modes were defined for 59 platoon tasks. A total of 15 crew training modes were defined for 33 crew tasks.

Figure 1 illustrates the proficiency gain estimates associated with the various ways of training the platoon task, "perform an attack by fire". For this task, the zero practice minimum was determined to be 0.01. In other words,

---

[4]Brooke, A., Kendrick, D., Meeraus, A. *GAMS: A User's Guide*, The Scientific Press, Redwood City, California, 1988

[5]Advanced Mathematical Software, Ltd., LAMPS USER GUIDE, Advanced Mathematical Software, Ltd, London, Great Britain, 1990.

if this task was not trained during a six month period, soldiers could not perform this task. LIVE FIRE and TWGSS were rated equally effective, so their curves overlap. Note that data for only one repetition if LIVE FIRE, TWGSS and TPGID modes are shown. This is due to the real-world constraint that those types of training can occur at most once during the training cycle - when the unit goes to its gunnery density training. In fact, the model contains a constraint which forces the selection of one and only one of these modes at the platoon and crew level. This accounts for the fact that the unit will conduct some kind of gunnery training prior to its crew and platoon gunner qualification events.

TABLE 1: PLATOON TRAINING DEVICES/MODES	
STX/MILES	Situational Training Exercise (STX) with Multiple Integrated Laser Engagement System (MILES) as the gunnery engagement simulator
LIVE FIRE	Table XI gunnery training: tanks on a firing range with a .50 caliber device
TPGID	Table XI gunnery training: tanks on a firing range with a 35mm caliber device (Tank Precision Gunnery Inbore Device)
TWGSS	Table XI gunnery training: tanks on a firing range with a laser device (Tank Weapons Gunnery Simulation System)
PGT	Table XI gunnery training using networked crew conduct of fire trainers (Platoon Gunnery Trainer)
CCTT	Successor to Simulation Networking (SIMNET). Close Combat Tactical Training vehicle simulators.
WARGAME	simulator training based on a system like JANUS
CLASSROOM	classroom training

TABLE 2: CREW TRAINING DEVICES/MODES	
LIVE FIRE	Tables VI and VII gunnery training: tank and full caliber ammunition on a firing range
TPGID	Tables VI and VII gunnery training: tanks on a firing range with a 35mm caliber device (Tank Precision Gunnery Inbore Device)
TWGSS	Tables VI and VII gunnery training: tanks on a firing range with a laser device (Tank Weapons Gunnery Simulation System)
COFT	home-station gunnery training using a simulator to train the vehicle commander and gunner (Conduct of Fire Trainer)
CLASSROOM	classroom training

Demonstration Training Strategy

Table 3 presents a summary of the training strategy solution. *The solution does not reflect the state of the world and should only be used to assess the state of model development and its potential capabilities.* The solution is not necessarily globally optimal. In order to prove that a solution is globally optimal, the state space must be exhausted. A user-defined tolerance factor was included in the model to accept a solution that is "close" to optimal. Without the tolerance factor, an unacceptably large of amount of computer time would be required to obtain a solution. The columns of table 3 refer to the model base case and four parametric excursions.

TABLE 3: NUMBER OF REPETITIONS BY DEVICE & LEVEL (from experimental data - Six Month Training Cycle)					
	BASE CASE AND PARAMETRIC EXCURSIONS				
PLATOON LEVEL *MODE*	**BASE CASE**	**NO CCTT**	**NO TPGID**	**+10 pts CCTT**	**-10 pts CCTT**
STX/MILES	9	14	9	7	8
CCTT	14	-	14	10	23
LIVE FIRE	0	0	1	0	0
TPGID	1	0	-	0	0
TWGSS	0	1	0	1	1
PGT	0	0	0	0	0
CLASSROOM	13	10	13	13	11
WARGAME	4	16	4	2	3
CREW LEVEL *MODE*	**BASE CASE**	**NO CCTT**	**NO TPGID**	**+10 pts CCTT**	**-10 pts CCTT**
LIVE FIRE	1	1	1	1	1
TPGID	0	0	-	0	0
TWGSS	0	0	0	0	0
COFT	45	45	45	43	45
CLASSROOM	10	10	10	11	10

Table 4 shows the annual cost and OPTEMPO expenditures for the base case and four parametric excursions. The amount of OPTEMPO is multiplied by the OPTEMPO cost factor and included in the annual training cost.

TABLE 4: ANNUAL COST and OPTEMPO REQUIREMENTS (from experimental data)					
PARAMETRIC EXCURSION	**BASE CASE**	**NO CCTT**	**NO TPGID**	**+10 pts CCTT**	**-10 pts CCTT**
ANNUAL OPTEMPO	195	258	195	170	183
ANNUAL COST ($M)	4.3	4.4	4.2	4.0	4.2

DISCUSSION

The column labeled "Base Case" in table 3 contains the number of repetitions (reps) of training on each device that are required to meet training standard during a six month training cycle. The platoon level training strategy consists of 9 STX/MILES repetitions. There was a total of 13 STX/MILES modes; of the 13 STX/MILES modes, the model selected 3 reps of 1 mode, 2 reps of 2 other modes and 1 repetition of 2 other modes. The model selected 14 reps of CCTT training and chose TPGID for platoon gunnery training in preparation for qualification. Thirteen CLASSROOM and 4 WARGAME reps were included in the solution.

The crew level training strategy consists of 45 COFT reps, 10 reps of CLASSROOM training and LIVE FIRE. LIVE FIRE (Tables VI and VII) was selected as preparatory training for qualification. This alternative is much more expensive than the TPGID alternative, which was already selected for platoon training. The reason the model chose LIVE FIRE is that this alternative trained some gunnery tasks to standard which TPGID could not. The requirement that all tasks be trained to standard drove the model to select a more expensive, but more capable method of training. The annual OPTEMPO requirement is 195 miles per tank and the annual cost is $4.3 million. The annual cost includes the battalion's prorated share of the acquisition cost of the developmental components of the strategy (CCTT, TPGID, WARGAME).

No CCTT Excursion: The NO CCTT excursion removed CCTT from the set of available training devices. Because CCTT is a platoon training simulator and not used to train crew level tasks, the crew training strategy was not affected by the elimination of CCTT. The platoon training strategy is sensitive to the elimination of CCTT. The model selected 14 reps of STX/MILES as opposed to 9 reps in the base case. It also selected somewhat less CLASSROOM training, but significantly more WARGAME training. TWGSS was selected as the gunnery preparation trainer instead of TPGID as in the base case. This strategy requires 258 annual OPTEMPO miles per tank - 63 more miles per tank than the base case. Interestingly, a independent payback analysis of the CCTT life cycle cost found that each tank would have to tradeoff 61 miles of its annual OPTEMPO allocation to pay for CCTT.[6] The model results are consistent with the payback analysis. The annual cost of training rose only slightly, since the battalion does not have to buy its portion of CCTT, but must expend more for OPTEMPO.

No TPGID Excursion: The NO TPGID excursion removed TPGID from consideration. The results are identical to the base case except that LIVE FIRE is substituted for TPGID. The OPTEMPO miles expenditure is the same as in the base case, but the solution is slightly less expensive than the base case. The reason that the solution is less expensive is because platoon LIVE FIRE training with the .50 subcaliber device is less costly ($108K) than training with the TPGID device ($208K). When TPGID was eliminated, the model selected the LIVE FIRE mode.

Increase CCTT Effectiveness Excursion: The next excursion increased the effectiveness of CCTT by adding 10 points to all tasks that are trainable by a CCTT mode. The purpose of increasing the effectiveness of CCTT by 10 points was to consider the effect of the SMEs' having underestimated the effectiveness of CCTT. The results included a reduction from the base case of 2 STX/MILES reps, 4 CCTT reps, and 2 WARGAME reps. Again, TWGSS was selected as the gunnery preparation trainer instead of TPGID. There was a slight modification in the number of crew COFT and CLASSROOM reps that can be attributed to the tolerance factor. The number of annual OPTEMPO miles was reduced by 25 miles per tank. The cost of this excursion is $300K less than the base case. The savings are obtained from a decrease in the OPTEMPO miles associated with the STX/MILES modes, the reduction in CCTT usage cost and the lower cost of TWGSS training.

Decrease CCTT Effectiveness Excursion: The last excursion decreased the effectiveness of CCTT by 10 points for all tasks that are trainable by a CCTT mode. This excursion considered the effect of the SMEs' having possibly overestimated the effectiveness of CCTT. The original CCTT estimates were decremented by 10 points for this excursion. The model selected 9 more reps of CCTT as compared to the base case, and 8 reps of STX/MILES (one rep less than the base case). TWGSS was selected as the gunnery preparation trainer. The number CLASSROOM reps decreased by 2 from the base case to a total of 11 CLASSROOM reps. The number of WAR GAME reps was reduced to 3 (one less than the base case).

Conclusions

The base case training strategy demonstrates that the model is capable of generating a solution that consists of a mix of TADSS, field and gunnery training. The solution is guaranteed to train all of the soldiers to standard on all of the tasks at the lowest possible cost. The parametric excursions show that the model is sensitive to input data and can provide a range of solutions based upon a variety of conditions under which training may occur.

The methodology is the result of a number of complex concepts that have been integrated into an innovative tool and provides the Training Developments community with an important advance in the evaluation of current, new and proposed training systems. It represents the combined efforts of a number of military analysts, operations research analysts, engineering psychologists and training analysts from several Army analysis agencies and the TRADOC schools.

The model offers the training developments community an opportunity to conduct analyses of the potential worth of a proposed training system in analogous way to that of the combat developments community. The

---

[6]Johnson, D. R., Miller, D.V., Cox, J., *SIM2: Cost Analysis Update*, Department of the Army, Training and Doctrine Analysis Command - White Sands Missile Range, White Sands, New Mexico, May 1993.

model can be used to determine effective, less costly combinations of training modes that will meet specific training requirements under various conditions. The model can identify training deficiencies in a particular trainer or strategy. The model can be used to conduct many "what if" excursions to obtain a clearer picture of the expected utility of a trainer. The model could be used at the early stages of the training device development process by providing analytical documentation for regulatory requirements. In effect, the model could be part of a TADSS cost and operational effectiveness analysis. The model could also be used to develop generic and specific training strategies to help military services meet their mission requirements within resource constraints.

Virtual Environment And Computer-Aided Technologies Used For System Prototyping And Requirements Development

Cory Logan, James Maida
NASA, Lyndon B. Johnson Space Center
2101 Nasa Rd. 1, mail Code SP-52,
Houston, Texas 77058
logan@ssmtf3.jsc.nasa.gov, maida@graf6.jsc.nasa.gov

Michael Goldsby
Lockheed Engineering and Sciences Company
2400 NASA Rd. 1, mail code C-44
Houston, Texas 77058
goldsby@graf10.jsc.nasa.gov

Jim Clark, Liew Wu
Johnson Engineering Company
1290 Hercules #201, mail code SP-52
Houston, Texas 77058

Henk Prenger
Barrios Technology Inc.
1331 Gemini Blvd., mail code CA-3
Houston, Texas 77058

ABSTRACT

The Space Station Freedom (SSF) Data Management System (DMS) consists of distributed hardware and software which monitor and control the many onboard systems. Virtual environment and off-the-shelf computer technologies can be used at critical points in project development to aid in objectives and requirements development.

Geometric models (images) coupled with off-the-shelf hardware and software technologies were used in The Space Station Mockup and Trainer Facility (SSMTF) Crew Operational Assessment Project. Rapid prototyping is shown to be a valuable tool for operational procedure and system hardware and software requirements development.

The project objectives, hardware and software technologies used, data gained, current activities, future development and training objectives shall be discussed. The importance of defining prototyping objectives and staying focused while maintaining schedules are discussed along with project pitfalls.

1.0 INTRODUCTION

The Crew Task Assessment project started in the spring of 1991 as a proposal to the Flight Crew Support Division (FCSD) for a UNIX based computer system for the Space Station Mockup and Trainer Facility. This facility (see figure 1.0) is used for Space Station Freedom design development and for instructor and crew training. The prime objective is to construct an Operator Interface System that can be used to develop and evaluate SSF user interfaces. The secondary objective is to demonstrate the human-computer interface of the SSF Data Management System and use the system for instructor and crew training.

A three monitor UNIX X Windows workstation with SSF user interface viewing capabilities was completed in the summer of 1991(see figure 1.1). In the fall of 1991 sensor and transducer control was added to the system with off-the-shelf programmable logic controllers and in the early summer of 1992 data, video and control connections to the Graphics Analysis Facility (GRAF) were added. In the fall of 1992 robotic hand controllers where added. The system presently has SSF onboard display viewing, SSF and Space Shuttle camera images and control, and SSF and space shuttle robotic image and control capabilities.

As our knowledge increased and equipment came on line, the objective expanded to rapid prototyping. Today our objective is to have hardware, software, graphics, and interface capabilities to do rapid (less than 1 month) prototyping for SSF DMS related development. These computer and graphics capabilities coupled with the Mockups and Trainers give us the tools to assist in design, writing requirements and specifications and verification of man-machine interfaces.

2.0 SETTING OBJECTIVES

Our main objective was to choose hardware and software which could be used to emulate flight systems early in the program to assist requirements development. This system would be used to give future users a chance to evaluate and comment on the proposed flight systems in time for changes. The hardware and software is off-the-shelf and required no video drivers or kernel programming. For each project or task the customers, users and stake holders are defined. The success of the task is also defined. The original startup project success definition was:

> A UNIX based computer system in the SSMTF which will run SSF onboard displays in a flight like environment. These displays shall have process control and monitoring capabilities. The system shall provide positive crew and user training. The system shall be built with off-the-shelf components and software. The system shall be in operation by the summer of 1992.

The original customer definitions were:

USER/CUSTOMER	TASK
Space Station Reconfiguration Office	display reviews and requirements development
Crew Office	crew task assessments.
Human Computer Interface Lab	display standards development and review
Mission Operations	training development.

Today the objectives have expanded to address the following:

- Direct Views: of assembly and maintenance tasks by IVA crew
- External Lighting: for direct and video views
- Visual Cues: alignment aids and targets
- Displays: procedures, data, video views, communications, graphics needed to perform tasks at the workstation
- Workstation Design: anthropometry of workstation and hardware switches, controls arrangement, adequacy of task lighting
- Functionality and operability of robotic controls

3.0 EXAMPLE EVALUATIONS

Following are two examples of ongoing evaluations being performed in the facility. Over five evaluations have been held to help define workstation and flight display requirements.

3.1 CREW OPERATIONAL ASSESSMENT OF SSF WORKSTATIONS

A Crew Operation Assessment of Space Station Freedom Workstations was performed in February 1993. The objective of this assessment was to define the off-screen requirements for camera and robotic controls. The test scenario was a robotic operation, as planned for Utilization Flight 1 (UF-1). The operations involved controlling the Space Station Remote Manipulator System (RMS) in manual mode, automated mode and single joint mode. Grappling, maneuvering and berthing actions were accomplished, simulating Cryo Carrier Installation tasks. The Command and Control (C&C) workstation (see figure 1.1) in Node 1 was used to simulate operations in a Node or Cupola. All displays were Space Station prototypes. They included several shuttle arm displays, Communication & Tracking, Camera Configuration and Camera Control displays, and text procedure viewers. Figures 3.0 and 3.1 show typical display scenarios.

Thirteen cameras were available for evaluators to use throughout the scenario. Manual control panels were mounted on the workstation rack front face. Thirteen camera views were programmed at the GRAF lab. These views had pan, tilt, and zoom capabilities with keyboard, on-screen (computer mouse) or off-screen (key board or manual panel) control and selection. Split screen capabilities (two views sharing one NTSC window) could also be selected. Arm motion was simulated using predefined positions selected from the keyboard because the hand controllers were not operational at the time. Off-screen and on-screen control of the space shuttle arm brakes, auto sequence controls, and single joint controls were also available.

The operational scenarios evaluated contributed to the recommendations made for workstation features and capabilities. Figure 3.2 gives the overall view of the SSF in the Man Tended Configuration (MTC). It shows some of the camera positions used in camera viewing and control evaluations. Figures 3.2 through 3.10 show Shuttle and Space Station camera views, as seen at the workstation, during robotic arm operations. Figure 3.4 shows a Space Station truss camera view of the stowed shuttle and Space Station arms. Figures 3.5 through 3.8 show the shuttle and Station views required to grapple a payload in the Shuttle payload bay and move it to the Space Station arm. Figures 3.9 and 3.10 show the payload hand off using shuttle payload camera C and the Node camera. Users selected and controlled the 13 cameras to decide the best viewing for grapple, payload hand-off, and docking.

The results of the evaluation are detailed in "CREW OPERATIONAL ASSESSMENT OF SSF WORKSTATION" distributed by the Station-Exploration Support Office in April 1993.

3.2 DISPLAYS AND CONTROLS MODE TEAM CREW EVALUATIONS

The Displays and Controls (D&C) Mode Team's primary responsibility is to develop and document in the D&C Flight System Software Requirements (FSSR) the requirements for on-board displays. The SSMTF facility is being used to develop and evaluate displays. The virtual environment created using the GRAF camera images has contributed to the development of flight menus for camera control and on and off screen controls. The simulated robotics displays and images coupled with functional hand controllers and camera views can be used to develop flight robotic displays and controls, and for training.

Figure 3.11 is the Main Menu of the flight user interface. The Communication and Tracking (C&T) display tree takes you to the Video Configuration display figure 3.12. This display is used to power cameras up and down and to make initial camera to monitor connections. The display used for camera control is shown in figure 3.13. When the camera control display is opened the keyboard gains camera control functions. Camera images can be panned, tilted, and zoomed from the arrow keys on the keyboard. Camera selection can also be made using the keyboard. The camera control displays coupled with the Shuttle arm and Space Station arm displays, figures 3.15 and 3.16, are used to control the robotic arms in a virtual environment in the SSMTF. The robotic control and camera

selection and control commands come from the SSMTF computers and interfaces. This control information drives the GRAF image generation. The images are converted from RGB to NTSC format and sent to the SSMTF for display.

4.0 SYSTEM SOFTWARE

4.1 SPACE STATION MOCKUP AND TRAINER FACILITY

The User Support Environment (USE) prototyping activities during 1990 identified Kinesix Corporation's "SAMMI" product as the display build and display execute product for the SSF onboard displays. SAMMI is a Graphical User Environment for building X Window based user interfaces for managing networked information systems. The networked systems in the SSMTF project include:

Command and Control Workstation
Cupola Workstation
Development workstation in the computer lab
Programmable logic controller
PC in lab
Silicon Graphics machines in the Graphics Analysis Facility

4.1.2 COMMUNICATIONS

The Workstation to GRAF software is shown in figure 4.0. Remote Procedure Calls (RPCs) are used between the SAMMI run time display code and applications and/or data bases run locally or on a remote machine/s.

Ethernet and TCP/IP is used to connect all computer based CPUs. Control panels and hand controllers are connected to programmable logic controllers (PLCs). These controllers have plug and go input/output boards for interfacing control wiring to a CPU.

The programmable controller uses a "remote I/O protocol" to communicate ladder logic program information to the PC. PLC I/O data tables are mirror imaged on the PC. Using FTP INC. and Allen-Bradley libraries a C program transfers the PLC information to the UNIX workstations, or vice versa. The UNIX data base is a mirror image of the PC data base.

Workstation displays interact with a data base running locally, or remotely, using an application program interface (API) and Remote Procedure Calls (RPC). Examples of application interfaces running in the SSMTF which provide display to data base connectivity are:

Camera control API	peer to peer: display to PLC and display to GRAF control information
Robotic control API	peer to peer: display to robotics model model to PLC, model to GRAF
dp4ndbm	client-server: polled data base

Figure 4.1 shows the data flow from the three computer environments: SUN, Silicon Graphics machine in the GRAF, and the programmable logic controller with personal computer.

4.1.3 HAND CONTROLLERS

Robotic hand controllers are connected directly to the PLC analog input ports. The PLC sends a digital number to the PC data base for each of the six hand controller motion commands and three hand grip switches. The data base of motion and switch status commands, located on the PC, is read by the workstation robotic model. The

robotic model process sends position and status information to the workstation data base which streams the information to the display.

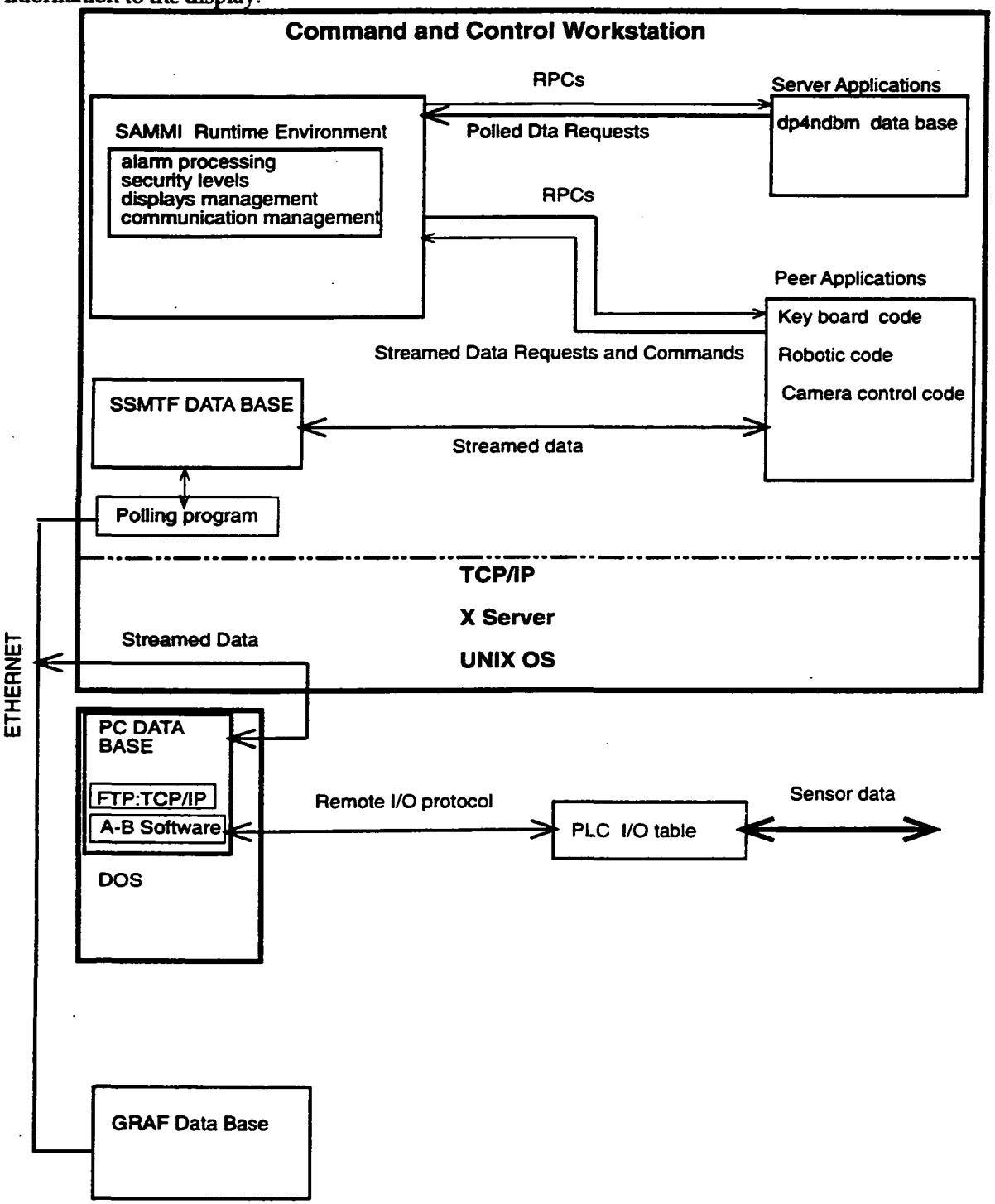


Figure 4.0 SSMTF Software Overview

Fig. 4.1 = 366p

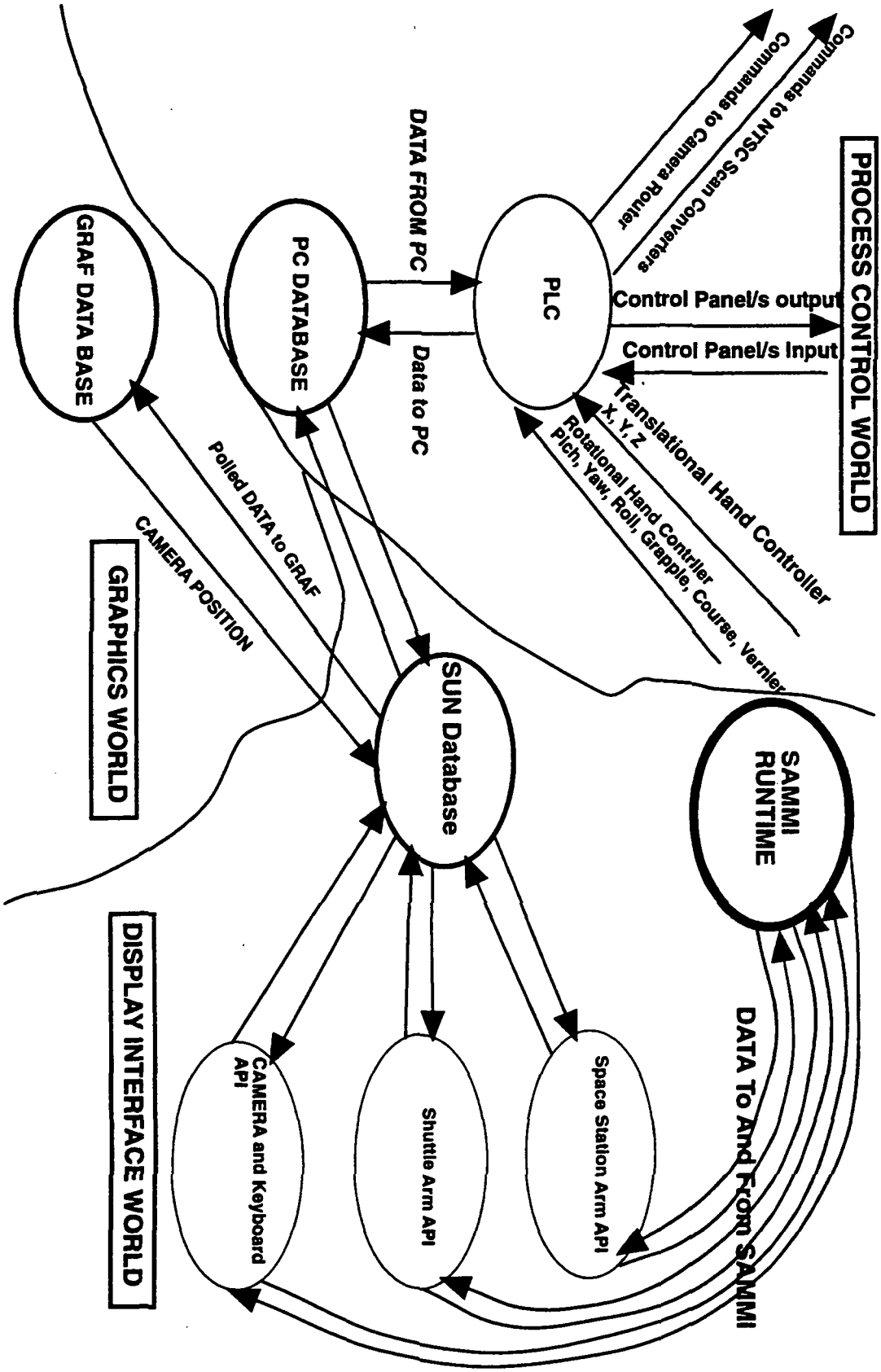


Figure 4.1 SSMTF Software Data Flow

The joint angles calculated in the robotic model are sent to the GRAF for robotic graphic model manipulation. It should be noted that a polling program, running on the UNIX machine with the data base, checks for changes in the data base before sending the data file to the GRAF. This reduces network traffic.

4.1.4 CAMERA CONTROL

Camera control and selection can be done using the SAMMI display, keyboard, or manual control panel. Command information is sent to the GRAF from the data base running on the workstation. The difference is how the data base is manipulated. The hard control panel communicates through the PLC, the keyboard through an X-window server based API, andthe SAMMI display through a SAMMI based API.

4.1.5 NTSC SCANNER CONTROL AND CAMERA ROUTING CONTROL

The PLC has an ASCII Basic card which outputs RS 485 commands given input from the PLC I/O table. This RS 485 command is used to control the camera router and NTSC scan converters.

4.2 GRAPHICS ANALYSIS FACILITY

The synthetic camera views shown on workstation monitors in the mock-ups are produced on two Silicon Graphics Reality Engines in the Graphics Analysis Facility (GRAF) in Building 15 and conveyed to SSMTF in Building 9 via coaxial cable. So far, each SGI workstation has been used to produce the images for one monitor, but special video-splitter hardware installed in the workstations enables each Reality Engine to produce images for up to three monitors.

The software which produces the camera views consists of two programs, one of which runs on the SGI workstation and the other on the Sun workstation used for the Sammi system. The program on the Sun treats the Sammi files as read-only and polls them every tenth of a second. It extracts camera commands from the PLC records and shuttle arm commands from the Sammi data base. If any of the camera or robotic arm control variables have changed since they were last polled, a command record containing the current data is sent to the SGI workstation via the Internet. Traffic on the network is minimized by transmitting only when a variable has changed. Except for small acknowledgment records sent back to the Sun, all information flow is one-way. The communication is carried out via Berkeley sockets using the TCP/IP protocol.

Two copies of the polling program run on the Sun workstation, one to communicate with each SGI workstation. Whenever a change occurs in a relevant data base or PLC variable, each polling program detects it and sends the appropriate commands to its SGI workstation. Thus, even though the programs running in the two SGI workstations do not communication with one another, their representations of the camera and robotic arm positions remain identical because they both see the same sequence of commands.

The program running on the SGI workstation consists of two processes, one to handle the communication and the other to draw the images. Using a separate process to handle communication guarantees that no command record is ever lost, even if it is sent while an image is being drawn. It is possible for several command records to arrive while an image is being drawn; the effects of the commands are accumulated until they can be carried out by the drawing process.

The drawing process runs the GRAF's in-house display program, DMC, augmented with a special interface to allow it to interpret the commands received over the network. The interface reads the accumulated network commands and turns them into DMC commands, which it issues to DMC. If any command could have caused a change in the image, the image is redrawn. In order to handle the split-screen feature of a simulated monitor, the display is divided into two viewports; if only one viewport was changed by the commands, only that viewport is redrawn. (Likewise, when one workstation is used to simulate multiple monitors, the display is divided into multiple viewports, and only the affected viewports are redrawn.) Software clipping and polygon meshing were added to DMC to increase its drawing rate for this application.

Redrawing is not begun until a half-second's worth of camera motion has been accumulated, so that a redraw is not wasted on an undetectable amount of motion. (If the motion command is terminated before a half-second, it is performed immediately.) While image-changing actions are in progress, the display is redrawn as often as possible. The redraw time causes camera and arm motion to appear to occur in increments rather than smoothly. However, the total affect of the motion is accurate, as if it had been performed continuously.

The two processes in the SGI program are implemented as so-called light-weight processes, sharing memory space. Their access to shared variables (the accumulated commands) is controlled by means of semaphores.

5.0 SYSTEM HARDWARE

The major hardware components in the SSMTF are shown in figure 5.0.
They are:

Three monitor UNIX 486 PC Cupola Workstation
Three monitor UNIX 486 PC Node Command and Control Workstation (CCWS)
Three monitor SUN Lab Workstation
386 PC with Ethernet card and Allen Bradley programmable logic controller (PLC) scanner card.
Programmable Logic Controller (PLC) with various I/O boards and ASCII basic Card
Hand controllers
keyboard, track ball
manual camera control and robotics control panels
NTSC to RGB scan converters
RS-485 to RS-232 smart node boxes.
video router and cameras

5.1 CONTROL SIGNALS TO UNIX

The software available off-the-shelf drove the hardware purchases. The requirement most difficult to satisfy was the I/O to UNIX interface. A solution was found using a PC, an Allen Bradley scanner card, and an ethernet card. (PLCs are now being made with a direct ethernet connection and RPC/streams development software giving a more elegant solution). The PLC, PC, and UNIX workstation have communication connections. After the I/O to UNIX solution was found application interfaces were written between SAMMI and the PLC. Rapid prototyping of candidate control panels is achieved using PLC, I/O boards, scanner card, PC, and ethernet. In the latest review camera control and selection information from a manual panel was sent to the GRAF lab and images manipulated using the panel input.

5.2 THREE HEADED X TERMINALS USING PCs

The three monitor workstation is a three head 386 PC running UNIX and a three screen X-server. The PC was used because of its low cost. The Command and Control Workstation PC in the mockups is acting as a three monitor X terminal. The SAMMI displays, APIs, robotic model etc. are all running in the LAB.

5.3 NTSC VIDEO ON SVGA MONITORS

The second most difficult task was getting NTSC video in a movable, resizable, window, controlled by RS232, on a SVGA RGB monitor driven by a 386 CPU running UNIX and X. It was done using RGB Spectrum model 2050 boxes. The GRAF NTSC video is sent across site to the video router and the RGB 2050 boxes. The RGB 2050 boxes are controlled using RS232 ports, RS485 addressable boxes, and the ASCII basic card in the A-B PLC. SAMMI communicates to the data base, the PLC reads the data base and the ASCII BASIC card sends out a command frame over the RS485 network. The correct address acknowledges the command and acts. The RS 485 network is used to emulate the flight 1553 contol because of its low cost.

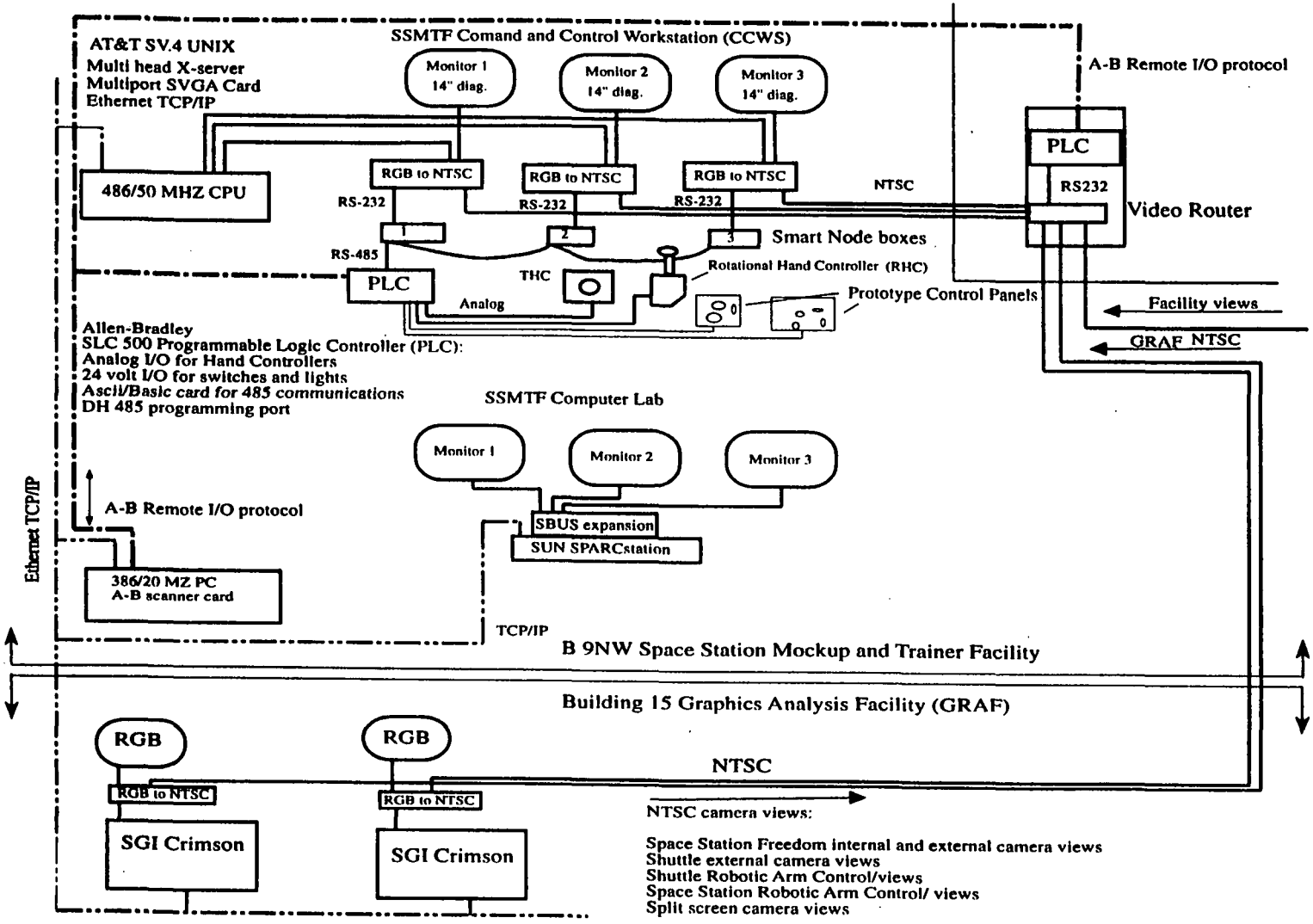


Figure 5.0 SSMTF HARDWARE OVERVIEW

5.4 CAMERA CONTROL AND SELECTION USING GRAPHICS

The camera selection is done on-screen using the camera control display push buttons (see figure 3.13). The peer to peer API between SAMMI and the PLC data base reads the button press event and a corresponding bit is set in the PLC data base. The polling program on the UNIX machine sends the data to the GRAF Silicon Graphics machine and the image is redrawn from the new camera's point of view. The pan, tilt, zoom functions act in much the same way with the keyboard key press and key release events being sent to the GRAF. A manual off screen control box can also be used to select and control cameras. One of the buttons on the manual control box sets a bit in the PLC data base. This set bit tells the GRAF lab that the of-screen control is activated. All camera selection and control commands are read from the off screen control box.

5.5 ROBOTICS USING GRAPHICS

In simple terms, shuttle arm hand controllers are connected to A-B analog cards. These cards feed a digital number to the PLC and UNIX math model. Software converts this delta position information to joint angles. These angles are read by the GRAF Lab and graphics are manipulated using the joint angle information.

6.0 LESSONS LEARNED

In the short time the system has had robotics and camera capabilities a few conclusions can safely be drawn:

1. Virtual environments can aid requirements development for man-machine and human computer interfaces.

2. Products like SAMMI are excellent tools for prototyping control panels/interfaces quickly. Users can test designs economically and with flexibility.

3. Mockup environments can be supplemented with virtual environments and computer technologies. Combining facilities and talents can give additional capabilities with improved asset utilization.

4. We learned that given scenarios such as the SSF camera control:

- Possibly 20 or so cameras each with pan, tilt and zoom, power on/off, available/not available.
- Three possible workstation destinations with three monitors each.
- Three monitors at each workstation, two with split screen capabilities.
- Camera selection and control capabilities from the computer screen and/or the key board and/or the manual control panel.

A quick working prototype will save many hours of writing requirements and meetings.
Users do not know 100% of their requirements. Give them something to help them decide.

5. In the computer world what you spend may not be proportional to the value of what you get.

6. If you build it they will come.

7.0 FUTURE PLANS

Voice control tests and more extensive display and camera control tests are being scheduled for the near future.

Real time UNIX software is being evaluated and multi headed X-servers developed.

The current capabilities give active hand controllers and screen refresh rates of approximately 1 second. Its hoped that Space Station software and hardware development groups will use the facility to test system and subsystem software user interfaces and control logic.

Facilities to measure CPU usage (Xstones, Whetstones, network impact) are being added.

As development engineering slows the facility will be used for crew training. All hardware was purchased with dual use in mind. The PC's and UNIX machines can be used in an office environment. The PLCs can be used to control the trainer mockup systems. The video equipment will be used in instructors stations.

As/if the Space Station is redesigned the GRAF facility can send the Space Station Mockup and Trainer Facility new images and evaluations can continue.

Figure 1.0 Mockup and Trainer Section

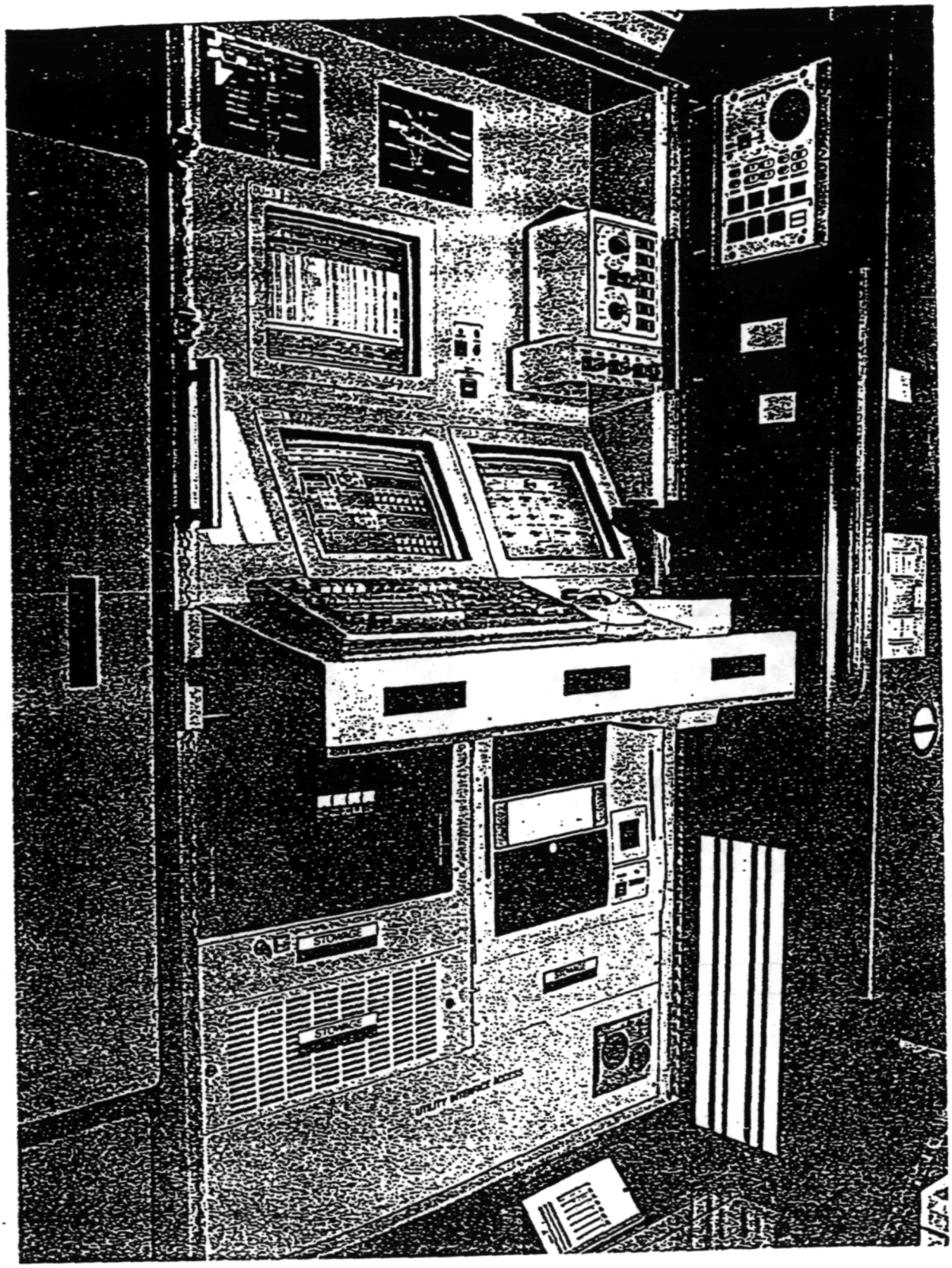
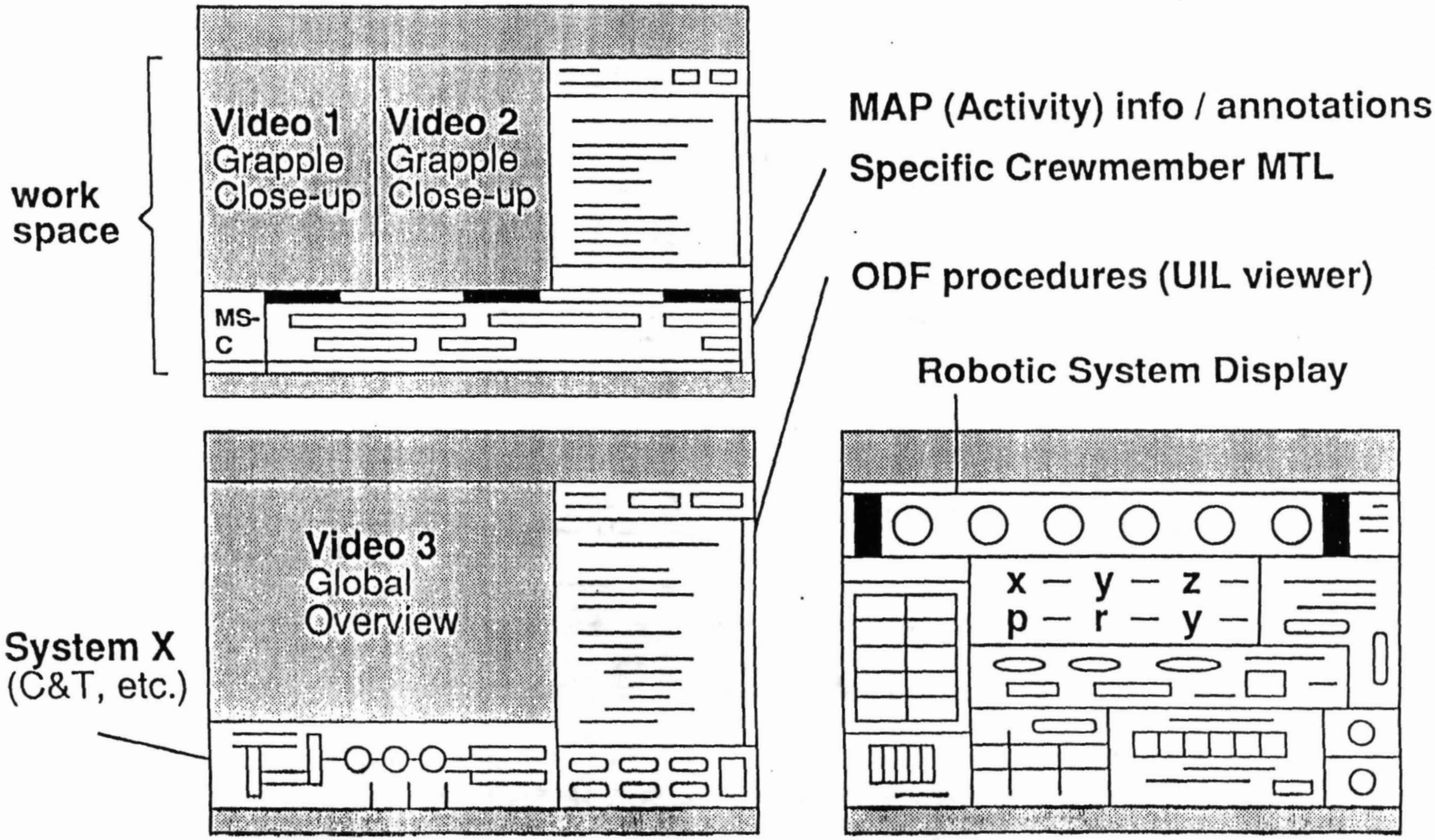


Figure 1.1 Mockup of Command and Control Workstation

ROBOTIC OPERATIONS
at U.S. workstation (3X14" Monitors)

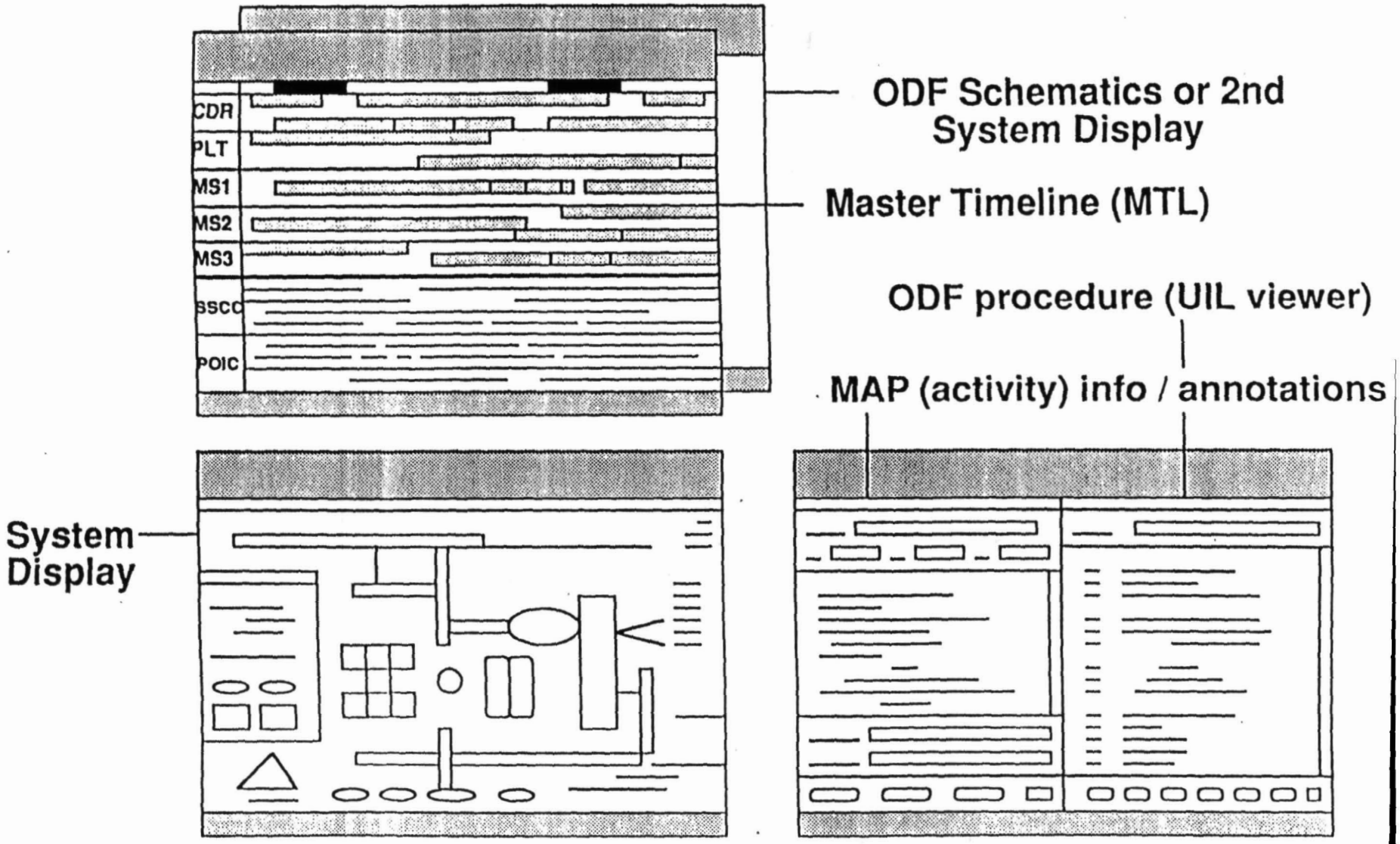


Int'l CSR, December, 1992

FCOD/CA3, Henk Prenger

Figure 3.0 Robotic Operations
374

Figure 3.1 System Operations



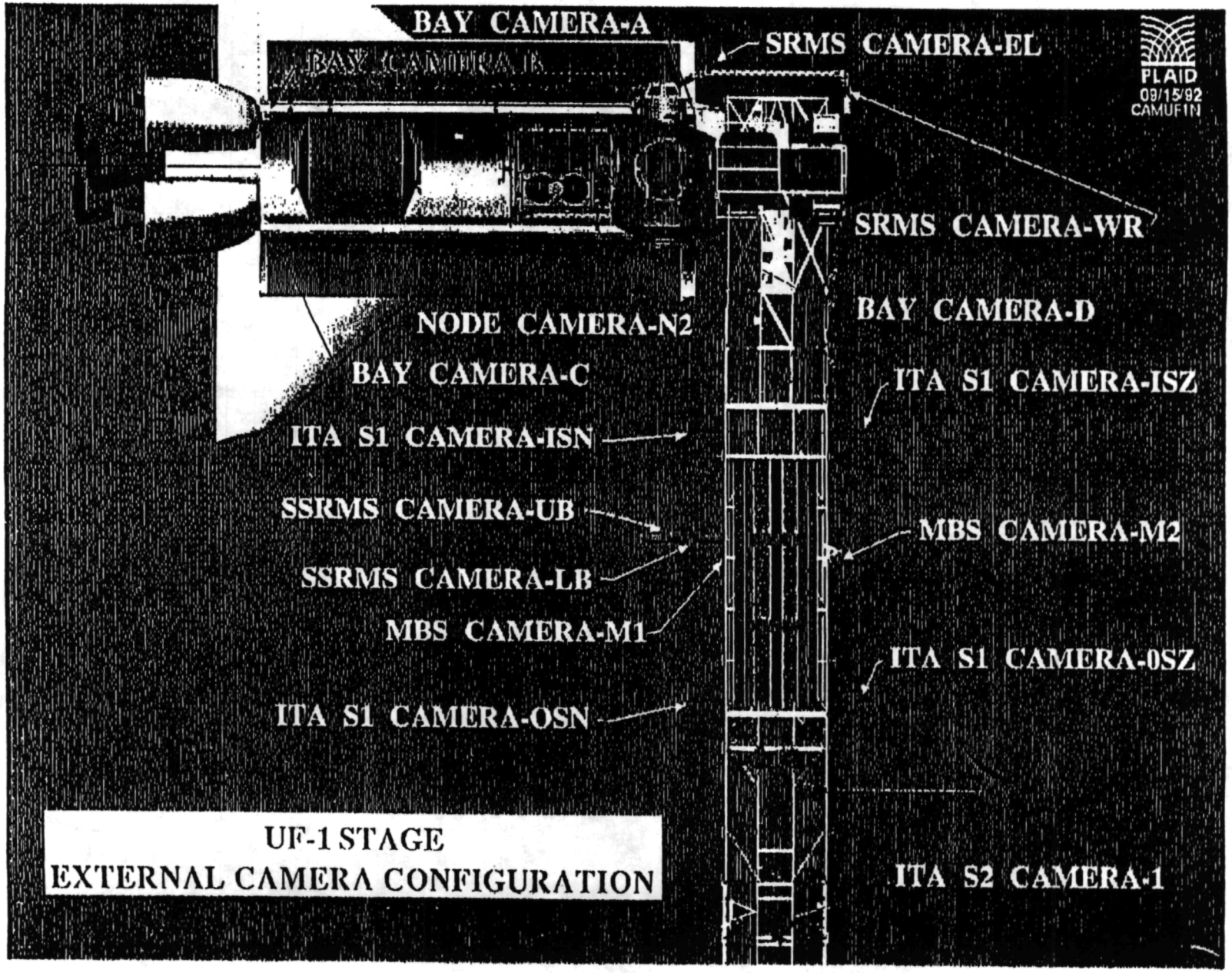


Figure 3.2 Space Station, Man Tended Capability

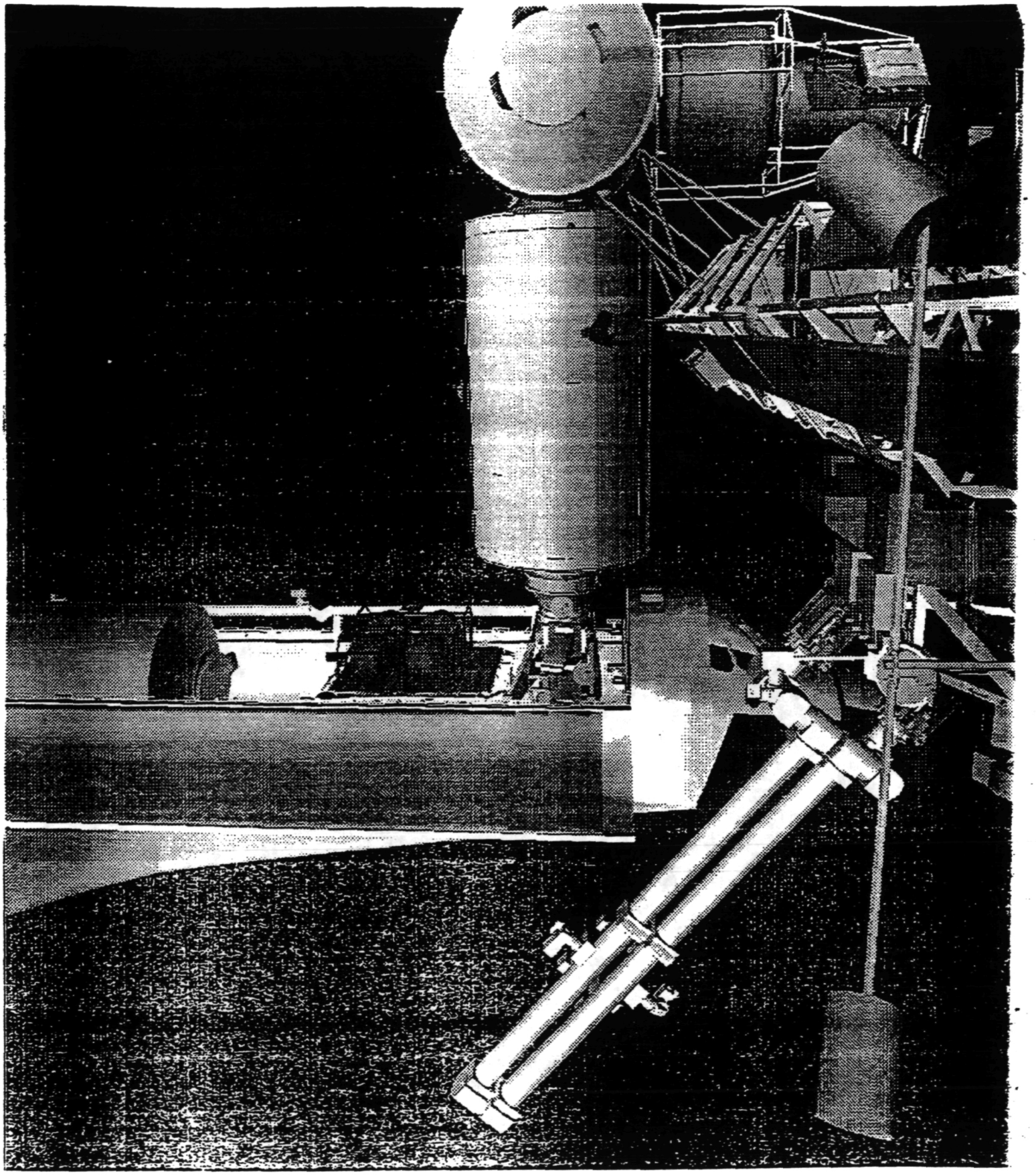


Figure 3.4 Space Station Outer Truss Camera View

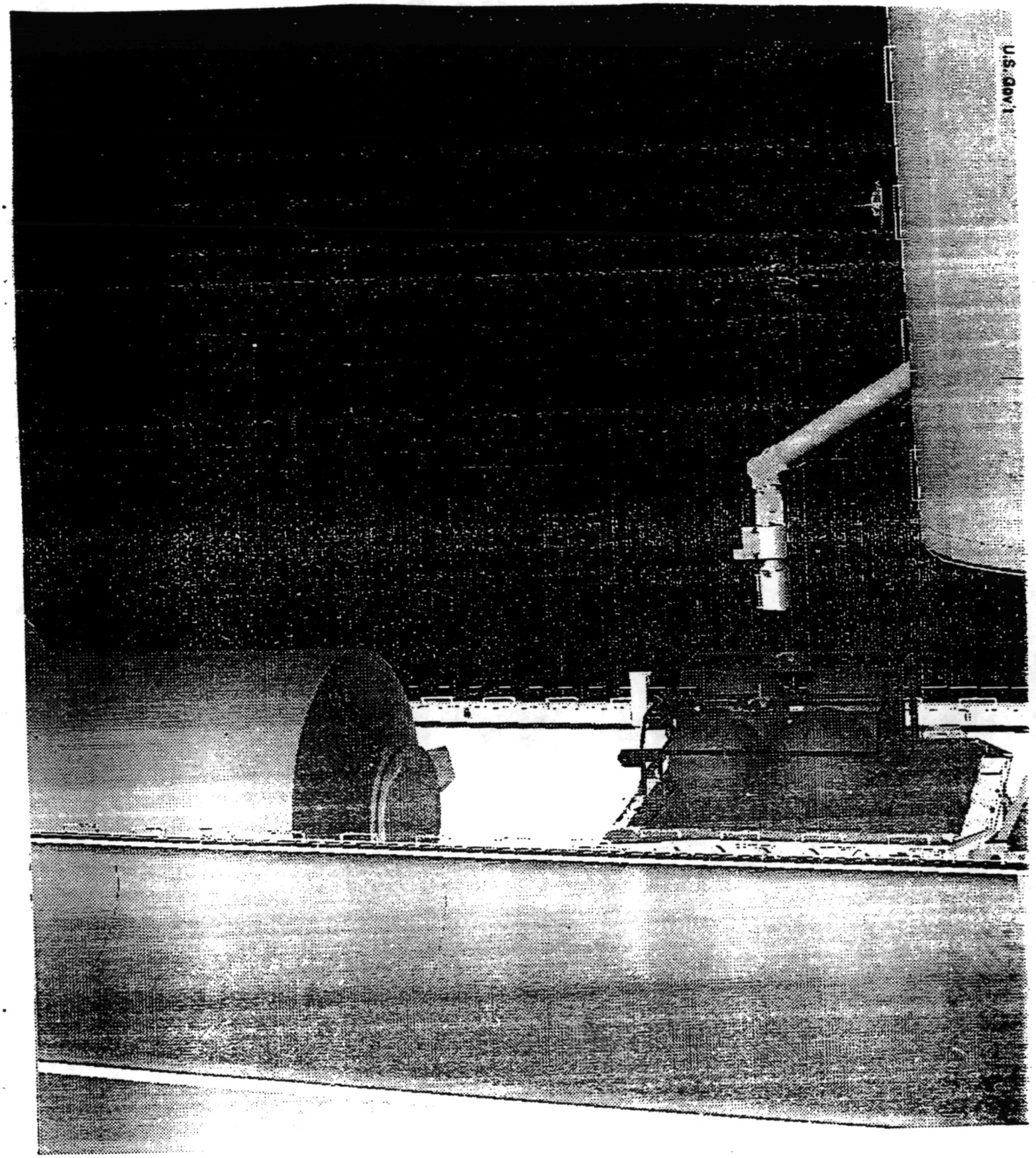


Figure 3.5 Space Station Inner Truss Camera View

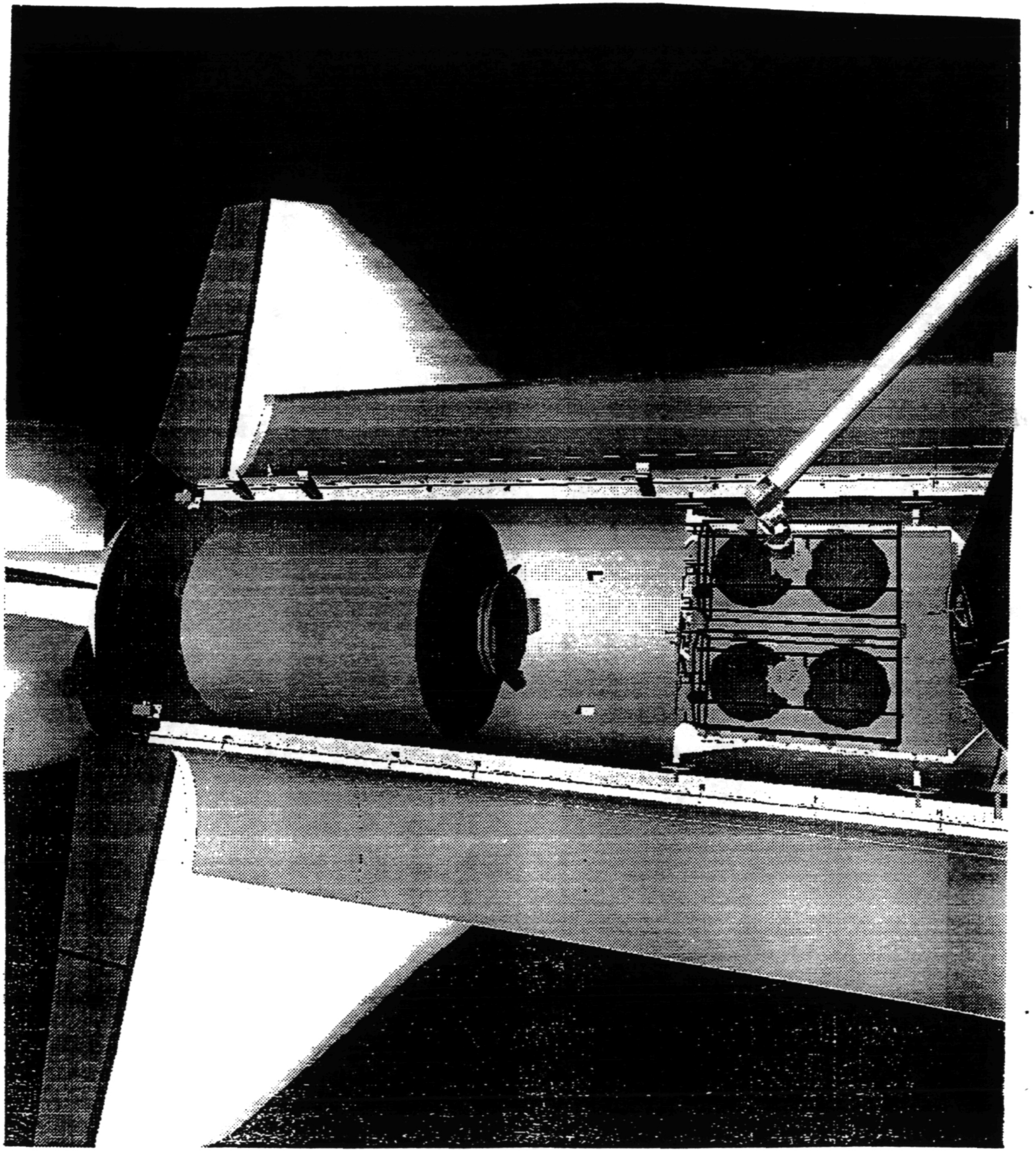


Figure 3.6 Node Camera View

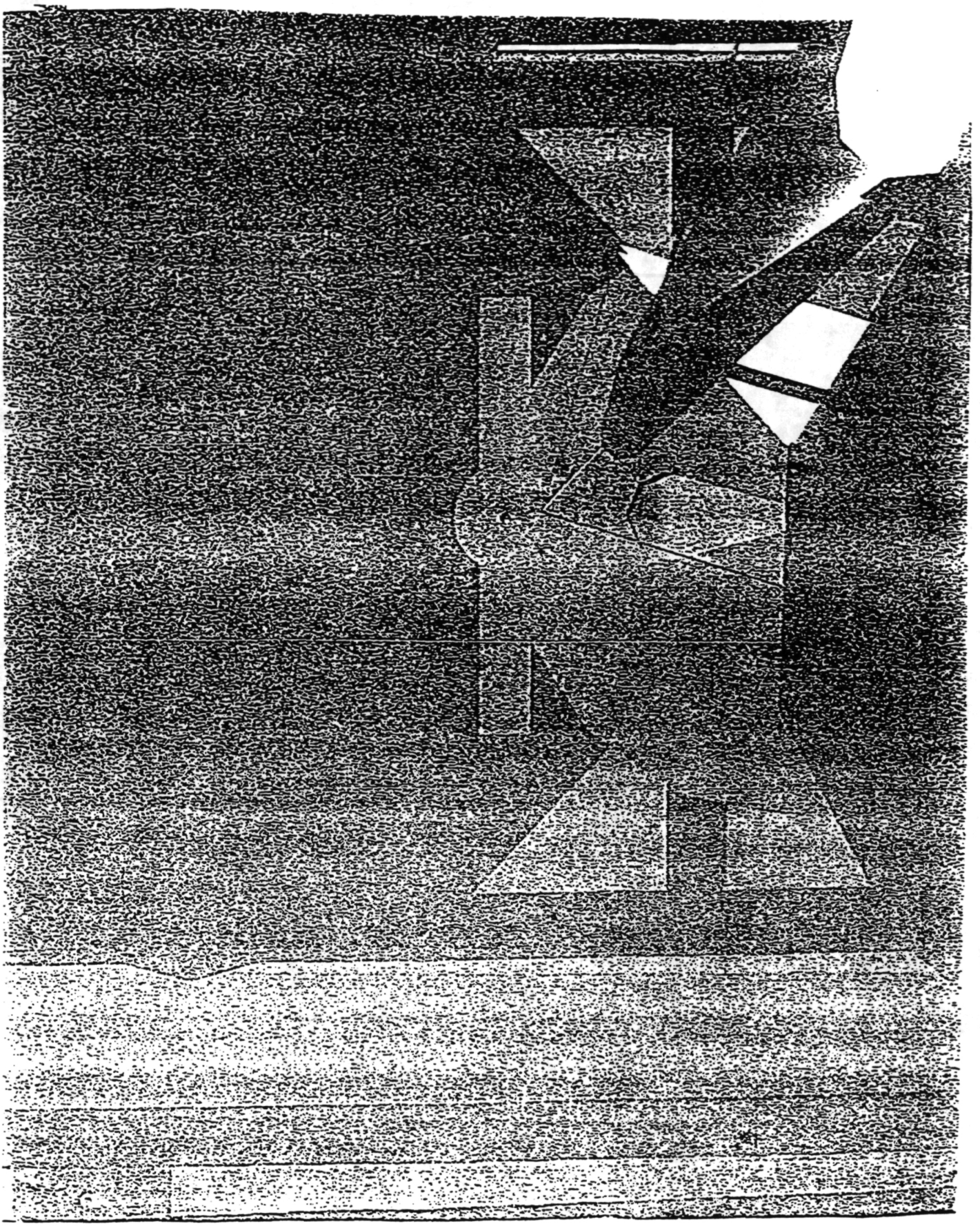


Figure 3.7 Shuttle Arm End Effector Camera View

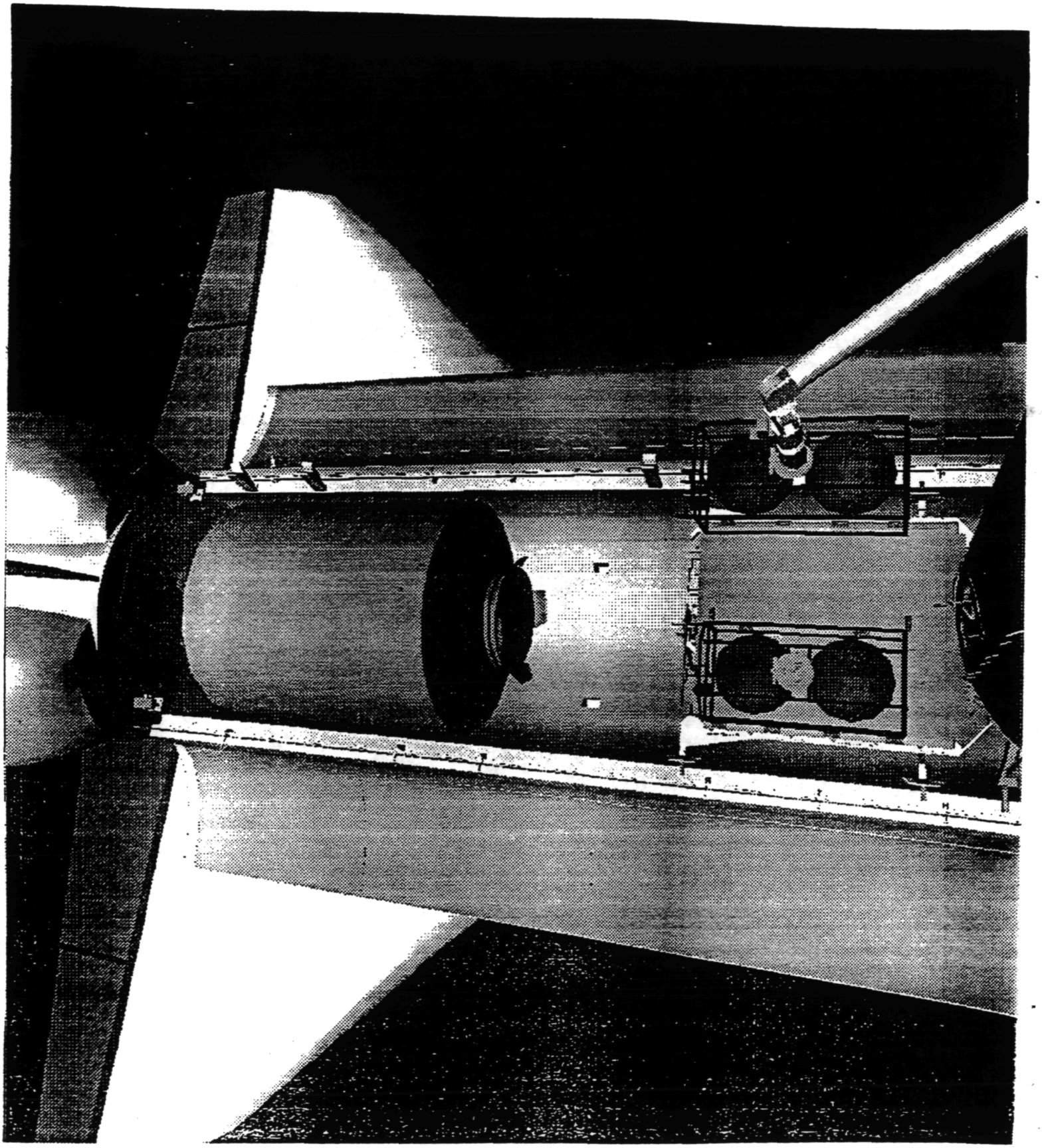


Figure 3.8 Node Camera View Showing Payload Movement

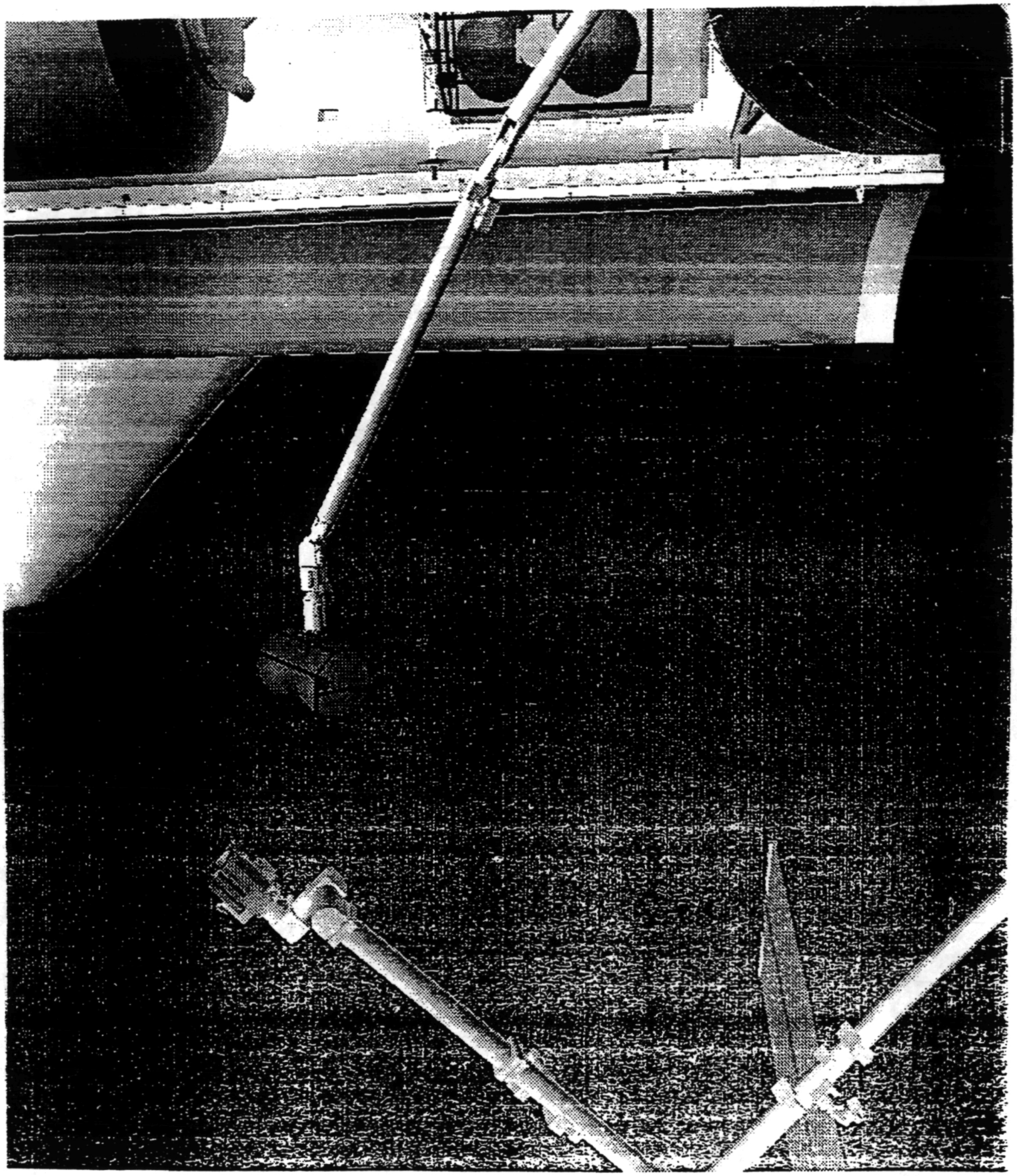


Figure 3.9 Node Camera Showing Shuttle Arm and Space Station Arm

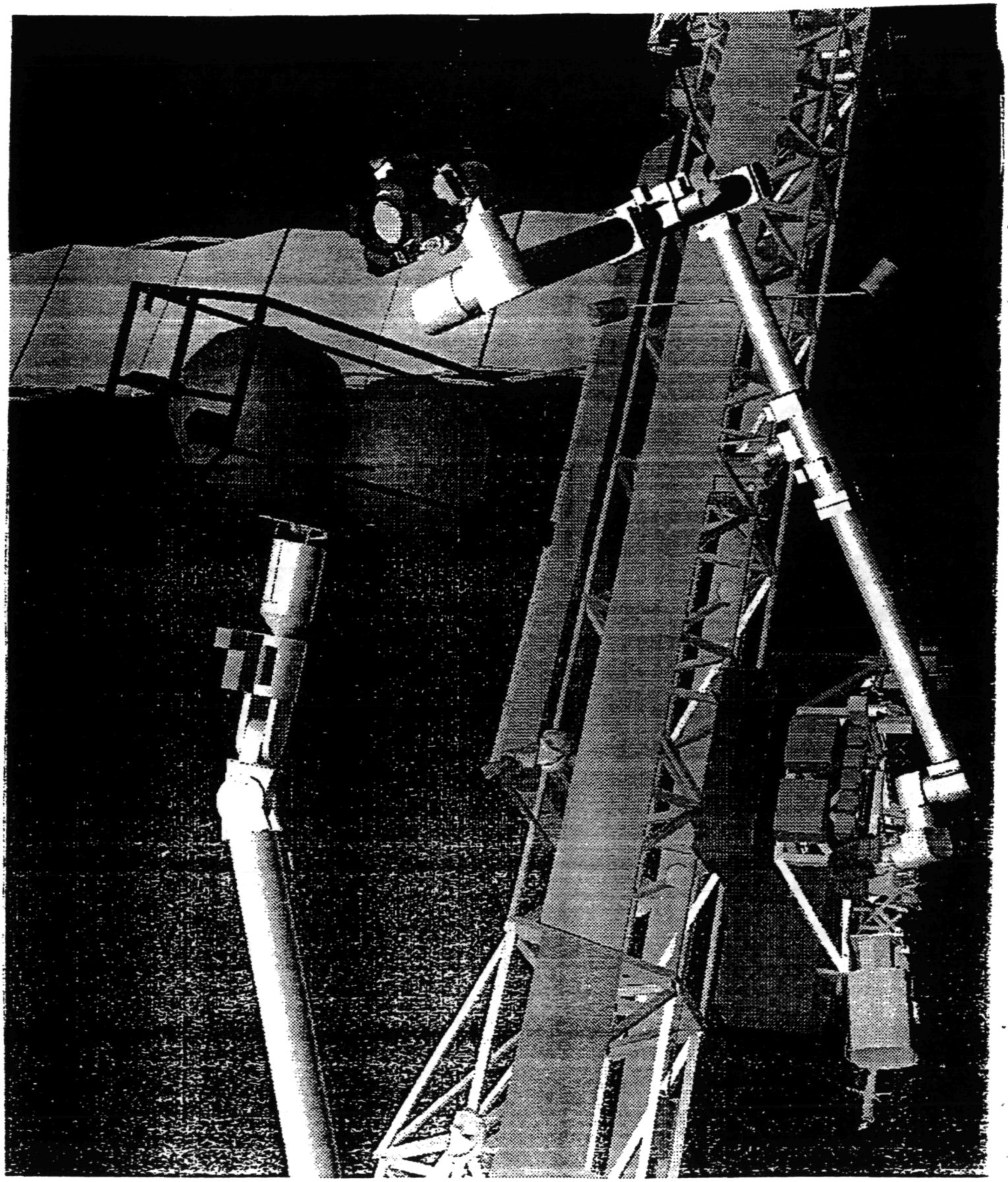


Figure 3.10 Shuttle Payload Bay Camera Showing Payload Handoff to the Space Station Arm

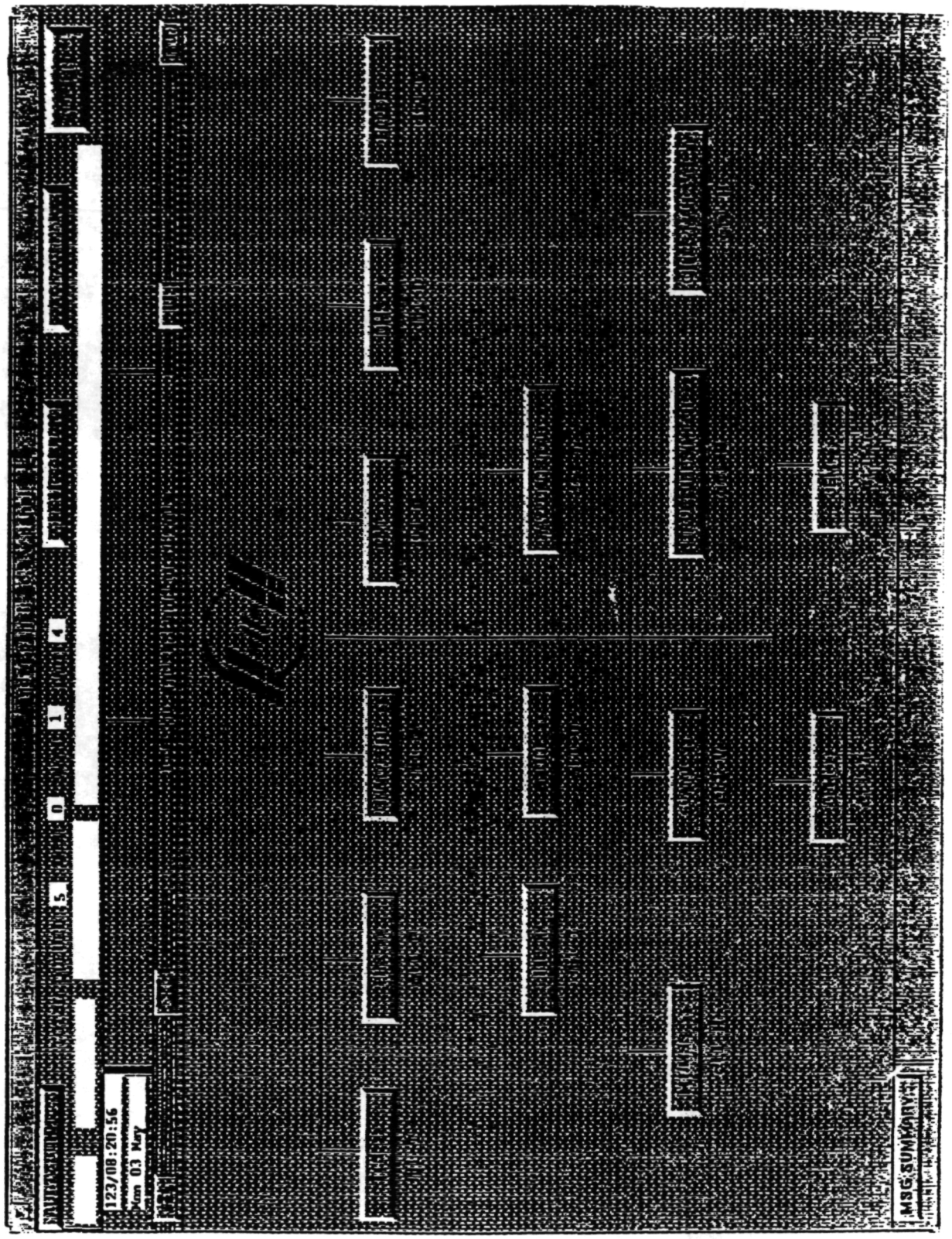


Figure 3.11 Main Menu

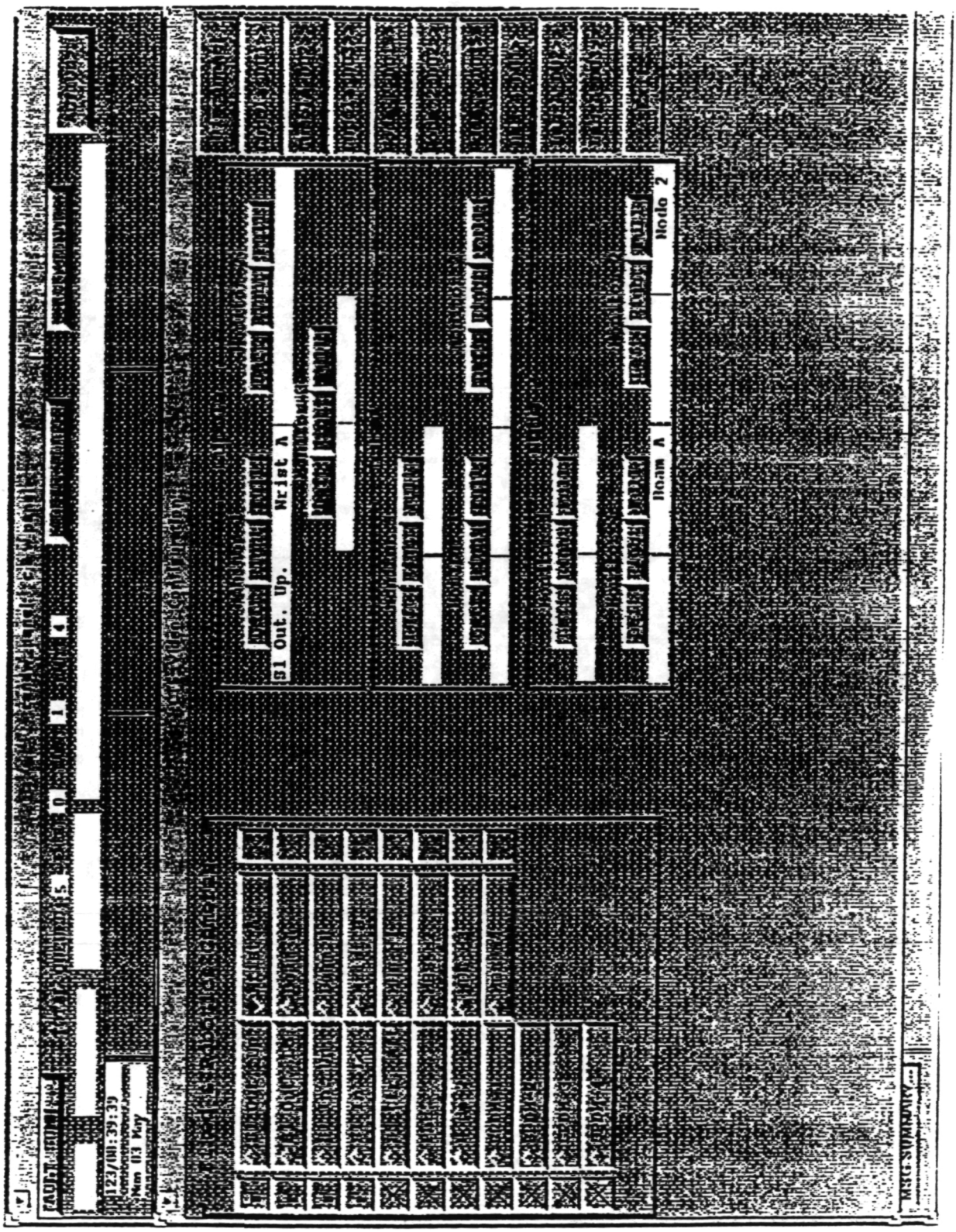


Figure 3.12 Video Configuration Display

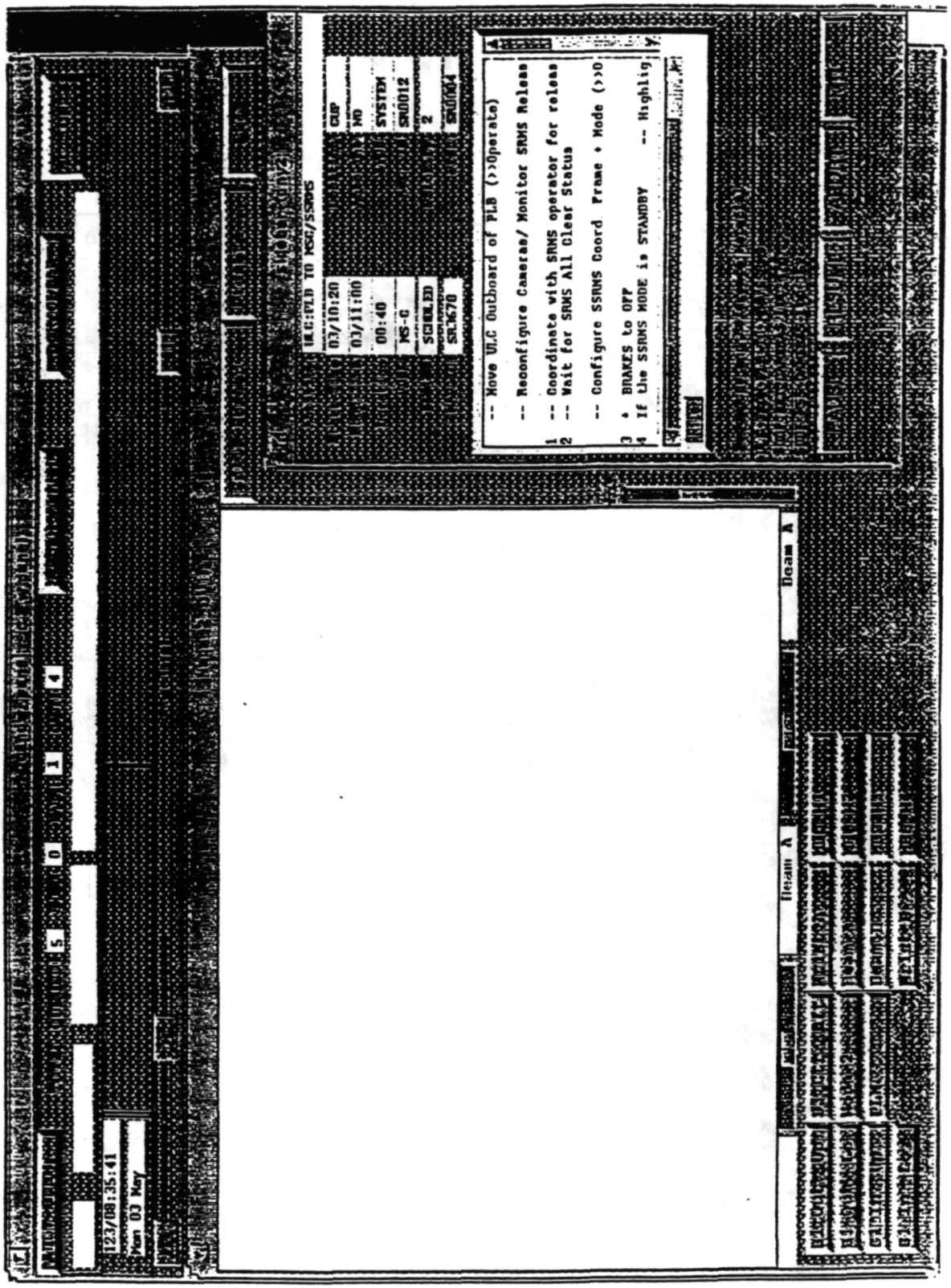


Figure 3.13 Camera Control

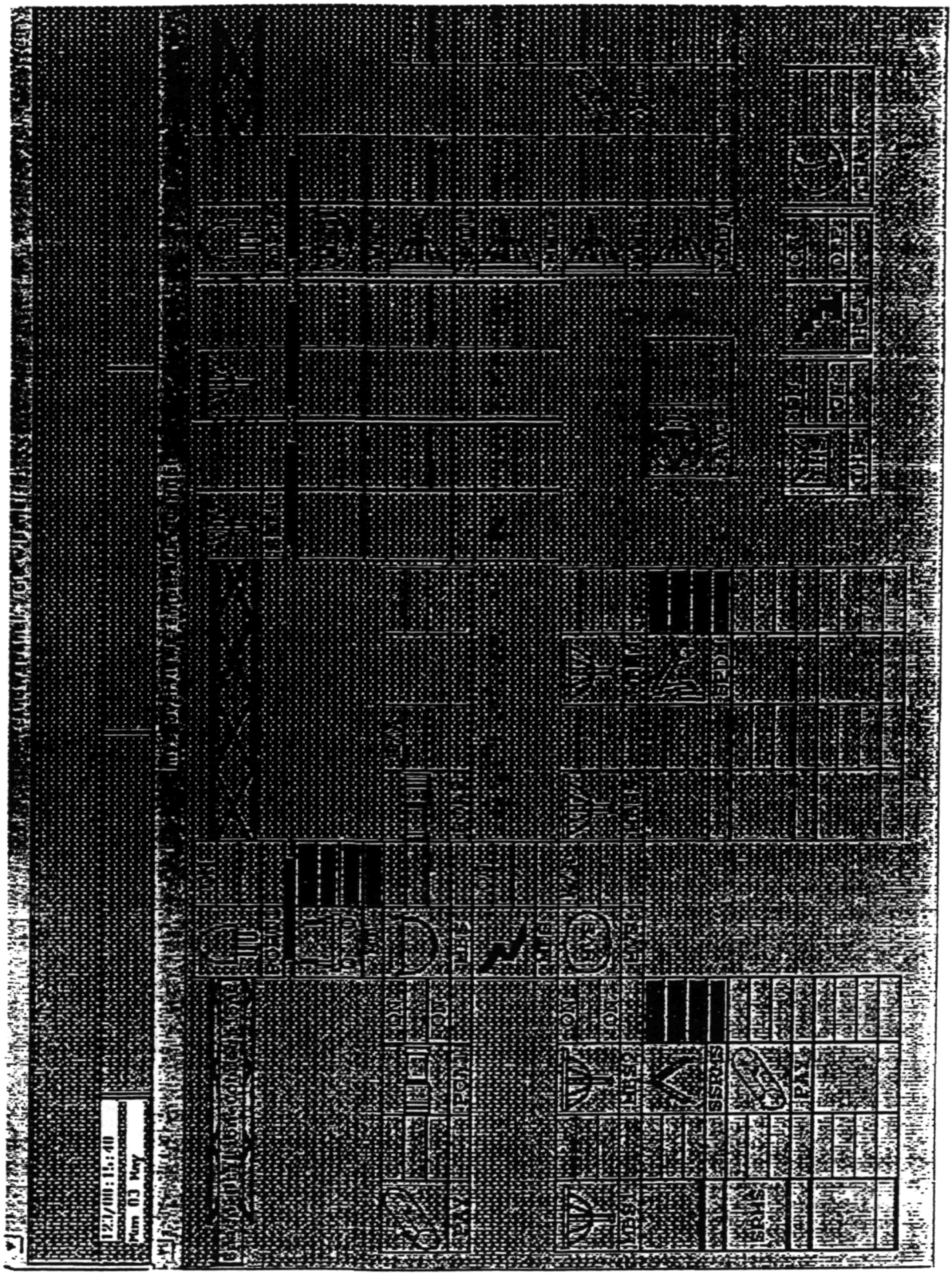


Figure 3.14 Robotic Operations Main Menu

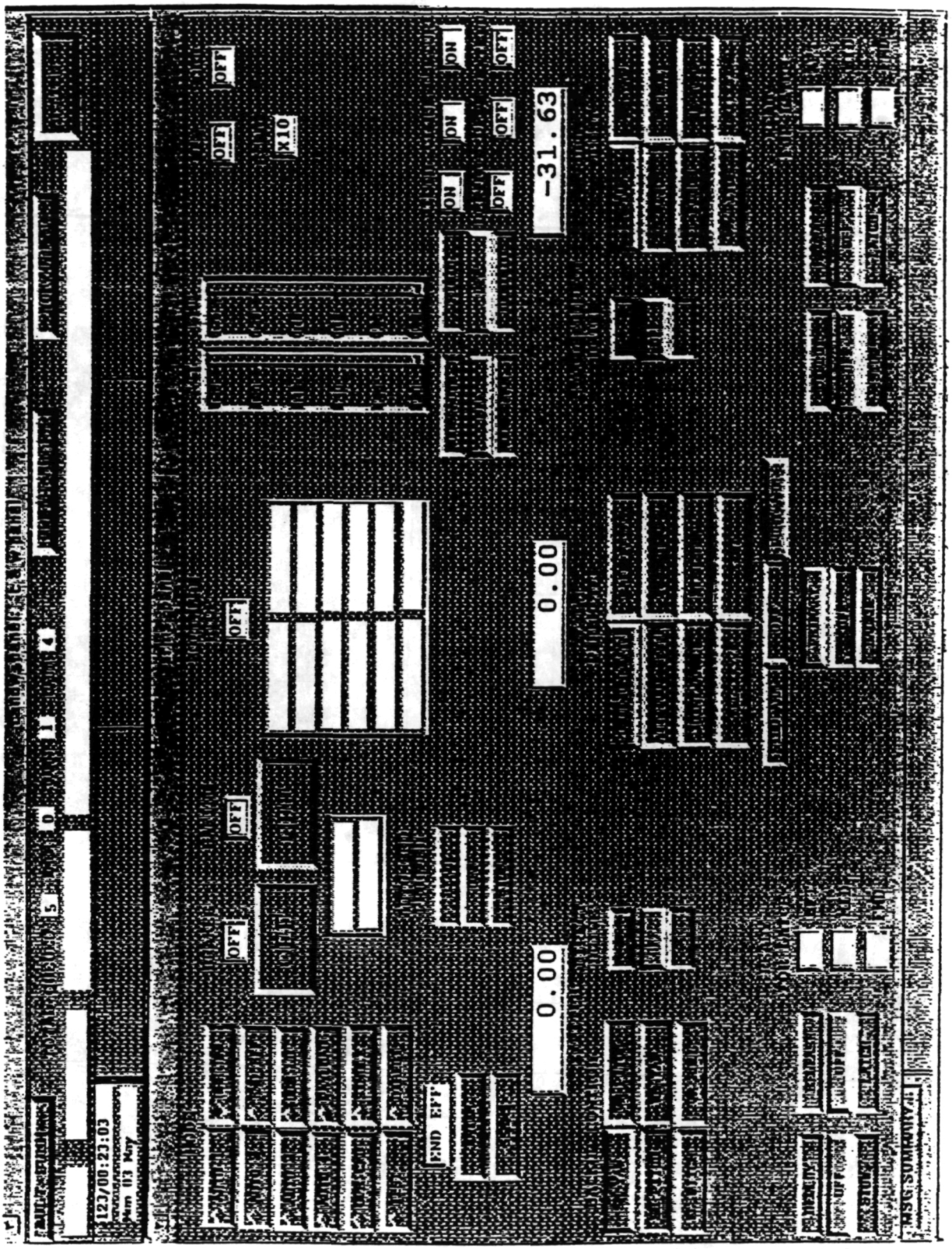


Figure 3.15 Space Shuttle Robotic Arm Operations Display

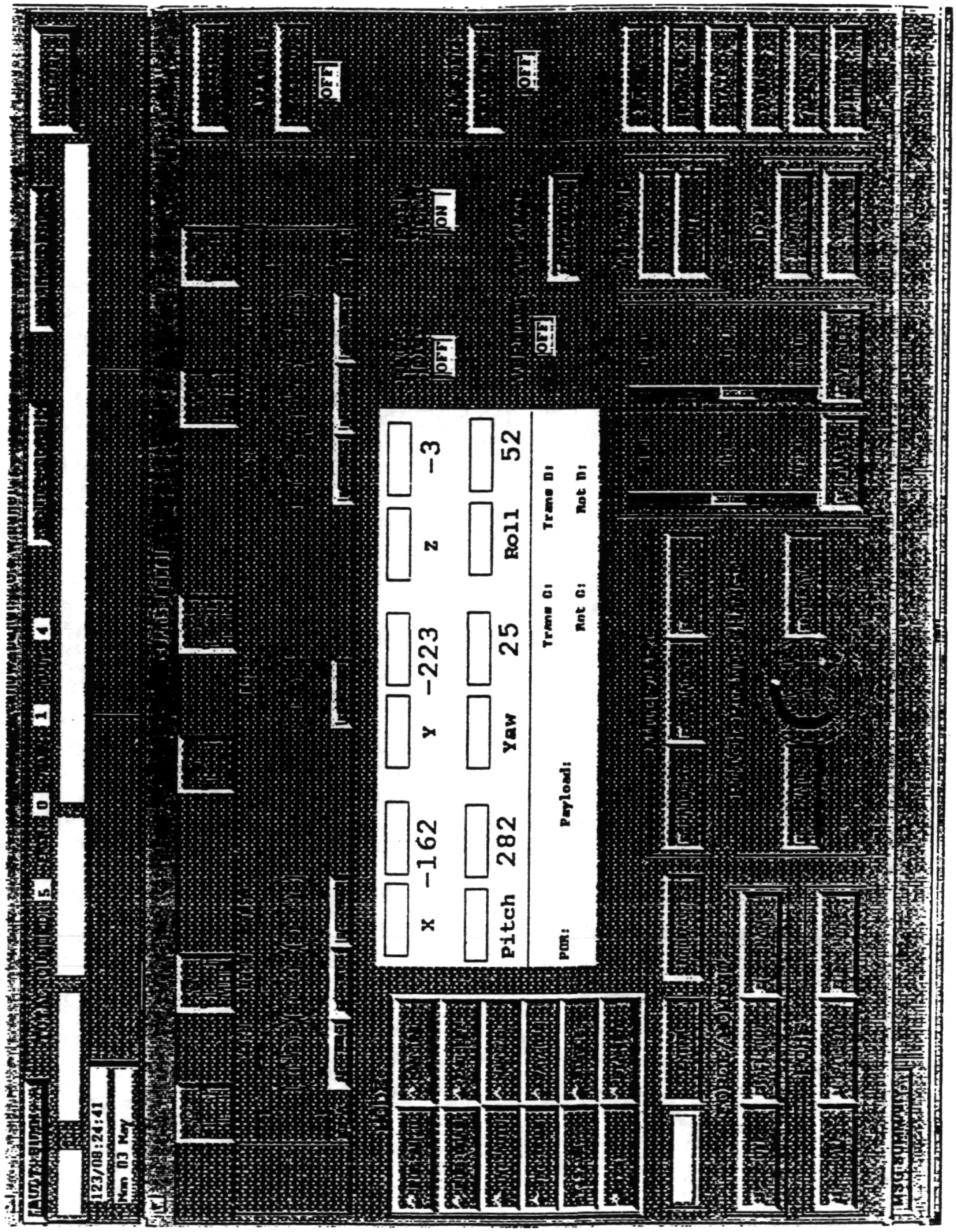


Figure 3.16 Space Station Robotic Arm Operations Display

The Virtual Windtunnel: Visualizing Modern CFD Datasets with a Virtual Environment

Steve Bryson, Computer Sciences Corporation
Applied Research Branch, Numerical Aerodynamic Simulation Systems Division
NASA Ames Research Center, Moffett Field, Ca. 94035-1000
bryson@nas.nasa.gov
Note: This paper has been accepted for presentation at VRAIS '93, Seattle Washington

Abstract

This paper describes work in progress on a virtual environment designed for the visualization of pre-computed fluid flows. The overall problems involved in the visualization of fluid flow are summarized, including computational, data management, and interface issues. Requirements for a flow visualization are summarized. Many aspects of the implementation of the virtual windtunnel were uniquely determined by these requirements. The user interface is described in detail.

1. Introduction

The virtual windtunnel [1][2] is the application of virtual reality interface techniques to the problem of the visualization of the results of computational fluid dynamics (CFD) computations. These results are typically vector and scalar fields in three-dimensional space which change over time. CFD datasets are typically extremely complex, involving time and space-varying structures such as vortices, recirculation, and oscillation.

It was expected that the three-dimensional display and control offered by virtual environment systems would greatly facilitate the investigation of fluid flow data sets, allowing the researcher to explore the data directly. While these expectation have been met, the requirements of flow visualization and the requirements of virtual environment systems were often at odds, forcing compromises between the two sets of requirements. These compromises were critical for the success of the virtual windtunnel and are discussed in this paper. The other topic is the design of the interface which facilitates the visualization task.

The virtual windtunnel is a work in progress. During 1993 it is expected that the virtual windtunnel will be released as a tool for use by a limited user community at NASA Ames. As it is not currently in general use, we can offer only preliminary evaluations of its actual utility. Even in its preliminary stages, however, it has been widely demonstrated and has received an enthusiastic response from both the fluid research and the computer graphics communities.

2 Requirements for the Virtual Windtunnel

The virtual windtunnel is at the intersection of two highly demanding applications of computer graphics: real-time interactive virtual environment systems and unsteady fluid flow visualization. We shall discuss the requirements of these two fields separately.

2.1 Requirements for Unsteady Fluid Flow Visualization

The visualization of modern unsteady fluid flow data sets must confront the following issues:

• **Data management**: The datasets are often in the several gigabyte size range. This includes several timesteps

C-3

of vector and scalar data. The data sets addressed by the virtual windtunnel are defined on several stretched, overlapping grids per timestep. The grids are stretched to conform to the body of the aircraft around which the air is flowing.

- **Computation:** The visualization techniques, such as those based on particle integration (streamlines, streaklines, and particle paths) (figure 1) and isosurfaces, require computations using the CFD data. The computations must be sufficiently accurate to reflect flow phenomena. These computations can be considerable, and for some visualization techniques, i.e. particle paths, potentially requires unpredictable access to the entire data set.
- **Graphics:** The results of the visualization computations must be rendered with sufficient accuracy to represent the flow phenomena. Some visualization techniques, i.e. streamlines, can be rendered as simple lines. Isosurfaces, however, can contain several hundreds of thousands of polygons.

Complex flows may require several visualization displays to be operating simultaneously, further compounding the computation and graphics problem.

- **Cooperative visualization:** The fluid flow community, like any scientific community, operates by cooperative investigation of phenomena. Thus a flow visualization system should support shared, cooperative visualization.
- **User acceptance:** Flow researchers will use a system when the difficulties and training investment are outweighed by the advantages of the visualization system. This means that as much functionality as possible should be included, with no features that do not contribute to that functionality. Difficulty of use should be kept to an absolute minimum.

The benchmark data set used for the virtual windtunnel system is that of a simulated harrier aircraft in hover [3]. This data set has 106 timesteps, with 18 grids/timestep for a total of 2,833,700 points per timestep, or 56 megabytes per timestep, with a total size of 5.6 gigabytes.

2.2 Virtual Environment Interface Requirements

Virtual environment systems rely on an illusion of immersion in an interactive three-dimensional world. This illusion is typically attained through a head-coupled wide-field stereoscopic display combined with a three-dimensional tracker and dataglove. To sustain the illusion and allow useful interaction, the virtual scene must be rendered faster than about 8-10 frames per second. The requirement of good three-dimensional interactivity and control further demands that the time from when a user initiates an action such as movement of the hand to the time when that movement is reflected in the display should be less than 0.1 seconds. Longer times significantly impact user performance in tracking and pick and place tasks [4][5]. Thus if the user interaction controls a visualization task, as is the case in the virtual windtunnel, all computation and display involved in that visualization must take place within 0.1 seconds.

3 Implementation

Simultaneously meeting the requirements of large size data management, extensive computation, and extensive graphics within the virtual environment time constraints in the virtual windtunnel required careful choice of software architecture, hardware, algorithms, and interface design. This section will summarize the design choices that were made to meet these requirements.

3.1 Design Choices

The requirements listed in the last section were each met in different ways:

- **Data management:** The requirement that the data be accessed in apriori unpredictable ways within 0.1 seconds forces the data to be resident in physical memory. No mass storage devices have sufficient bandwidth to access even a single timestep within this time constraint. Example: a single timestep of only the velocity vector field data for the harrier data set described in section 2.1 is 36 megabytes in size, requiring a bandwidth of 360 megabytes per second. When the available physical memory is not sufficient to hold the entire data set, a subset of the data must be chosen. This subset may be generated by either subsampling in time or specification of a small volume of space.
- **Computation:** Particle integration visualization techniques such as streamlines and particle paths are performed by an adaptive second-order Runge-Kutta integration algorithm. The adaptive step size is chosen so that the particle integration takes n steps in a grid cell. The choice of n is controlled in real-time by the user. The integration is performed in grid coordinates, where the coordinates represent actual indices into the data array. In this way time-consuming lookups of the current location for each point of integration using the physical position grid is avoided. The points which are the result of the integration are converted into physical coordinates for rendering via the position grid. When performing the integration, particles may move from one grid to another, invalidating the current computational coordinates (which are defined only for the current grid). Finding the computational coordinates for the new grid requires a table search to convert the current physical coordinates to the new computational coordinates. This extra computation effectively prohibits the vectorization of the particle integration, severely impacting performance on vector processors. This choice of integration method is capable of integrating several thousand particles within the 0.1 second time constraint, allowing the user to observe the paths change in real time as the sources of the integrations are moved about. The marching cubes algorithm for the computation of isosurfaces is adaptively implemented, with the user able to control the step size in computational coordinates.
- **Graphics:** The virtual scene in the virtual windtunnel contains the following: representation of an object, typically an aircraft, around which the simulated air is flowing; various visualization graphics; virtual tools such as menus and sliders; reference markers such as the hand cursor and a floor/horizon reference. Isosurface and object rendering may contain many more polygons than can be rendered within the time constraint, requiring subsampling, compromising quality of image for speed. The user has real-time control over the amount of subsampling. The graphical representation of the paths that arise from particle integration can be simple lines. These lines become a performance bottleneck when the number of integrated points approaches 10,000. The virtual tools can be turned on and off at will, avoiding scene clutter. The hand is represented by a simple three-dimensional crosshair. An articulated hand model is not used to avoid performance overhead and to avoid scene clutter.

3.2 Hardware

There were two hardware configurations implemented for the virtual windtunnel: stand-alone and distributed. Each configuration used the same virtual reality interface hardware. The choice of hardware architecture is primarily driven by the data management requirements. It is expected that as the physical memory and computational power of workstations increases, the stand-alone architecture will become the most useful architecture.

The stand-alone system was implemented on a single workstation, which performed the computation, managed data, handled the I/O devices, and rendered the virtual scene in stereo to the display (see figure 2). The primary workstation used is a Silicon Graphics 380 4D/VGX workstation with 8 33 MHz R3000 processors for a total computational performance of 37 megaflops and 256 megabytes of physical memory. This system has a graphics performance rated at 800,000 polygons/second. The software architecture separates the computation, rendering, and I/O collection into parallel processes using shared memory. In this way no one task slows another. This is important as the graphics must update to reflect the new position of the user's head even though new computations may not have completed. Also, collection of hand and head position can occur as fast as possible. Currently the glove data is collected at 38 Hz, while the head position is collected at 45 Hz.

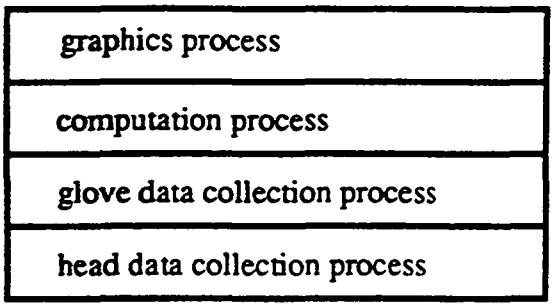


Figure 2: Parallel processes in the stand-alone architecture.

The distributed system uses a Convex C3240 computer with four vector processors and one gigabyte of physical memory for computations. Silicon Graphics VGX family workstations are used for rendering the virtual scene and handling the virtual environment interface hardware. The distributed architecture supports shared interaction, supporting two workstations with virtual environment interfaces (see figure 3). The design of the distributed architecture is greatly facilitated by the use of the Distributed Library by Michael Gerald-Yamasaki [6]. The communications between the Convex and the workstations is over the UltraNet, a gigabit network. Due to limitations with the UltraNet interface card in the workstations, the UltraNet is capable of 13 megabytes/second into the workstations. The primary motivation for the distributed architecture is the access to the gigabyte of memory. The software architecture is shown in figure 4.

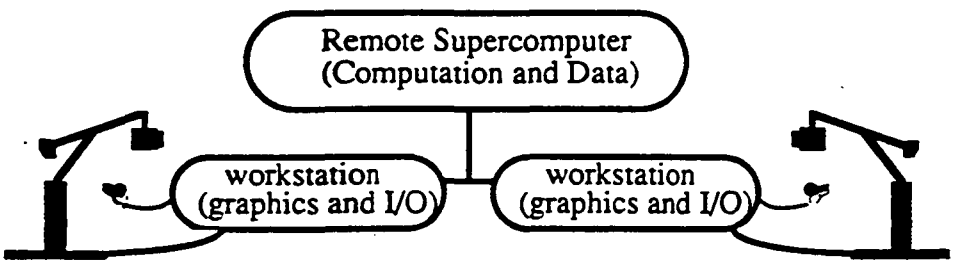


Figure 3: Distributed shared architecture of virtual windtunnel

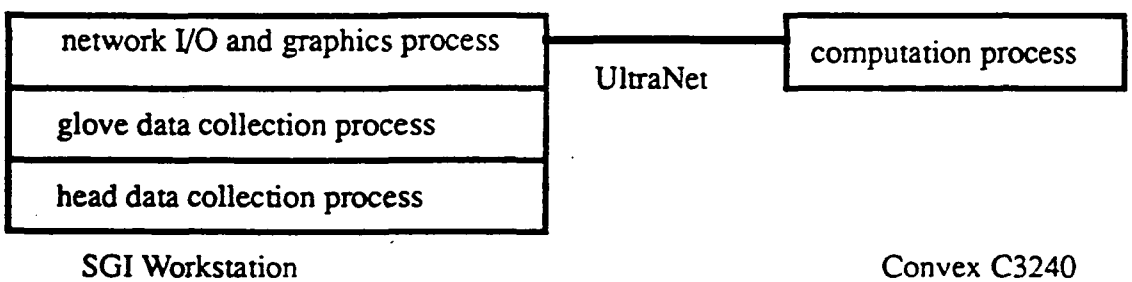


Figure 4: Software architecture in distributed version of virtual windtunnel

The choice of virtual environment display is forced by the requirement that the displayed image be of as high a resolution as possible. The resolution of LCD-based head-mounted displays was considered unacceptable for the purposes of flow visualization. The Fake Space Labs BOOM IIC, a boom-mounted head-coupled stereoscopic display (based on a system described in [7]) with

1000x1000 pixel resolution under the wide field optics was chosen because of its superior display. The BOOM IIC also has superior head-tracking capability via optical encoders at the joints of the supporting boom structure. The ease of use of the boom is also a major advantage over head-mounted systems, greatly facilitating user acceptance. The user interaction is via the standard VPL Dataglove Model II, which uses a Polhemus Isotrak three-dimensional tracker for hand position and orientation, and fiber optic technology to measure the bend of finger joints. The virtual environment interface is shown in figure 5.

3.3 Interface

All operations in the virtual windtunnel are performed with the Dataglove interface. There are two classes of operations: direct manipulation of objects in the environment, and indirect control via virtual menus and sliders (figure 6). All operations are performed with the glove using only two gestures: grab (fist) and point.

Sources of particle integration are grouped into lines known as rakes. There can be several rakes in the environment, each of which may contain sources for one or all of the particle integration types described in section 2. The rake is moved via direct manipulation. Grabbing the center of the rake with the Dataglove causes the rake to move rigidly with the user's hand. Grabbing either end causes the end grabbed to be moved while the other end remains stationary. A sphere is drawn when the virtual hand is sufficiently close to a rake to grab part of it, providing feedback to the user (figure 7). This interface allows a rake to be positioned arbitrarily with arbitrary orientation. This method of controlling the orientation of the rakes was chosen over the use of the hand orientation due to the limited range of motion of the human wrist.

Various aspects of the environment are controlled via virtual menus[8]. Making a point gesture in empty space in the virtual environment causes a multi-level hierarchical menu to pop up in three-dimensional space within the user's field of view. The menu remains as long as the point gesture is held by the user. While the menu is up, the user's hand orientation information is used to point at various menu items. Releasing the point gesture while pointing at a menu item causes that menu item to be executed.

Rake parameters such as the number of particle integration sources are indirectly controlled via virtual sliders. These sliders exist in the three-dimensional environment and output values determined by the user making a point gesture in the active region of the slider. The sliders can be moved by making a grab gesture in the region of the slider. Sliders are toggled on and off via the virtual menus.

Navigation within the environment uses a paradigm in which the user stays in one place and moves the environment about. This is accomplished by making a grab gesture in empty space and moving the entire environment with the motion of the user's hand. When combined with a variable scale controlled via a virtual slider, this interface allows rapid and high-precision maneuvering in the environment. The virtual tools such as menus and sliders are not effected by this motion. The scale and grab paradigm is significantly easier to control than the point and fly navigation paradigm.

To summarize, these are the following actions of the gestures in the virtual windtunnel:

context \ gesture	fist	point
empty space	move data	pop up menu
virtual slider	move slider	change slider value

context \ gesture	fist	point
rake grab point	move rake	no action

The extensive use of hand gestures read by the glove in the virtual windtunnel requires a robust and reliable gesture recognition algorithm. This is accomplished using only the middle joints of the four fingers. The values measured at the knuckle joints and the thumb are ignored. First the raw values output by the Dataglove are calibrated to actual finger bend angles. Then the angles read by the glove are compared with a lookup table, to identify if the gesture is either a fist or a point. Recognizing only three gestures (fist, point, no gesture) allows forgiving and tolerant gesture recognition. With this gesture recognition algorithm, new users require a few minutes training and practice to make the gesture reliably. The glove can be calibrated from within the environment, using only two gestures. The calibration process is controlled via buttons on the BOOM IIC display.

4 Conclusions

The virtual windtunnel system has successfully implemented a flow visualization application in a virtual environment. While many refinements will be required to turn the virtual windtunnel into a useful tool, the basic issues have been addressed and solved.

Acknowledgments

The virtual windtunnel has had many participants. Creon Levit and the author designed the initial concept and prototype. The distributed architecture was developed by Michael Gerald-Yamasaki and the author. Al Globus and Jeff Hultquist contributed computational code, and Rick Jacoby and Diglio Simoni contributed interface code.

References

[1] Bryson, S. and Levit, C.," The Virtual Wind Tunnel: An Environment for the Exploration of Three Dimensional Unsteady Flows", *Proceedings of Visualization '91 San* Diego, Ca, Oct. 1991, also *Computer Graphics and Applications* July 1992

[2] Bryson, S. and Gerald-Yamasaki, M., "The Distributed Virtual Wind Tunnel", *Proceedings of Supercomputing '92 Minneapolis*, Minn, Nov. 1992

[3] Smith, M., Chawla, K., and Van Dalsem, W., "Numerical Simulation of a Complete STOVL Aircraft in Ground Effect", paper AIAA-91-3293, American Institute of Aeronautics 9th Aerodynamics Conference, Baltimore Md. 1991

[4] Bryson, S., "Impact of Lag and Frame Rate on Various Tracking Tasks", *Proceedings of SPIE Conference on Stereoscopic Displays and Applications*, San Jose, Ca., Feb. 1993

[5] Sheridan, T. and Ferriell, W, *Man-Machine Systems*, MIT Press, Cambridge, Ma. 1974

[6] Yamasaki, M., Distibuted Library, NAS Applied Research Technical Report RNR-90-008, April 1990

[7] MacDowall, I., Bolas, M., Pieper, S., Fisher, S. and Humphries, J., Implementation and Integration of a Counterbalanced CRT-based Stereoscopic Display for Interactive Viewpoint Control in Virtual Environment Applications, *Proceedings of the 1990 SPIE Conference on Stereoscopic Displays and Applications*, Santa Clara, Ca. 1990

[8] Jacoby, R. H., Using Virtual Menus in a Virtual Environment. *Proceedings of the Symposium on Electronic Imaging Science & Technology*, International Society for Optical Engineering/Society for Imaging Science & Technology, Volume 1668.

Practical VR--Five Years of Lesson's Learned

Mark Bolas
Fakespace, Inc.
4085 Cambell Avenue
Menlo Park, CA 94025
415-688-1940 415-688-1949

Fakespace Inc. began in 1988 with its Molly tm Telepresence System and Boom tm Stereoscopic Viewer at the NASA Ames VIEW laboratory. This technical and human factors based foundation has proved invaluable over the past five years as our product line and customer list has grown with down-to-earth practical Virtual Environment products and applications. This paper will describe the evolution of these products and how they are currently being used everyday in real applications to solve problems and to make engineers and designers more productive.

Specific examples will be drawn from various clients in three major application areas: the visualization and analysis of design data, training applications, and scientific visualization.

In conjunction with these specific examples, a theoretical framework for interpreting and predicting the success of a Virtual Environment system for a specific application will be presented.

Using Virtual Environment Technology for Preadapting Astronauts to the Novel Sensory Conditions of Microgravity

K. M. Duncan*, D. L. Harm PhD**, W. G. Crosier*, and J. W. Worthington*
*KRUG Life Sciences, Inc.,
1290 Hercules Drive, Suite 120,
Houston, Texas 77058

**Space Biomedical Research Institute, NASA/JSC, Houston, Texas 77058

ABSTRACT

A unique training device is being developed at the Johnson Space Center Neurosciences Laboratory to help reduce or eliminate Space Motion Sickness (SMS) and spatial orientation disturbances that occur during spaceflight. The Device for Orientation and Motion Environments Preflight Adaptation Trainer (DOME PAT) uses virtual reality technology to simulate some sensory rearrangements experienced by astronauts in microgravity. By exposing a crew member to this novel environment preflight, it is expected that he/she will become partially adapted, and thereby suffer fewer symptoms inflight.

The DOME PAT is a 3.7 m spherical dome, within which a 170° by 100° field of view computer-generated visual database is projected. The visual database currently in use depicts the interior of a Shuttle spacelab. The trainee uses a six degree-of-freedom, isometric force hand controller to navigate through the virtual environment. Alternatively, the trainee can be 'moved' about within the virtual environment by the instructor, or can look about within the environment by wearing a restraint that controls scene motion in response to head movements.

The computer system is comprised of four personal computers that provide the real time control and user interface, and two Silicon Graphics computers that generate the graphical images. The image generator computers use custom algorithms to compensate for spherical image distortion, while maintaining a video update rate of 30 Hz.

The DOME PAT is the first known such system to employ virtual reality technology to reduce the untoward effects of the sensory rearrangement associated with exposure to microgravity, and it does so in a very cost-effective manner.

INTRODUCTION

In the weightless environment of space flight, the absence of a gravity stimulus to the gravity receptors in the vestibular, proprioceptive and somatosensory systems alters the relationship among sensory inputs. The altered sensory stimulus conditions are responsible for space motion sickness, spatial orientation and motion perception disturbances experienced by many astronauts and cosmonauts. Two devices have been designed to partially simulate the sensory conditions present in microgravity in order to help astronauts more quickly adapt to these conditions during a Shuttle mission. The DOME PAT is designed to achieve graviceptor stabilization to simulate the absence of gravity information. In this system, the trainee can move about the virtual environment in six degrees of freedom (X, Y, Z, roll, pitch, yaw) without a change in the gravity vector. Actual head rotation about an earth-vertical axis keeps the trainee's orientation fixed with respect to the gravity vector while preserving the normal relationships between rotation of the visual field (in one plane at a time) and semicircular canal outputs. Actual head rotation about non-vertical axes is prevented, although virtual rotation may be enabled about any

axis. By helping astronauts become comfortable with rotations in all body axes without a tilting gravity vector, the DOME PAT should facilitate crew members' performance and sense of well being in microgravity.

MECHANICS

The DOME PAT is a 12 ft (3.7 m) diameter spherical dome. The inner surface is painted white and serves as a projection surface for two video projectors with custom wide angle optics. The field of view for the trainee is 170° horizontally by 100° vertically. The projectors, along with an adjustable trainee restraint assembly, are mounted on a 6 ft (1.8 m) diameter rotating base in the bottom of the dome. The base rides on a 92 cm diameter bearing, and is gear driven by a servomotor to allow computer-controlled rotation of the trainee and the projectors about a vertical axis. This single degree of freedom of rotation is the only real movement allowed by the system.

The trainee restraint adjusts for three different trainee positions: 1) sitting upright, 2) lying on left or right side, or 3) lying supine looking up at the top of the dome. For the upright and on-side positions, the projectors' optical axes are approximately horizontal, and for the supine position the projector assembly is tilted back so that images are projected on the top of the dome.

CONTROLS

A computer generated virtual environment is projected on the inside of the spherical dome. The trainee uses a six degree-of-freedom isometric force hand controller to move through the virtual environment. Joystick or forceplate emulation for the hand controller is configured with a software switch. There are three different motions that the hand controller can simulate: position, velocity or acceleration. It is generally configured as a forceplate in the acceleration mode with no damping; however, there is a deceleration factor to provide fast cancellation of motion in each degree of freedom. The instructor has an identical controller and may 'move' the trainee to provide instruction or demonstrate passive motion. The trainee also wears a head restraint that is mounted in the center of the dome. This head restraint contains position and force transducers which sense actual and attempted head movements. The only actual head movement allowed is about an earth-vertical axis in order to keep the trainee's orientation fixed with respect to the gravity vector. Attempted off vertical head movements can drive virtual rotation allowing the trainee to 'experience' the rotation without actually changing their orientation with respect to the gravity vector.

COMPUTERS

The DOME PAT computer system consists of four AST personal computers (PCs) and two Silicon Graphics computers (SGIs) networked together. Refer to Figure 1. The four PCs are networked using Arcnet and Novell Netware software. The SGIs use ethernet and NCSA Telnet software to communication with each other and with one of the PCs.

The Real Time Positioning System (RTPS) gets the raw signals from the hand controller and the head transducers and calculates a delta position vector (DPV). It sends the DPV to the Real Time Controller (RTC). The RTPS also drives the servomotor when actual rotation is enabled. The RTC takes the DPV from the RTPS, calculates the current eyepoint and sends it to the SGIs. The RTC also performs collision detection to prevent the trainee from moving "outside" the virtual environment. The SGIs function as a two channel computer image generator. They draw the next frame of the scene based on the current eyepoint and send RS343 video signals to the projectors at a rate of 30 Hz to display the virtual environment with a resolution of 960 X 1280 pixels for each channel.

Paradigm Simulation, Inc. of Dallas provided the image generation software which is based on their commercially available product, VisionWorks™, with some custom code for special functions and tools. Due to our spherical dome configuration with the very wide viewing area, in addition to the standard image generator functions (3-D to 2-D mapping, clipping, drawing, etc.), the SGIs must perform some special functions such as spherical distortion compensation, edge tessellation, and edge blending. The distortion correction is also required because the projectors are not located at the same position as the trainee's eyes.

There is a distortion map for each channel and each orientation (left upright, left supine, right upright and right supine). Paradigm built a nice interactive tool that is used to build and modify the distortion maps. Basically the distortion map takes each vertex of the polygons to be displayed and remaps it from flat space to a new "warped" location so that the projected image appears with the correct perspective on the curved surface of the dome. This vertex remapping approach had a potential impact on our visual database design, because any line or polygon edge that was "too" long would not get distorted properly. And this brought up another issue: How long is "too" long? The answer is: it depends! It depends on what percent of the viewing area the line or polygon is taking up. For example, if you are only several centimeters from a "small" polygon, it will appear very large in the viewing area.

One way to resolve this problem is to use visual databases consisting of many very small polygons. However, since we are limited to a finite number of polygons that can be processed and still maintain the 30 Hz update rate, an extremely complex database (one with many polygons) is not feasible. Another approach is tessellation or breaking up large polygons into smaller ones. Ideally, once a polygon crosses a threshold of "largeness" in the viewing area, it could be broken into smaller polygons in real-time. However, due to real-time processing limitations, it is not possible to do this across our entire viewing area. Therefore, since it is crucial to maintain proper distortion along the edges especially in the center of the field of view where the left and right projections meet (the "seam"), a special function known as edge tessellation was implemented. This automatically breaks up polygons that intersect with the edges into small polygons, allowing them to be properly distorted which creates smooth projection edges. This facilitates a smooth seam down the middle.

To smooth the seam between the two projections even more, particularly since the colors between the two projectors are not perfectly matched, another special function blends the colors along the center seam. This is called edge blending. The two projections are slightly overlapped. Using translucent edge matching polygons, the intensity of the overlap portion from each projection fades out toward the center edge. Therefore, the colors from the two projections are blended down the center seam where they overlap. The width of the blending band and the intensity function can be adjusted offline to achieve the desired effect.

FUTURE

There are a number things we would like to do in the future to improve the DOME PAT trainer. First, we would like to upgrade the image generator hardware to the Silicon Graphics reality engine in order to increase processing speed and power capability. With the increased processing power, a different approach for distortion compensation could be implemented. More accurate distortion compensation than our current technique could be achieved by remapping each pixel instead of just remapping each polygon vertex. This would eliminate any anomalies that currently occur if a polygon is too long to be properly distorted. Also, it would eliminate the anomalies that occur since the polygon warping look up table is not a one-to-one map. Some points are remapped to the same location, while other locations are blank because no point is mapped there. With the extra processing capabilities and new distortion approach, texture could also be incorporated. Texture would increase the realism of the virtual environment and reduce the number of polygons in the visual databases.

Second, we would like to combine the functions of all of the DOME PAT PCs onto one machine. This would simply the operation for the user. It would also reduce maintenance costs by eliminating of some of the non-standard hardware and software (special graphics boards and their FORTRAN library, 386 co-processor board inside one of the 286 machines and the special software for shared memory and communications.)

Finally, we plan to add more visual databases such as the mid-deck, flight-deck, payload bay with doors open, etc. This would allow us to present different virtual environments, or perhaps combine them (flight-deck, mid-deck, with either the spacelab or the payload bay with open doors) for a large virtual environment with different areas to explore.

CONCLUSIONS

In conclusion, the DOME PAT meets some complex scientific requirements for (1) holding vestibular graviceptor signals constant, (2) allowing 6 degrees-of-freedom of full virtual motion, and (3) allowing active trainee

interaction with the novel sensory environment. This is accomplished in a very cost-effective manner using commercially available hardware and software with some custom software. Our image generators provide real-time distortion correction for a wide angle spherical display. Other simulators use analogous systems which employ multi-million dollar computer image generator systems. Our hardware and maintenance costs are greatly reduced by using standard computers, and system upgrades are simplified.

The DOME PAT is currently being used in an actual training environment. It is being used to demonstrate perceptual illusory phenomena to astronauts and to teach them to describe their perceptual experiences. Anecdotal reports from several crew members suggest that a number of perceptual experiences produced by the DOME PAT system are similar to those experienced inflight, during entry, or immediately postflight.

DOME PAT

Device for Orientation and Motion Environments
Preflight Adaptation Trainer

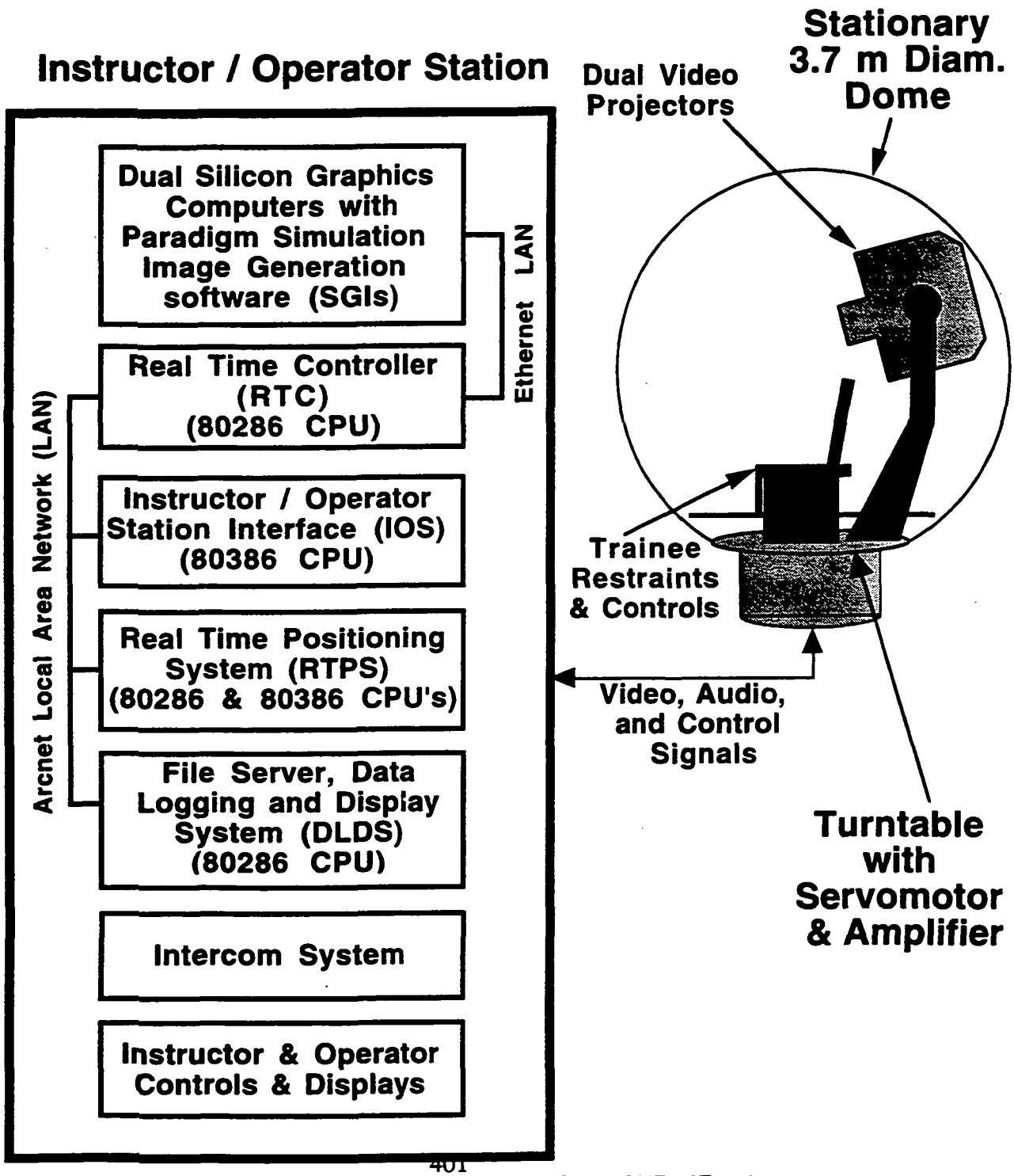


Figure 1 Block diagram of the DOME PAT system.

Development and Implementation of Inflight Neurosensory Training for Adaptation/Readaptation (INSTAR)

D. L. Harm*, F.E. Guedry, Donald E. Parker***, and M.F. Reschke***

*Space Biomedical Research Institute,
NASA/JSC
Houston, TX 77058

**Naval Aerospace Medical Research Laboratory
Pensacola, FL 32508

***Dept. of Psychology
Miami University
Oxford, OH 45056

ABSTRACT

Resolution of space motion sickness, and improvements in spatial orientation, posture and motion control, and compensatory eye movements occur as a function of neurosensory and sensorimotor adaptation to microgravity. These adaptive responses, however, are inappropriate for return to Earth. Even following relatively brief space Shuttle missions, significant re-adaptation disturbances related to visual performance, locomotion, and perceived self-motion have been observed. Russian reports suggest that these disturbances increase with mission duration and may be severe following landing after prolonged microgravity exposure such as during a voyage to Mars. Consequently there is a need to enable the astronauts to be prepared for and more quickly re-adapt to a gravitational environment following extended space missions. Several devices to meet this need are proposed including a virtual environment - centrifuge device (VECD). A short-arm centrifuge will provide centripetal acceleration parallel to the astronaut's longitudinal body axis and a restraint system will be configured to permit head movements only in the plane of rotation (to prevent "cross-coupling"). A head-mounted virtual environment system will be used to develop appropriate "calibration" between visual motion/orientation signals and inertial motion/orientation signals generated by the centrifuge. This will permit vestibular, visual and somatosensory signal matches to bias central interpretation of otolith signals toward the "position" responses and to recalibrate the vestibulo-ocular reflex (VOR).

INTRODUCTION

Stable vision, posture and locomotion, and accurate perception of how one is oriented in and moving through space depends on complex physiological integration of signals from virtually every sensory system. Changes in the environment and the way we interact with the new stimuli in that environment will result in a different interpretation by the nervous system of the incoming sensory information. Initially on-orbit, the response outputs from the nervous system that control vision, posture, perception of orientation and motion are inappropriate and result in space motion sickness and disturbances in these functions. Over time, however, people adapt to these changes in appropriate ways. Changes in the sensory stimulus conditions in the environment are referred to as sensory stimulus rearrangements. The environment of orbital flight and return to Earth represent the stimulus rearrangements that are our immediate concern.

Neurosensory adaptation may be defined as a transient or long-term modification of sensory apparatus and perceptual processes which involves forming new associations between stimuli and responses to those stimuli. Adaptation is enhanced by active exploratory behavior in a new environment with resulting feedback and the formation of associations between sensory inputs and response outputs that promote appropriate orientation and movement in that environment (microgravity). *Readaptation* is the process(es) by which the nervous system relearns the associations between sensory inputs and response outputs that promote appropriate orientation and movement on Earth. Russian reports suggest that these disturbances increase with mission duration and may be severe following landing after prolonged microgravity exposure such as during a voyage to Mars or and extended stay on space stations.

On orbit, the absence of an effective gravity vector creates the requirement for adaptive changes in the sensorimotor and perceptual systems that, on Earth, subserve the voluntary and reflexive control of eye, head, and body orientation and motion relative to Earth. The favorable consequences to the astronaut of the adaptive changes to microgravity are improved sensorimotor performance, reduction in spatial orientation and motion perception disturbances (illusions of self and/or surround motion), and abatement of space motion sickness (SMS). The unfavorable consequences for these improvements on-orbit are maladaptive sensorimotor and perceptual reactions during return to Earth and for some time after landing.

Our overall goals are to develop training devices and procedures that will allow astronauts first to develop appropriate responses preflight (on Earth) to the sensory conditions of microgravity and second to maintain appropriate respones (on-orbit) to the sensory conditions of Earth. A brief description of the Preflight Adaptation Training (PAT) program currently underway and a proposed Inflight Neurosensory Training for Adaptation/Readaptation (INSTAR) program will be presented in the next two sections of this paper. The underlying design principles for INSTAR are essentially the inverse of those for PAT. That is, PAT provides an environment where: (1) in the presence of gravity, stimulation to the graviceptors is held constant (graviceptor stabilization) during real and visually induced self-motion in the pitch, roll and yaw planes, and (2) adaptation to rotation without changes in the gravity vector is accomplished within a visual context representative of the interior of the spacecraft. INSTAR, on the other hand, should provide an environment where: (1) in the absence of gravity, the equivalent of a 1g stimulus to the graviceptors is provided during static and dynamic head rotation, and (2) maintenance of adaptation to Earth would be accomplished within visual contexts representative of familiar places on Earth.

PREFLIGHT ADAPTATION TRAINING (PAT)

The general concepts underlying PAT are: (1) that the brain can learn to "recalibrate" the incoming sensory signals in a manner that would be appropriate to microgravity, and (2) because the central nervous system is "plastic," people can learn and store perceptual and sensorimotor responses to different sensory stimulus conditions. That is, they can develop *dual-adapted states* and can learn to switch rapidly between different stimulus conditions. For example, experienced SCUBA divers exhibit altered compensatory eye movement gains immediately upon donning their diving masks. Apparently, they have learned that a higher gain is required to compensate for the magnification of the visual scene due to the air-water interface. Removal of the mask is associated with a nearly immediate return of the normal gain. The pimary goals of this effort are to develop trainers that can simulate the stimulus rearrangements of microgravity and develop training scenarios and procedures to facilitate adaptation to microgravity and readaptation to Earth (dual-adaptation), and the ability to switch rapidly between these states.

Our general approach was to develop two part-task trainers to induce the appropriate adaptive responses. The Device for Orientation and Motion Environments Preflight Adaptation Trainer (DOME PAT) incorporates virtual enviornment technology to: (1) produce the perception of self-motion (vection) without changes in gravity inputs to the vestibular system, and (2) provide an environment where the trainee can "virtually" position themselves in any orientation, and view their visual environment from an infinite number of perspectives without changes in gravity inputs to the vestibular system. A more detailed description of this system in available in this proceedings in a paper titled "Using Virtual Environment Technology for Preadapting Astronauts to the Novel Sensory Conditions of Microgravity."

The tilt-translation device (TTD) couples translational visual motion with body tilt to: (1) produce the perception of translational (linear) self -motion, and (2) facilitate central nervous system reinterpretation of tilt motion as translational motion. Adaptation induced by this device is based on the otolith tilt-translation reinterpretation (OTTR) hypothesis of neurosensory adaptation to microgravity. This hypothesis states that during adaptation to microgravity, the brain learns to process information from the vestibular graviceptors (otoliths) and other graviceptors differently. On Earth, inputs from otolith receptors is interpreted by the brain as linear motion or as head tilt with respect to gravity. Because of the absence of stimulation from gravity during orbital flight, interpretation of otolith responses as tilt is meaningless. Therefore, the brain adapts to sustained microgravity by reinterpreting all otolith receptor output as linear motion. (see Bibliography for additional readings on PAT)

INFLIGHT NEUROSENSORY TRAINING FOR ADAPTATION/READAPTATION (INSTAR)

The overall goals of INSTAR are to enable astronauts to be prepared for and more quickly readapt to Earth's gravitational environment following extended space missions in order to reduce eye movement control, postural/locomotion, and perceptual disturbances associated with return to Earth and the potential impacts of such disturbances on: (1) orbiter control during landing, (2) egress from the orbiter, and (3) physical injuries from falls. Long-duration missions require inflight countermeasures because, as they are more remote in time, preflight countermeasures (PAT) become less effective, that is, there is a limited time period that dual adapted states can be retained.

Our general approach will be to develop a set of training devices, coupled with VET, that can provide astronauts with visual, inertial and somatosensory stimuli analogous to those associated with voluntary activity in normal gravity. The equivalent of an Earth gravitational force could be provided by a centrifuge and by elastic head and limb loading. Figure 1 depicts our initial concepts for a Virtual Environment - Centrifuge Device (VECD). The VECD requires the development of a short-arm centrifuge to provide a gravitational force stimulus to otolith and somatosensory graviceptors, and that is configured to permit head movements only in the plane of rotation to prevent semicircular canal "cross-coupling" effects. A head-mounted virtual environment system is incorporated to support the development of appropriate "calibration" between *visual* motion/orientation signals and the *inertial* motion/orientation signals generated by the centrifuge.

The anticipated benefits of inflight training using the VECD would be the maintenance of the "dual-adapted" state initially developed in the PAT by providing sensory signal matching, similar to the 1-g Earth environment between: (1) otolith (graviceptors) and semicirucular canal signals - to bias the otolith response toward the "tilt" interpretation (OTTR hypothesis), (2) visual, otolith and semicircuar canal signals to - recalibrate the VOR and (3) visual, vestibular and somatosensory signals.

SUMMARY AND CONCLUSIONS

Changes in the sensory stimulus conditions present in a given environment result in disturbances in eye movement control, posture and locomotion, and perception. People can adapt to altered sensory conditions which would result in the abatement of senory and sensorimotor disturbances. With appropriate training procedures and schedules, people can develop dual-adapted states and learn to switch rapidly between these states. However, retention time of dual-adapted states is limited without regular exposure to both sets of sensory conditions. Therefore, both PAT and INSTAR will be necessary for extended statys on-orbit, and VET is a critical component of devices being designed to adapt space travelers to microgravity and to maintain adaptation to Earth.

BIBLIOGRAPHY

Harm, DL, Parker, DE (1993). Preflight adaptation training for spatial orientation and space motion sickness. *Jrn. Clinical Pharmacology.* (in press)

Harm, D.L., Zografos, L.M., Skinner, N.C., and Parker, D.E. (1993). Changes in compensatory eye movements associated with simulated stimulus conditions of space flight. *Aviation, Space, and Environmental Medicine.* (In Press).

Carpenter-Smith, TR, Parker , DE (1992). The effects of unidirectional visual surround translation on detection of physical linear motion direction. In: B. Cohen, DL Tomko, F. Guedry (eds.) *Sensing and Controlling Motion, Vestibular and Sensorimotor Function.* Annals of the New York Academy of Sciences; 656: 817-819.

Parker, DE (1991). Human vestibular function and weightlessness. *Journal of Clinical Pharmacology, 31:* 904-910.

Parker, D.E., Reschke, M.F., Arrott, A.P., Homick, J.L. and Lichtenberg, B.K. (1985). Otolith tilt-translation reinterpretation following prolonged weightlessness: Implications for preflight training. *Aviat. Space Environ. Med., 56,* 601-606.

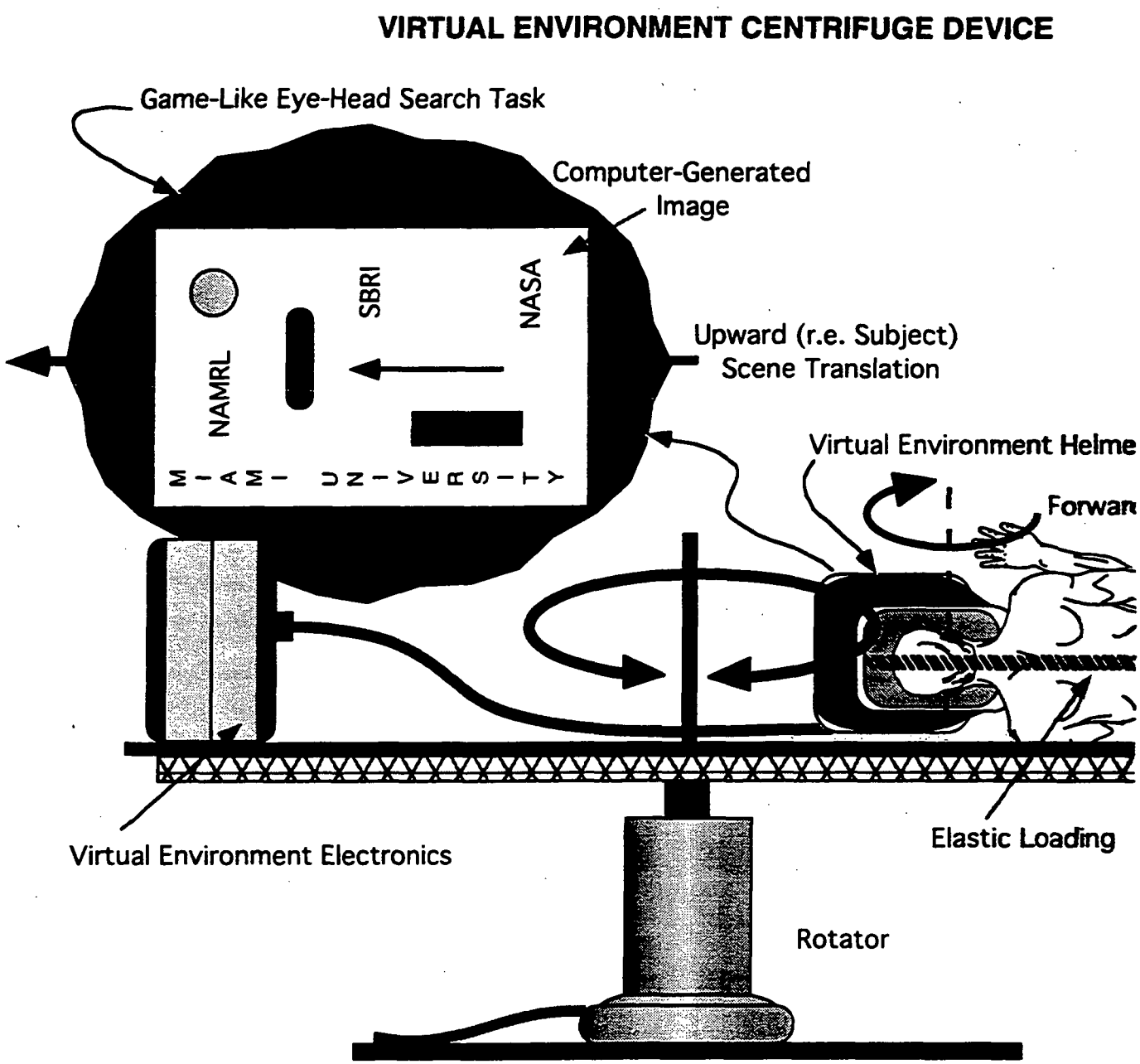


Figure 1. Conceptual Design for INSTAR Device

Ataxia and Other Posteffects of Flight Simulators: Should Virtual Reality Systems Have Warning Labels?'

Robert S. Kennedy, Kevin S. Berbaum, Lawrence Hettinger, and Michael Lilienthal
Essex Corporation
1040 Woodcock Road, Suite 227
Orlando, FL 32803

In the early days of simulation, if sickness or unsteadiness were reported, the problem was typically attributed to poor "fidelity" such as visual or inertial transport delays and asynchronies. It was thought that improved equipment would alleviate the problem; but instead, as fidelity improves, simulator sickness increases. This implies that emerging virtual environments will be prone to the same problem.

In order to maximize skill transfer, virtual reality interfaces are often built to resemble well-learned behaviors (e.g. moving a manipulator arm through the interface is like moving one's own arm). The same combination of like and unlike reality that seams to produce sickness and ataxia in simulators may produce similar problems for users of virtual reality systems.

It is proposed that liability may become a major concern in future simulators and VR systems as they become increasingly used by the private sector. Should such a system become implicated in an accident involving personal property damage, the VR system could be cited for "failure to warn."

Autonomously Acquiring Declarative and Procedural Domain Knowledge for ICAT Systems

Vincent J. Kovarik Jr.
Software Productivity Solutions
122 4th Avenue
Indialantic, FL 32903
vjk@sps.com

Abstract

The construction of Intelligent Computer Aided Training (ICAT) systems is critically dependent on the ability to define and encode knowledge. This knowledge engineering effort can be broadly divided into two categories domain knowledge and expert or task knowledge. Domain knowledge refers to the physical environment or system with which the expert interacts. Expert knowledge consists of the set of procedures and heuristics employed by the expert in performing their task. Both these areas are a significant bottleneck in the acquisition of knowledge for ICAT systems. This paper presents a research project in the area of autonomous knowledge acquisition using a passive observation concept. The system observes an expert and then generalizes the observations into production rules representing the domain expert's knowledge.

INTRODUCTION

Knowledge acquisition remains a bottleneck in the construction of expert systems. There have been a number of projects which have sought to automate the process of knowledge acquisition but have typically focused on the acquisition of knowledge as machine learning. From early systems such as AM [4] to more recent projects such as Cyc [6] investigating the construction of new knowledge has been based primarily on reasoning and discovery within the system.

Acquisition of knowledge for an ICAT system has a somewhat different perspective. Rather than covering a wide range of concepts, domain knowledge acquisition is concerned primarily with the capture and codification of the set of heuristics required for expert performance in that domain. While this does lead to system brittleness as problems are encountered which are on the fringe of the knowledge base or require commonsense reasoning based on real-world experiences or knowledge, it remains a proven and viable approach for special-purpose expert systems in the diagnosis, control, and procedural task oriented domains.

The focus of this effort has been on the capture and codification of knowledge relating to procedural and diagnostic tasks. These domains provide systems which are typically physical and, therefore, can be *instrumented* and the expert performs some *action* as part of performing their tasks. These two concepts, instrumentation and actions are key to enabling *i*Shadow to autonomously acquire knowledge.

Knowledge Acquisition Bottleneck

The knowledge acquisition process is a human intensive effort representing a serious impediment in the development of knowledge bases for expert systems and ICAT systems [5]. The knowledge engineer must work with the expert for extended periods of time. The domain of the expert must be learned by the knowledge engineer such that the knowledge engineer may encode the necessary domain information. Effective transfer of the acquired knowledge is a critical component [1] but continues to require significant human involvement.

The current methodology for acquiring knowledge requires the dedication of an individual to gather the knowledge from the expert and codify it into a set of rules or other form which can be interpreted by a machine. This involves conversations and interviews with the expert, observing the expert perform the task, eliciting

additional information or rationale for the behaviors observed, and then encoding the collected interviews, observations, rationale, and behaviors into a knowledge representation language.

Clearly this continues to be a significant bottleneck in constructing knowledge bases. It presents a significant problem for ICAT systems in NASA and the military because it is difficult to obtain uninterrupted time for the domain experts during which the knowledge engineer can extract the knowledge base. Yet, with downsizing of the military, limited government funds, and increased workload, more efficient methods must be found for extracting and encoding the knowledge applied by experts in performing their tasks.

Alternative Approaches

Other approaches to knowledge acquisition focus primarily on discovery [6] or pattern recognition in sets of data (i.e. database mining) [3]. These approaches rely on a closed world in the former and a broad base of facts in the latter. *i*Shadow follows the same general tactics as discovery-based learning in that it generalizes empirical observations to ascertain general domain rules. This approach relies on inductive reasoning and the related explanation-based reasoning.

Knowledge generation from databases typically applies abductive reasoning in attempting to develop knowledge which fits a particular set of data.

KNOWLEDGE CLASSIFICATION

Expert knowledge can be broadly classified into one of three categories:

1. Declarative
2. Process
3. Meta-Knowledge

These three categories reflect the domain in which the expert performs their task, the actual task actions, and the strategic knowledge applied to the performance of the task.

Declarative Knowledge

Declarative knowledge forms the basis of the domain in which the knowledge acquisition must be performed. This portion of the *i*Shadow knowledge base is developed using the object and relation editors defined within the *i*Shadow system. It is not necessary to build a "high-fidelity" model of the domain which is capable of simulation. Rather, a basic definition of the different classes of object in the domain and then the creation of instances of those classes within *i*Shadow that map to the external system.

The key is that *i*Shadow uses the states and relationships of the various objects in the domain to build a rule base. Representation of complex interactions and simulation is not needed to develop a set of rules.

Process Knowledge

Process knowledge is the set of observations collected by iShadow as it observes the expert performing their task. The process knowledge is divided into two categories, monitoring actions and state-change actions. Monitoring actions are those actions performed by the expert which does not alter the state of the domain or domain simulation. Examples of monitoring actions would be inspecting a gauge, observing the state of a switch, or querying the value of an ohm meter.

State-change actions are those actions performed by the expert which alter the state of the domain simulation or the domain itself. These might be opening or closing a valve, turning a rheostat, or flipping a switch on or off.

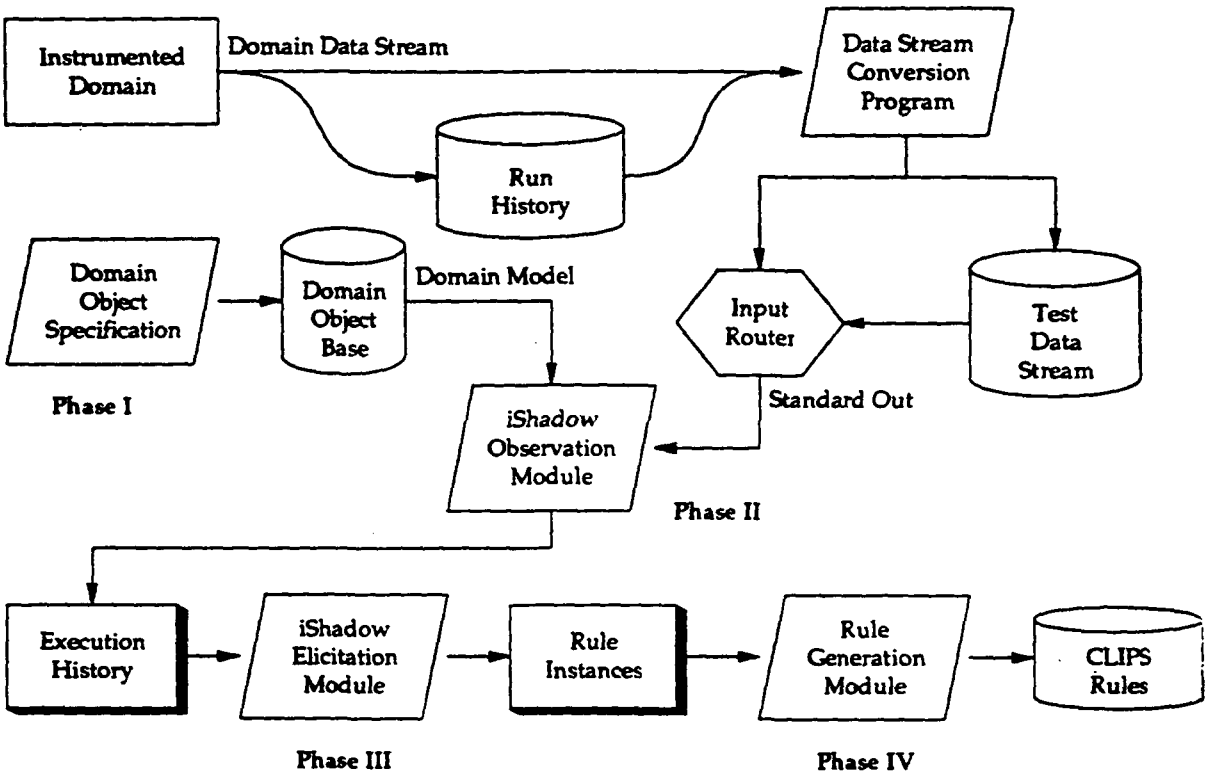


Figure 1: *i*Shadow Operational Flow

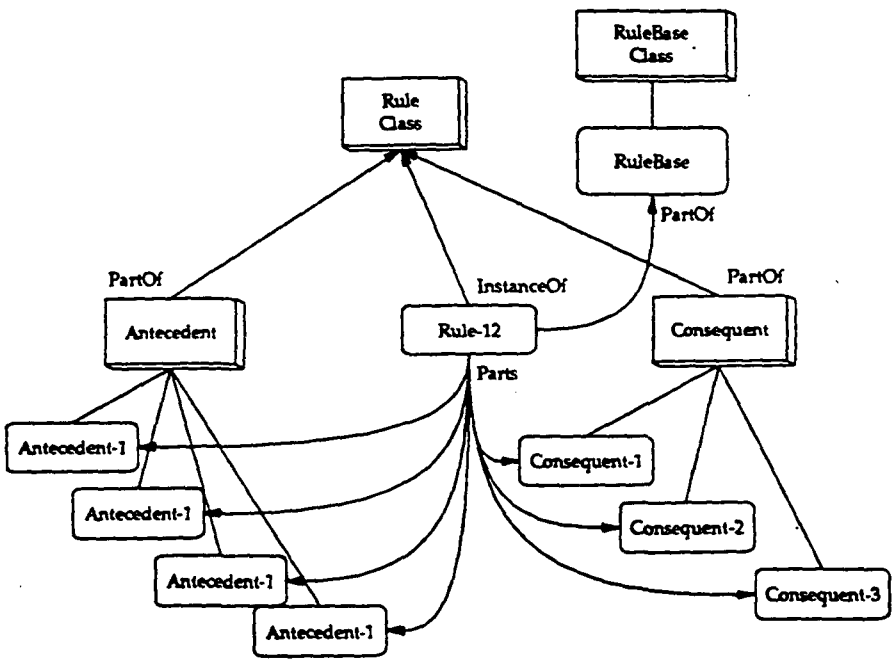


Figure 2: Organization of Rule Tuples in Object SimBioSys

Meta-Knowledge

Meta-knowledge refers to the set of heuristics applied by the expert which lead them to perform actions in a particular order. While *i*Shadow does not capture this class of knowledge automatically, it does elicit explanations and rationale from the expert regarding their selection of actions to perform

PASSIVE KNOWLEDGE ACQUISITION

The focus of this work has been in the development of a knowledge acquisition approach based on passive observation. Passive acquisition was chosen as the acquisition paradigm to emulate the observation process performed by a human knowledge engineer. The system is called *i*Shadow for an intelligent Shadow of the expert's actions.

Since the projects initiation, other efforts, such as [2], have also investigated the automated learning of diagnostic knowledge. These have focused on the automated acquisition of knowledge for a expert system as opposed to an ICAT system.

Project Goals

The goals of this effort have been refined as the project progressed and a better understanding of the problem and feasibility of the approach became more clear. At first our goal was to capture all classifications of knowledge but it soon became apparent that certain tasks were not easily captured in a passive scenario. Tasks such as solving differential equations, or psycho-motor tasks, such as controlling the remote manipulator arm of the Shuttle were difficult or impossible to instrument.

The focus of this effort has homed in on the generation of initial knowledge bases. The goal is to obtain approximately 75% of the initial knowledge base required for system operation. Coupled with the above coverage goal is the goal to achieve this initial set of knowledge in approximately 25% of the time it would take to build the knowledge base with a knowledge engineer.

Prerequisites to Passive Acquisition

The *i*Shadow approach, as mentioned previosly is not amenable to all types of knowledge acquisition. Knowledge domains which are highly cognitive in nature and those requiring psycho-motor skills, for example, are not suitable to this approach. The domain must consist of a combination of physical environment that can either be instrumented or simulated and a set of actions performed by the expert which can be observed through the system instrumentation

While these requisites limit the range of domains to iShadow can be applied, they also represent a significant portion of the types of domains and tasks for which ICAT systems are developed.

Four Phase Process

The automated acquisition process followed by iShadow is divided into four distinct phases, as illustrated in Figure 1. These are:

1. Domain Object Specification
2. Expert Observation
3. Elicitation and Rule Construction
4. CLIPS Rule Generation

Phase I: Domain Object Specification

In the first phase, a specification of the objects which comprise the expert's domain is developed. These objects are defined using the iShadow domain class editor. These classes define the generic set of entities and their states for the domain. Individual instances are then created to mirror the particular configuration of the system. Each configuration is called a model instance within *i*Shadow. Multiple models may be defined for a given domain.

Phase II: Expert Observation

In the second phase *i*Shadow connects to the instrumented system with which the expert interacts. As the expert performs activities in solving the domain problem, these are captured by the instrumentation and sent to the observation component. The activities captured are logged by *i*Shadow and form a temporal chain of events performed by the expert. Multiple observations may be captured for a single model.

Phase III: Elicitation and Rule Construction

After a set of observations have been performed, *i*Shadow then performs an elicitation and rule construction process. The elicitation consists of playing back the various sets of observations and querying the expert for rationale regarding their decision process, why certain actions were performed before others, and any strategy explanations they may have. These comments help to describe the meta-knowledge regarding the domain and the task ordering performed by the expert. *i*Shadow does not perform any generation of rules based on these elicitations but does embed them within the generated rules as comments. Thus enabling the knowledge engineer to add such strategic knowledge to the generated knowledge base.

The rule construction process begins by segregating the observed actions into sets of tuple pairs. These tuples consist of expert actions which represent the components of a rule. Figure 2 illustrates the organization and relationships between the various rule components.

The first element of the pair consists of observation tuples. Each observation action is encoded into an observation tuple. A set of these tuples is constructed until a state-change action is encountered. This set of observation tuples becomes the initial primitive set of antecedents for a rule.

iShadow then processes the state-change actions, collecting them into a set of action tuples until an observation action is encountered. The current set of action tuples is then paired with the current set of observation tuples to form a basic rule. This process continues until the end of the action list is encountered. If the end of the action list is encountered during a observation tuple set then iShadow disregards the current set of observation tuples being collected and stops processing. The basis for disregarding the observation set is that the previous tuple set had, in fact, been the end of the problem set and the last set of observations with no associated action were simply verification by the expert that the diagnosis and last action were indeed correct.

iShadow the adds to each set of observation tuples tests for the observed state of the objects references in the observation tuples and the current state of the objects referenced in the action tuples. This ensures that the rule will reflect the full set of states for all objects referenced within the rule.

The next step in the generation process takes the set of rule tuples and builds a symbol table consisting of all specific object references within the tuples. For example, the expert may have inspected gauge-1 and closed valve-2. While this level of specificity is appropriate for the selected model, the goal is to develop rules appropriate for the domain in general and not a specific model instantiation. Thus a symbol table is built and a single variable name is generated for all unique object names in the tuples. These variables names are then inserted into the tuples, replacing the specific object references.

Once the specific names have been replaced by variables, *i*Shadow then compares each rule in the generated rule base to each of the other rules. The process identifies those rules which are duplicates, have common antecedents or consequents. Duplicate rules are removed from the set. Rules with common antecedents but different consequents are merged and a notice to the developer is generated identifying the merged rule and the two original forms. Rules with different antecedents and common consequents are flagged for further inspection by

the developer as they represent some potentially serious gap in the knowledge base. Typically these last occurrences indicate some aspect of the model or domain has not been adequately captured and/or represented. For example, there may be some connectivity relationship which was not specified in the original model which reflects a distinct set of conditions.

Phase IV: CLIPS Rule Generation

The forth and final phase of the acquisition process is the generation of the CLIPS rule base from the internal rule tuples. This component has been intentionally designed as a modular, back-end process to allow for the generation of other rule syntaxes.

Implementation

*i*Shadow has been implemented on a SUN SPARC Station. It runs under either OpenWindows 3.0 or X Windows with OSF/Motif 1.1 or later. The system core was developed using SimBioSys, a object-oriented programming and knowledge representation system developed at SPS.

The current version uses the serial port of the SUN SPARC to provide an input channel for the observations generated via the instrumentation of the expert's domain. The link to the input port is purely a software link and the user can write any program required to provide input to iShadow.

To establish a communication link to the instrumentation, the user selects the Communication option in the Acquisition menu of the main window. This brings up a dialog box in which the user specifies any UNIX command which writes to standard output. The command is then executed in a background child process by *i*Shadow which also sets up a read pipe between the background child process and the main process. This enables *i*Shadow to simply read the output of the process.

CONCLUSIONS

The initial conclusions, based on observations of *i*Shadow in operation on simple domains is that *i*Shadow can generate a set of rules based solely on observations of the expert performing their tasks. A critical issue is the focus on procedural tasks, particularly in the diagnostic and troubleshooting area for physical systems and domains in the computer software realm. These domains provide opportunity for instrumentation which is critical to the success of the acquisition process.

More work is needed to refine the generation process. Initial work has been completed in the identification of duplicate rules, merge rules, and conflicting rules. More data needs to be gathered which can be used to determine the efficiency with which iShadow

Initial experiments have also been performed to test the feasibility of using iShadow to perform dynamic analysis of a rule base in action. Because the domain objects, expert actions, inference engine components and internal rule representations are all defined within SimBioSys, SimBioSys can control the execution of the rule set. This allows SimBioSys to perform single step execution of the rule base. Between each step, the conflict set and working memory can be examined and analyzed. This provides a significant capability not available in other expert system shells. It, in effect, provides a introspection of the behavior of the rule set at run time.

REFERENCES

[1] Gaines, B.R. and Shaw, M. L. G., *Eliciting Knowledge and Transferring It Effectively to a Knowledge-Based System,* IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 1, p. 4-14, February 1993.

[2] Giordana, A. et al, *ENIGMA: A System That Learns Diagnostic Knowledge,* IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 1, p. 15-28, February 1993.

[3] Han, J., Cai, Y., and Cercone, N., *Data-Driven Discovery of Quantitative Rules in Relational Databases,* IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 1, p. 4-14, February 1993.

[4] Lenat, D.B. and Guha, R.V., *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*, Addison-Wesley Publishing Company, Inc., 1990.

[5] Lenat, D. B., Prakash, M., and Shepherd, M., *CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks*, AI Magazine, 6, 4, Winter, 1985.

[6] Lenat, D. B., AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search, Rep. No. STAN-CS-76-570, Computer Science Dept., Stanford University.

Automating Knowledge Acquisition Processes: An Analysis of the Psychological and Technological Issues Surrounding the Automated Elicitation of Expert Knowledge'

Will Lidwell (Nimbus Technologies) and David B. Palumbo (UHCL)
Nimbus Technologies, Inc.
15407 Lanswick Drive
Houston, TX 77062
713-286-2230

If ICAT and related intelligent systems are to realize their potential as practical instructional environments, the need for the efficient and accurate elicitation of expert mental models is paramount. Although significant advances in knowledge acquisition have been accomplished through the development of cognitive theory-based task analysis methodologies, a number of theoretical and practical problems remain: e.g., adequate logical representations of declarative and strategic knowledge, maximization of expert recall and verbalization through domain situation and/or priming; conscious versus subconscious assignments of semantic relationships, etc. These issues are, of themselves, theoretically formidable. However, they are often overshadowed in importance by more practical issues such as subject matter expert availability, verbal skills of experts, etc.

In an attempt to at once address the psychological and practical aspects of knowledge acquisition, many have attempted to develop highly-interactive computer-based knowledge acquisition environments which empower experts to communicate their knowledge in multiple ways without the aid of an elicitor. This paper will present a detailed psychological and technological analysis of a number of existing knowledge acquisition systems and the KA processes which they employ. Further, it will provide proposed solutions and an empirical research agenda within the context of a prototypical system: the Knowledge Interaction Toolkit (KIT).

Beyond Knowledge Acquisition - A Blueprint for Knowledge Construction

David Germain, Bob McNenny, and David B. Palumbo (UH-CL)
McDonnell Douglas Aerospace
13100 Space Center Blvd.
Houston, TX 77059
(713) 280-1527 (713) 280-1689
germain@sweetpea.jsc.nasa.gov

Knowledge acquisition(KA) is an integral part of the human experience. From childhood development through formal schooling to on-the-job training, all forms of learning are in essence a knowledge acquisition process. Formal KA can be defined as the identification and structuring of expert knowledge for use in advisory expert systems, performance support tools, and classroom or computer-based training programs. But, extracting knowledge from an expert is a difficult and challenging task.

Expert knowledge includes domain knowledge, procedural knowledge, and strategic knowledge. Unfortunately, most current KA methodologies only capture a distilled version of the expert's procedural knowledge. Most of the domain and strategic knowledge is lost along with the experiential context and supporting explanations.

If the ultimate goal of the KA process is the development of a training program, the resulting knowledge should be explicit, inspectable, executable and learnable. This is the major difficulty. Good pedagogic theory tells us that for knowledge to be learnable, it must be: chunked, sequenced, situated, interactive, and related to previous experience. However, the refined results of most KA methodologies lack the context and supporting explanations necessary for the knowledge to be learnable.

Building upon the constructivist learning theories of Papert, and Piaget and the work of Kolodner and Schank in reasoning and memory, MDC is side-stepping the pitfalls of conventional KA and exploring an exciting, interactive case-based approach which captures, not the compiled heuristics of the expert, but real, situated examples of expert performance which will enable students to construct their own knowledge through observation, reflection, and experimentation.

Robo-Cat--An Intelligent Robotics Trainer

Debra Bettis, Stephen Desrosiers, David Mulcihy, and Ken Ruta
McDonnell Douglas Aerospace
13100 Space Center Blvd.
Houston, TX 77059
(713) 280-1536 (713) 280-1689
desrosie@sweetpea.jsc.nasa.gov

Currently, expensive robotics simulators are used for training of the space shuttle astronauts. Robotics instructors provide one-to-one coaching and evaluation to help the astronauts improve their skills as much as possible in the limited time available. Because of the combined cost of the simulators, the instructors, and the facility, robotics training is very expensive. Even so, access to robotics trainers is limited by availability. As NASA prepares for the construction phase of Space Station Freedom in March of 1996, the problem is becoming critical.

In order to reduce the cost of robotics training, to provide more training time, and to provide on-board refresher training, less expensive, instructor-less robotics trainers must be developed. However, intelligent robotics trainers present a variety of unique challenges.

McDonnell Douglas is working to develop a cost effective robotics trainer which will provide adequate flexibility and realism to allow the user to practice a variety of Space Station Freedom assembly tasks.

Using a kinematics simulation, Robo-CAT monitors the movement of the RMS and the user's selection of camera views, providing varying levels of coaching to guide the user. Robo-CAT also provides feedback on the effectiveness of the user's actions as compared to those of more experienced operators and the user's own past performance.

An Intelligent Computer-Aided Tutoring System For Diagnosing Anomalies Of Spacecraft In Operation

Mark Rolincik, Michael Lauriente, Harry C. Koons, and David Gorney
NASA Goddard Space Flight Center
Code 420
Greenbelt, MD 20771
301-286-5690 301-286-5198
ENVNETonNSI/DECNET128.183.104.16onNSI/IP

A new rule-based, expert system for diagnosing spacecraft anomalies is under development. The knowledge base consists of over two-hundred (200) rules and provides links to historical and environmental databases. Environmental causes considered are bulk charging, single event upsets (SEU), surface charging, and total radiation dose. The system's driver translates forward chaining rules into a backward chaining sequence, prompting the user for information pertinent to the causes considered. When the user selects the novice mode, the system automatically gives detailed explanations and descriptions of terms and reasoning as the session progresses, in a sense teaching the user. As such it is an effective tutoring tool. The use of heuristics frees the user from searching through large amounts of irrelevant information and allows the user to input partial information (varying degrees of confidence in an answer) or 'unknown' to any question. The system is available on-line and uses C Language Integrated Production System (CLIPS), an expert shell developed by the NASA Johnson Space Center AI Laboratory in Houston.

Training Augmentation Device For The Air Force Satellite Control Network

Captain Keith B. Shoates, USAF
Air Force Institute of Technology/Education With Industry
Wright-Patterson AFB, OH 45433-6583

BACKGROUND

From the 1960s and into the early 1980s satellite operations and control were conducted by Air Force Systems Command (AFSC), now Air Force Materiel Command (AFMC), out of the Satellite Control Facility at Onizuka AFB, CA. AFSC was responsible for acquiring satellite command and control systems and conducting routine satellite operations. The daily operations, consisting of satellite health and status contacts and station keeping activities, were performed for AFSC by a Mission Control Team (MCT) staffed by civilian contractors who were responsible for providing their own technically "qualified" personnel as satellite operators. An MCT consists of five positions: mission planner, ground controller, planner analyst, orbit analyst, and ranger controller. Most of the training consisted of On-the-Job-Training (OJT) with junior personnel apprenticed to senior personnel until they could demonstrate job proficiency. With most of the satellite operators having 15 to 25 years of experience, there was minimal risk to the mission.

In the mid 1980s Air Force Space Command (AFSPACECOM) assumed operational responsibility for a newly established control node at Falcon AFB (FAFB) in CO. The satellites and ground system program offices (SPOs) are organized under AFSC's Space and Missile Systems Center (SMC) to function as a systems engineering and acquisition agency for AFSPACECOM. The collection of the satellite control nodes, ground tracking stations, computer processing equipment, and connecting communications links is referred to as the Air Force Satellite Control Network (AFSCN).

Unlike AFSC's practice of staffing their MCT with contractors, AFSPACECOM's concept of operations is based on Air Force officers serving as the MCT. Furthermore, AFSPACECOM has started transitioning satellite operations to Noncommissioned Officers (NCOs). For routine satellite operations, a single Air Force officer will serve as space operations crew commander and oversee the activities of several NCO satellite controllers. Because of the frequent turnover of military personnel, retaining trained crews for AFSPACECOM presents some unique challenges. Initial training and satellite controller position certification became critical issues to AFSPACECOM. The training pipeline for AFSPACECOM consists of 12 to 18 months of Formal Undergraduate Space Training (UST), 1-3 months Initial Qualification Training (IQT) on the satellite (e.g. Global Positioning System (GPS), Defense Support Program (DSP)) and its command and control system, and 30 to 45 days of control center specific training. After completion of training, a satellite operator would have only two years to at best two years and ten months left on station. Upon completion of their training the students would be qualified crew members ready to perform their duties in their Satellite Operations Center (formerly known as the Mission Control Complex (MCC) and Test Support Complex (TSC)).

GENESIS OF SATELLITE OPERATIONS CREW TRAINERS

In the early 1980s AFSC initiated development of satellite crew trainers to support Air Force satellite launch and on-orbit operations. Provisions were also made to interface with NASA to support Air Force/NASA shuttle launches and missions. Prior to this time, the standard training method for crew training was to conduct training exercise on their operational equipment using a "paper simulation" method. Team members were given paper scripts during a training exercise detailing scenarios to which they were to respond (some scripts would provide the operator with failed, telemetry values); the operator's appropriate response(s) would then be evaluated. The "paper simulation" method was laborious to develop and time consuming to conduct for training large groups of teams coordinating over multiple voice-nets and consoles in support of a launch exercise. Unlike the "paper simulation" method, crew trainers provide Air Force satellite control teams with a simulated satellite telemetry

stream that could be processed and displayed on the crew's command and control system in a dynamic and interactive manner.

Telemetry Simulation System. The Inertial Upper Stage (IUS) Satellite Operations Center (SOC) at OAFB became the first program to benefit from the development of a crew trainer, the Telemetry Simulation System (TSS). The system was developed by the Unisys Corporation (now PARAMAX Systems Corporation), designed using 8086 technology, and required a significant amount of program unique software development. Although initially fielded in 1984 for the IUS program, several other satellite application models were developed over the succeeding five years. An upgrade to the TSS hosted on 80286 technology was installed at both AFSCN satellite control nodes, OAFB and FAFB.

The TSS was the first Air Force trainer developed for space operations, which supported integrated training between the various positions of the SOC crew, the communications segment, and the Resource Control Complex (RCC). The TSS had several shortcomings. Each satellite software model was mission unique with very little reusable software code between programs. Model development time was from two to three years. Only one satellite model could be running at a given time on the TSS. The cost for satellite model development was approximately $5M to $7M per satellite application, $1.2 M for the hardware platform, and $3M per installation due to the hardware architecture and Air Force facilities and security requirements. The TSS did not simulate Air Force ground tracking stations; this limitation made the trainer inadequate for two of the five members of the satellite control crew (an orbit analyst who relied on satellite tracking data and a ground controller who needed tracking station configuration status information). The TSS software was written in PLM and Intel assembler. The code was costly to maintain and was not readily portable to other hardware platforms. Increased training and fidelity requirements exceeded TSS hardware architecture capabilities.

Generic Telemetry Simulator (GTSim). As the development on the TSS was reaching its apex in the middle 1980s, NASA through a contract with Singer Link (now CAE - Link) began developing the GTSim for the Centaur program. Following the Challenger accident, AFSC completed the development of the GTSim to provide the reusable, reconfigurable training platform to overcome the shortcomings of the TSS. An application for the Defense Satellite Communications Satellite (DSCS) was developed by General Electric under contract with the DSCS SPO and deployed in 1992.

The GTSim again proved the importance of computer simulation to mission success. However, the GTSim did not live up to expectations. Development cost for the satellite model were approximately $6 M ($1 M over initial projections because very little of the Centaur software could be reused). Installation cost due to unique GTSim hardware architecture, and Air Force facility and security requirements were in excess of $2.5 M. The software development time increased from one to nearly two years. Like the TSS, there was no ground tracking station model in the GTSim.

THE NEXT GENERATION

Satellite Control Simulation Study. In the latter part of the 1980s the Satellite Control and Data Handling SPO (SMC/CW) sponsored a study (Satellite Control Simulation Systems Study) exploring areas in which simulation and modeling technology could facilitate Air Force satellite control operations. The study concluded that several of the capabilities required for simulation and operations could share the same software modules if designed properly. For instance a satellite model and supporting orbit propagation models were needed to support mission team training, plan validation, and anomaly analysis aids. Software reuse, evolutionary acquisition procedures, extensive use of Commercial-Of-The-Shelf (COTS) hardware and software, and a common space vehicle application user interface were some of the study's recommendations. For satellite crew trainers the major recommendations were to maximize reuse of software simulation models, provide a standard trainer architecture for hosting various satellite models, employ rapid satellite modeling development tools to meet mission objectives, provide instructional control over telemetry values and training session, and support multiple contacts. This study set the course for the development of the next generation crew trainer.

Training Systems Working Group. A Training Systems Working Group (TSWG) was established within the Air Force Satellite Control community to specify trainer requirements and coordinate activities to promote the development of a standard Air Force satellite crew trainer. Its membership was drawn from HQ AFSPACECOM, both Air Force satellite control nodes including operators and training personnel, and the Air Force satellite ground system and satellite acquisition agencies. The TSWG supported the conclusions of the Satellite Control Simulation Systems Study and developed a comprehensive set of operational requirements reflected in an AFSPACECOM Statement of Operational Need (SON 012-88, Aug. 90).

Training Augmentation Device. The opportunity to meet AFSPACECOM's SON arose when the U.S. Navy and the Air Force jointly funded the Training Augmentation Device (TAD) to provide satellite crew training for Air Force SOC crew responsible for operating the Ultra High Frequency Follow-On (UHF F/O) satellite. The requirements for the TAD were specified in the 50th SpaceWing Technical Requirements Document (Feb. 91), and were translated into system requirements by SMC/CWD (Aug. 91). These requirements focused on providing a crew trainer that could simulate the UHF F/O satellite, certain satellite anomalies, and Air Force and Navy satellite tracking stations. PARAMAX Systems Corporation was selected to develop the TAD architecture based on their in-house prototyping efforts. PARAMAX subcontracted to Hughes Aircraft Company, the UHF F/O satellite contractor, to develop the satellite model for the TAD.

The development strategy for the TAD was based on an evolutionary development concept. While TAD requirements were fully flushed out by the user community in the initial stages, design and development would occur incrementally. A system design covering the overall software and hardware architecture was scheduled to provide the Air Force community insight into the TAD's architecture. Instead of a single massive software detailed design, a series of small detailed design meetings were held. The above "evolutionary development" approach was initiated to preclude rigidity in specifying a grand design that was likely to be plagued with innumerable changes when the actual design implementation occurred. Several critical program milestones were established that enabled the contractor to incrementally design and build the TAD architecture. This approach enabled the Air Force community to have significant insight and input into the TAD's architecture throughout its development cycle. Installation of TAD hardware at FAFB, completion of TAD's basic common software models and delivery of TAD's UHF satellite models were designated as major program milestones. TSWG team involvement was maintained throughout the course of the contract. TSWG members participated in the design of the TAD's operator interface and satellite component's building block paradigm. This insured user involvement in the TAD's design throughout its development.

SYSTEM DESIGN OVERVIEW

The primary purpose of the TAD is to provide satellite operations crew readiness training for launch, early-orbit, and on-orbit operations in their SOC. The TAD dynamically simulates the external environment to the SOC including the ground tracking stations and the satellite. The TAD permits the SOC crew to verify new satellite command(s) or command sequences prior to their transmission to a satellite by comparing the satellite's simulated response to the predicted response. The TAD supports off-line analysis of satellite anomalies, work around procedure, development, and mission planning. TAD satellite subsystem models permit the insertion of anomalous states permitting the user to match the actual anomalous condition aboard the satellite. Satellite workaround procedures can then be developed by SOC crews and verified against the TAD prior to their transmission to the satellite.

System Design. The TAD's system design is based on the "open systems" architecture approach. The software can operate in a single environment (e.g., workstation) or take advantage of a symmetrical multiprocessor environment. This scalability permits the TAD to address simulation applications that support 1 to N simulated objects. Symmetrical multiprocessing using portable parallel processing application software was selected to meet Air Force crew trainer growth and performance requirements. Symmetrical multiprocessing under a UNIX operating system was selected to meet the TAD's performance requirements. The TAD provides classified (red) and unclassified (black) data separation. Due to limited available floor space within the SOC, the TAD must be located outside the SOC. The TAD is connected to the SOC via the facilities communication segment.

Hardware Design. Except for a Multibus I interface to support unique AFSCN interfaces, all of the TAD's hardware is composed of COTS equipment. The TAD is hosted on two interconnected Unisys U6085 minicomputers. One minicomputer runs the ground tracking station network models, while the other minicomputer runs the satellite models. The hardware is based on an open systems architecture which permits heterogeneous software, scalability of processors, dynamic processor loading and interoperability. TAD's software language, operating system, display software file systems, computer interfaces and databases have been ported to other UNIX operating systems (i.e., Sun, HP and IBM). The U6085 is a parallel processor with four dual 80486/25 MHz boards capable of 11 Million Instructions Per Second (MIPS). The U6085 provides upward scalability to a total of fifteen dual boards per U6085. Each U6085 contains over 1 Giga Byte (GB) removable hard drive for storing program unique applications and running in a classified mode.

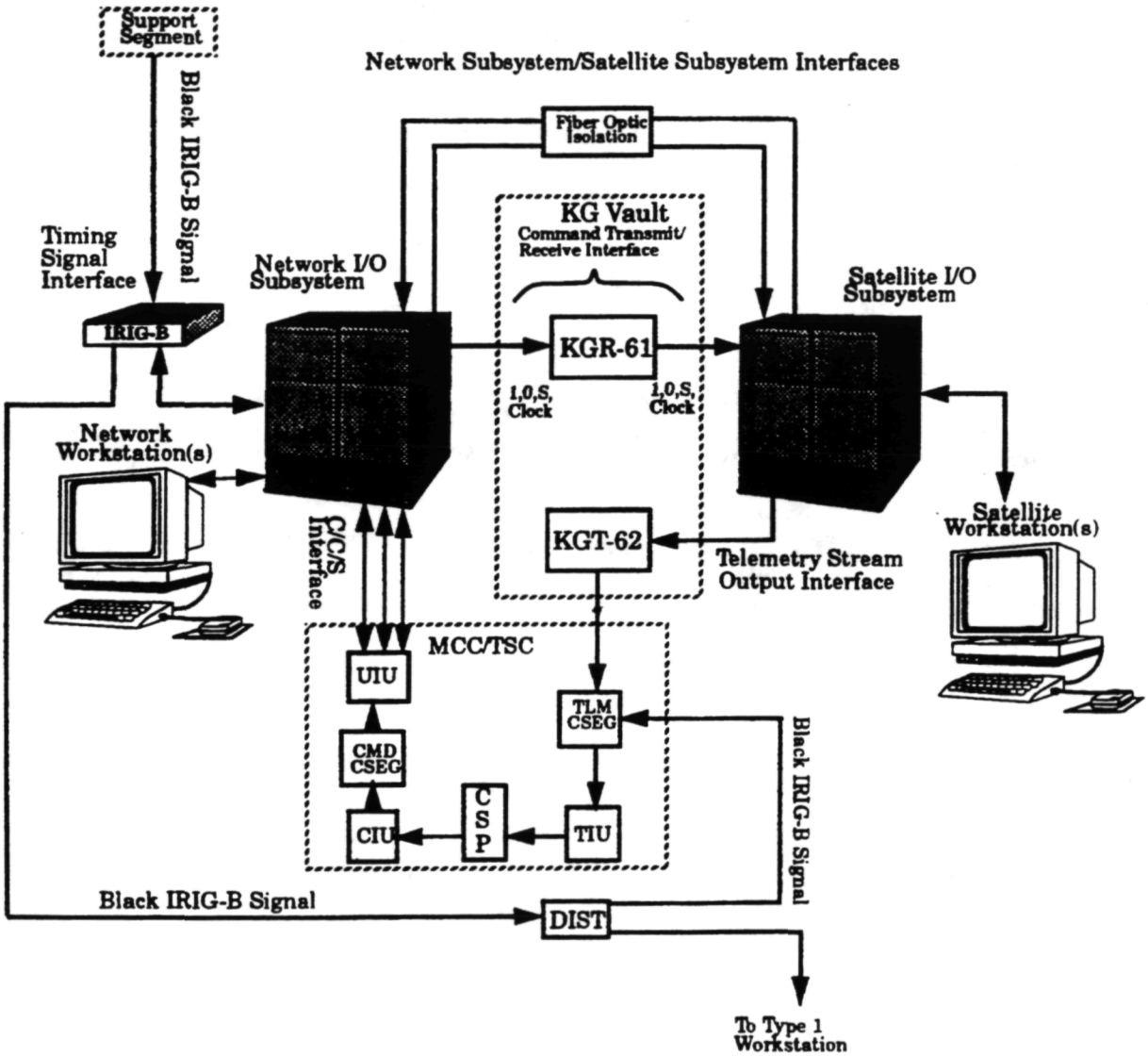


Figure 1. UTAD Hardware Design.

Connected to each U6085 minicomputer (satellite and network) is an Operator Interface Station (OIS) consisting of an IBM RS 6000/520 workstation that provides the human interface. This workstation allows real time monitoring of selected data points, injecting anomalies, manipulating simulation models, and changing simulation speed. Each workstation contains over 1 GB removable hard disk which contains any program specific data, training scripts or classified data. Display of selected satellite telemetry and student interactions is

via IBM GT4X 24-bit graphics card. A high speed printer is connected to each RS 6000 to print training data as necessary. Connectivity between the U6085s, Multibus I and the RS 6000s is via fiber optics or TCP/IP Ethernet LAN.

The TAD's Multibus I interface is used to connect to the SOC's AFSCN custom data interfaces. The Multibus I is composed of commercially available Single Board Computer (SBC) printed circuit boards. A custom board that generates TAD satellite telemetry attaches through a standard connector to a commercial Multibus SBC printed circuit board. It supports a variety of telemetry formats and telemetry data rates up to 2.5 Mbps. The telemetry generator is the only custom piece of hardware in the TAD's architecture. Data rates are programmable and encoding schemes are selectable. Telemetry commutation is performed on the host programmable SBC board. A Loral NASA Command Formatter (NCF) board resides on the Multibus I and serves as the command ternary receive port for SOC satellite commands into the TAD. Six different telemetry boards can be connected to the Multibus I, which could be driven by six separate simulations at the same time, thus supporting six different SOC training crews.

Software Design. The software design was predicated on the use of the Ada programming language and whenever feasible the use of COTS software to reduce development cost. Use of Ada programming language as the main software language was mandatory per DOD policy and Air Force policy. Use of other languages in the TAD are limited to special functions or routines such as existing I/O driver software, revised satellite models, interfaces to UNIX or COTS software, and fourth generation language (4GL) with Structured Query Language (SQL) database code. Two different operating systems support the TAD: U6085 Unisys System V Operating System, and the IBM RS 6000 AIX Operating System. Both systems support a security rating of C2 and provide discretionary access control, user identification, passwords, and audit trails. The TAD workstation employs X-Windows X.11 standard to establish its windowing environment. The user can open several display windows on his work station and monitor the status of several on-going activities. X-Windows standardize the manner in which a user can open or close display windows permitting increased operator productivity. The workstation's human interface is developed using the Dataviews tool from Visual Intelligence, Inc. Dataviews supports X-Windows.

Database development is supported by the Progress tool from Progress, Inc. The 4GL that is provided by Progress inherently supports concurrency control, recovery, import/export of data. The Progress 4GL Structured Query Language (SQL) standard is also used to build user data displays and report requests.

TAD local area network uses a standard IEEE 802.3 Ethernet protocol supporting the Network File System (NFS) and Remote Procedure Call (RPC) open standards sponsored by Sun Microsystems, Inc.

On-line documentation access to the TAD's user manual and another documentation is supported by the desk top publishing tool Framemaker, which uses quick look-up hypertext links provided by Frame Technology Inc.

Telemetry commutation software using PLM software developed for the TSS was adapted into the TAD architecture along with the NASA Command Formatter software. Figure 2 identifies the various commercial languages by percentage of total lines of code (573.5 KSLOCs) used in the TAD.

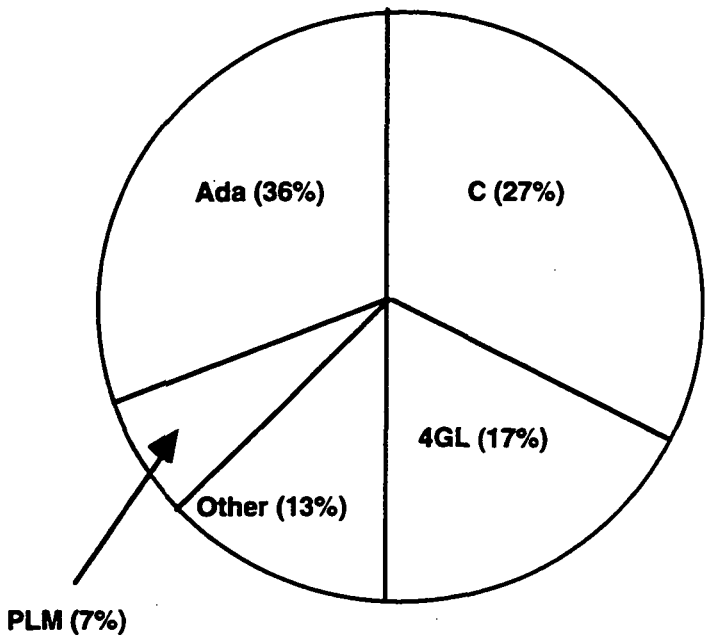


Figure 2. TAD Software Programming Languages.

Software Architecture. The software is divided into two primary Computer Software Configuration Items (CSCI). The TAD's Common User Software (CUS) CSCI includes all those functions which are common among satellites, while a separate Mission Unique Software (MUS) CSCI is required for each satellite application model, only if a high fidelity capability is required. The CUS models support rapid scripting of TAD simulation models, scenario generation and operates with the various COTS software to provide data to the TAD operator workstations along with executing satellite and space environment models during simulation run time to support telemetry generation and dynamic responses to crew initiated commands. Figure 3 identifies the TAD's Architecture.

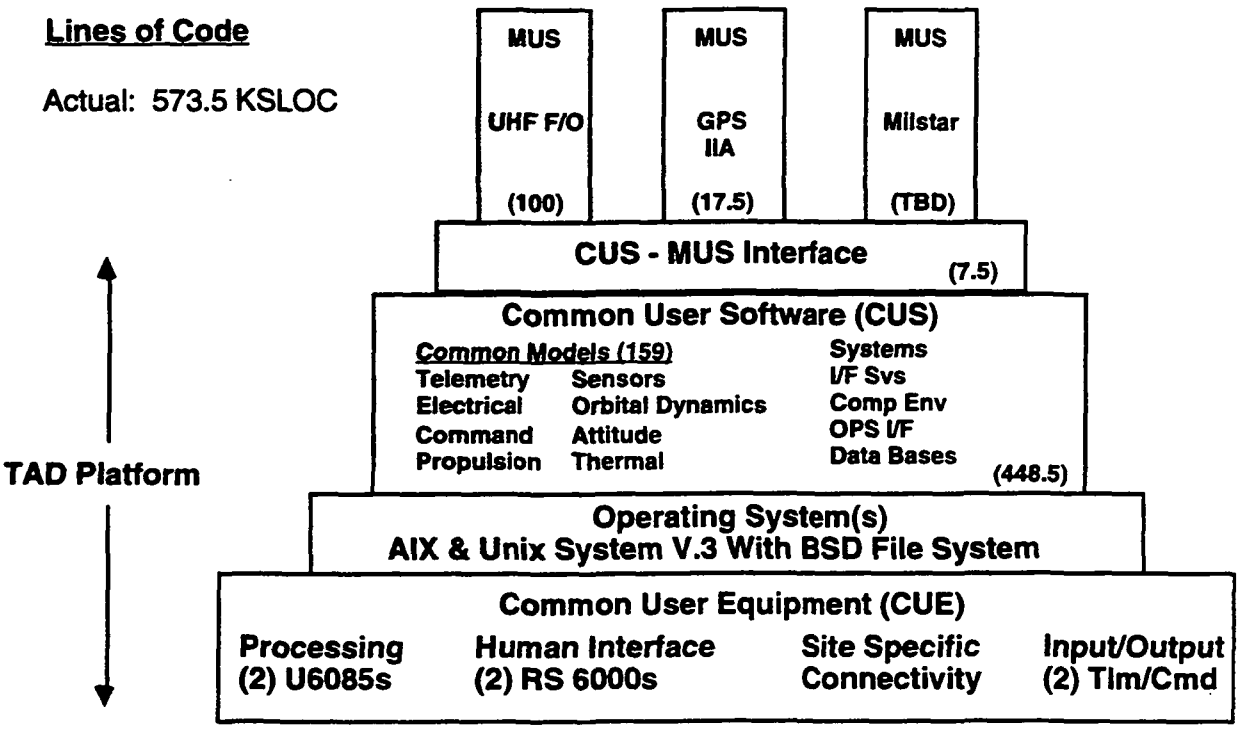


Figure 3. TAD Architecture.

Common User Software. The TAD CUS employs Ada tasking model constraints which permit the spawning of multiple Ada tasks, each representing a simulation unit (e.g., satellite and tracking stations). When a parallel processor environment is present with a parallel Ada run time system, the simulation units takes advantage of the additional processors. The TAD's common user software (CUS) consists of a collection of reusable or common models such as satellite commanding, orbit line of sight, telemetry, orbit dynamics, satellite attitude, satellite sensors, actuators, orbit line-of-sight, propulsion electrical power, telemetry, tracking and control, and Connectivity models. Figure 4 identifies the TAD's Computer Software Components. A control system based block programming paradigm is also provided which permits training personnel to rapidly develop medium complexity satellite simulations.

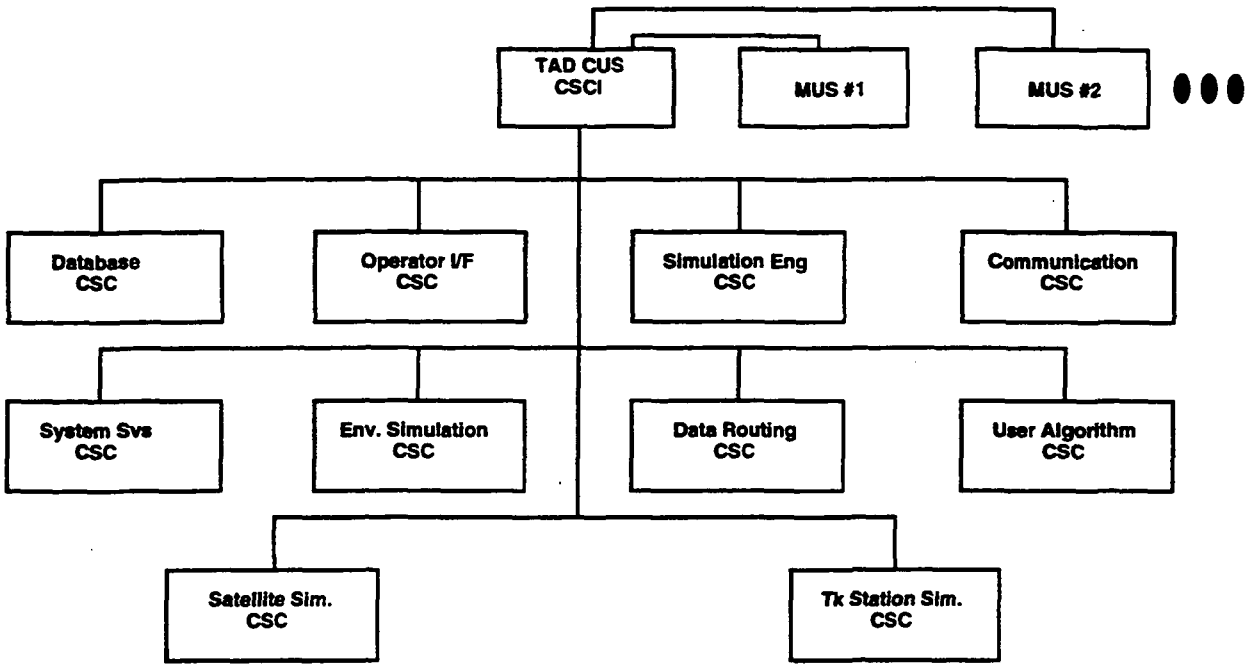


Figure 4. TAD Computer Software Components (CSCs).

The TAD CUS provides the software environment in which all TAD simulation activities take place. A broad suite of simulation tools is provided by the CUS such as scripting, logging, script generation from logs, SOC interface, simulation control, fast/slow forward time control, common satellite and sub component models, simulation state save/restart, multiple simulated time/spatial universes, multiple simulated objects, and evaluation utilities and reports. In addition to these features, the TAD CUS provides reconfigurable common math/logic models whose behavior is defined through database parameters. The CUS also provides a block programming capability that supports modeling of subsystems which have a high degree of variability that cannot effectively be handled by the common models.

Database. The TAD database CSC provides the means by which an operator can enter new data, change existing data, or delete data from the MUS database. The MUS database is the data source for tailoring the behavior of the simulation environment and the objects which are within the environment. These functions are accomplished by Database application code, database load, and the Progress Database development tool modules.

The Database module provides database management, environment definition, satellite definition, tracking station definition, and software discrepancy report/definition. The database manager provides templates from which the operator may create new databases which are automatically cataloged. These templates may be created for a family of satellites which are similar which are then tailored for specific mission requirements. The software discrepancy report/definition function provides on-line identification of hardware or software problems.

The Database Load module provides a means of uploading actual SOC databases used in the operational command and control system directly into the TAD databases.

The Progress Database development tool, a COTS 4GL client/server product, is used to develop and maintain TAD CUS database application code.

Operator Interface. The operator interface CSC handles all of the processing required to present information to and receive directions from the trainer during execution of a training session. These functions are accomplished by the Human Interface, Display Processing, Scripting, and DataViews Tool modules.

The Human Interface module creates a main operations control window which is used by the trainer to initiate a training session. Each Operator Instructor Station (OIS) hosts exactly one Human Interface process which maintains separate control for each environment and simulated unit that is controlled by that OIS. The Human Interface takes advantage of a graphical user interface features such as windowing, mouse-and-menu control, and color graphics.

The Display Processing module is used by the Human Interface module to provide processing of 2-D graphics displays. The Scripting module processes any scripts that have been defined for the training session. The DataViews Tool module supports the custom development and maintenance of 2-D graphics displays.

Simulation Engine. The simulation Engine CSC provides the run-time control functions on the compute engine (i.e., U6085 minicomputer) which are accomplished by Simulation Engine Control and Ada Task Control modules. The Simulation Engine Control module initiates the processes that execute on the minicomputer, establishes communication with the OIS, and starts the processes that provide communication with the AFSCN interface software components.

The Ada Task Control module provides control over several module such as the Environment Simulation, the Satellite Simulation, and the Tracking Station. It is started by the Simulation Engine Control module when a simulated space/time environment is requested.

Communications. The communications CSC provides the communications interface services between all the hardware and software components, regardless of whether they are on the U6085 minicomputer or the RS 6000 workstation. These functions are accomplished by Ada Communications, Multibus I Communications, and Intersubsystem Communications modules.

The Ada Communications module provides communications between Ada tasks running on the U6085 and UNIX processes residing on the RS 6000. The Multibus I Communications module provides communication between the TAD system and the Air Force communications switch interface to the SOC. The Intersubsystem Communications module provides communication between the U6085 satellite and network subsystems.

System Services. The system services CSC provides communications, data routing and software utility support for TAD software components. These components include system control processes, human interface and display handling processes, Multibus I input/output processes, user defined algorithms, space/time environment models, and satellite and network subsystem models. These functions are accomplished by Time Control, Physical Environment, and Environment Communications modules. The Time Control module provides the OIS control over the TAD's run time environment, such as fast/slow forward, freeze, state save, start/stop. The Physical Environment module provides control of the simulation of the necessary physical characteristics of space surrounding the earth (e.g., sun, moon and planets). The Environment Communication module provides control over communication between objects in the same or different space/time environments.

Environment Simulation. The environment simulation CSC provides the simulation of space/time environment in which the satellite and ground tracking stations operate. The simulation of an environment includes: modeling of the passage of time from a specified starting point and the modeling of the necessary physical characteristics of space surrounding the earth.

Data Routing. The data routing CSC provides the transparent exchange of data between TAD components. Each simulated unit possess a data pool which has been generated by the Simulation module (Environment, Satellite, or Tracking Station) for that unit. Data in an objects data pool can be accessed by any of the TAD's components. It is this loosely coupled nature of the system that provides the flexibility to turn on/turn off individual models and update values in a data pool from alternate means (databases, look-up tables, external sources).

User-Defined Algorithm (UDA). The user-defined algorithm CSC provides the mechanism for the trainer to manipulate data during the execution of a training session. UDAs may be used to generate telemetry measurands, manipulate data for display purposes, or generate specific simulation data. UDAs may interact with other algorithms and TAD models via the data pools.

Satellite Simulation. The satellite simulation CSC provides satellite simulations capable of receiving space vehicle commands from a SOC and producing dynamic telemetry output which reflects ongoing evolution of the simulation environment and the simulated satellite's response to the received commands. The simulation of a satellite includes the modeling of the health and status of various subsystems. Satellite modeling employs common math/logic models that obtain satellite-specific characteristics from parameters specified in databases. The TAD common satellite subsystem models include: Orbital Dynamics, Attitude, Sensors, Actuators, Propulsion, Electrical Power Subsystem, Commanding, Telemetry Processing, Eclipse, Telemetry and Commanding Control and Status (TCCS), and a Common Spacecraft Processor Model. These models communicate primarily through Data Routing.

Tracking Station Simulation. The tracking station simulation CSC provides for the simulation of both Air Force and Navy satellite tracking stations. These simulations respond to configuration and control directives from the SOC and return realistic tracking and status information. This module also provides modeling of core equipment and antenna subsystems of the tracking stations. Like the satellite simulation module, this module employs common math/logic models that define tracking station-specific characteristics from parameters (tracking station coordinates such as longitude, latitude, altitude and obscura data) specified in databases. The TAD common tracking station models include: Tracking and Antenna, Command Generation, Tracking Station Control and Status (TSCS), and Tracking Station Equipment.

Mission Unique Software. The TAD's Mission Unique Software (MUS) constitutes the specific satellite application model (e.g., GPS or UHF F/O). For instance GPS Block IIA software running on the TSS at FAFB and totaling 45 KSLOCs of FORTRAN 77 is being transcribed into Ada for use on the TAD. It is scheduled to be completed by the Spring of 1994. The total estimate Ada lines is 17.5 KSLOCs. The reason for the significant reduction in lines of code is partially due to the nature of Ada being a high order language, but mainly due to the ability of TAD's common models to supplant significant portions of the original FORTRAN code. In the case of the UHF F/O satellite model developed by Hughes Aircraft Company, a custom C code to Ada software interface was developed. This interface permits the Hughes model (previously developed) to run in the TAD's software environment. The Hughes UHF F/O MUS model consists 100 KSLOCs of C code and is a modification of a previously developed satellite simulation for the Hughes HS601 satellite family.

PROGRAM STATUS

The TAD development is currently completing development. The TAD hardware and an initial software capability was installed in the summer of 1992 at FAFB. A successful demonstration of generating a telemetry wavetrain and of receiving SOC commands was completed in February 1993. The Hughes UHF F/O satellite model was ported over to the TAD and successfully run with the CUS in a simulation mode at PARAMAX's development facility in March 1993. The installation of the basic TAD CUS and UHF F/O MUS is scheduled for the summer of 1993 with an enhanced version of the CUS with the block programming capability scheduled for delivery in early 1994.

Installation of a TAD at OAFB is underway with completion scheduled for the end of 1993. The GPS Block IIA software rewrite has started and is expected to be completed by the spring of 1994. The MILSTAR satellite SPO is in the planning stages of developing a MILSTAR MUS to support training in their SOC at FAFB. The GPS Joint Program Office (JPO) is considering acquiring a TAD to support training at its satellite Master Control Station at FAFB along with developing an MUS for its new GPS Block IIR satellite.

An additional use of the TAD is as a test driver in the AFSCN Data Systems test program. A development laboratory TAD has been installed at IBM's satellite ground control software maintenance facility to support software maintenance and test activities. It is expected to be operational by the summer of 1993.

FUTURE DIRECTIONS

An advantage of the TAD is its portability across hardware platforms. Current technological advances make it feasible to port TAD software onto high performance graphic workstations (parallel architecture) and place them into the SOC at a significant cost savings over the present hardware architecture. This alternative is being investigated. With the TAD being ported to a workstation and located inside the SOC, its facility installation and security cost could be significantly reduced.

The TAD has been adopted as the standard AFSPACECOM crew trainer. Additionally, it has been identified as the simulation element for the Air Force's next generation satellite command and control architecture, the Advanced Satellite Control (ASC). The ASC stipulates a distributed architecture with heavy reliance on open architecture and commercial standards interconnected via high capacity high speed fiber optic data links.

Bibliography

1. The Aerospace Corporation, TOR-0089(4485-01)-1, Satellite Control Simulation System (SCSS) Summary, February 1990.

2. AFSPACECOM Statement of Operational Need 012-88, Satellite Control Simulation System (SCSS), August 1990.

3. AFSPACECOM UHF Follow-On Training Augmentation Device Technical Requirements Document, 8 February 1991.

4. System Specification for the Ultra High Frequency Follow-On Training Augmentation Device (UTAD), UNISYS, SS-CUE-000349, 2 August 1991.

5. Software Requirements Specification for the Common User Software for the Training Augmentation Device (TAD), CG-000360, 20 January 1992.

6. Prime Item Development Specification for the Training Augmentation Device (TAD), CP-CUE-000358, DRAFT, 30 September 1991.