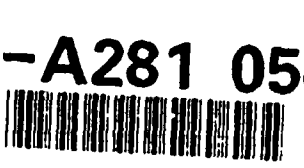

RL-TR-94-24
Final Technical Report
April 1994

VIRTUAL ENVIRONMENTS IN COMMAND & CONTROL FOR ULTRA-HIGH DEFINITION DISPLAYS

Massachusetts Institute of Technology

Sponsored by
Advanced Research Projects Agency
DARPA Order No. 8931
and the
Joint National Intelligence Development Staff (JNIDS)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-94-24 has been reviewed and is approved for publication.

APPROVED: Richard T. Slavinski

RICHARD T. SLAVINSKI
Project Engineer

FOR THE COMMANDER: John A. Graniero

JOHN A. GRANIERO
Chief Scientist
Command, Control & Communications Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL (C3AB) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

VIRTUAL ENVIRONMENTS IN COMMAND & CONTROL FOR ULTRA-HIGH DEFINITION DISPLAYS

Nicholas P. Negroponte
Dr. Richard A. Bolt

Contractor: Massachusetts Institute of Technology
Contract Number: F30602-92-C-0141
Effective Date of Contract: 1 July 1992
Contract Expiration Date: 30 June 1993
Short Title of Work: Ultra-High Definition Displays

Period of Work Covered: Jul 92 - Jun 93

Principal Investigator: Nicholas P. Negroponte
Phone: (617) 253-5960

RL Project Engineer: Richard T. Slavinski
Phone: (315) 330-2805

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by Richard T. Slavinski, RL (C3AB), 525 Brooks Rd., Griffiss AFB NY 13441-4505 under Contract F30602-92-C-0141.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 1994	Final Jul 92 - Jun 93

4. TITLE AND SUBTITLE
VIRTUAL ENVIRONMENTS FOR COMMAND & CONTROL
FOR ULTRA-HIGH DEFINITION DISPLAYS

5. FUNDING NUMBERS
C - F30602-92-C-0141
PE - 62708E
PR - H931
TA - 00
WU - 01

6. AUTHOR(S)
Nicholas P. Negroponte - P.I.
Dr. Richard A. Bolt - P.I.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Massachusetts Institute of Technology
20 Ames Street
Cambridge MA 02139

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
*Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington VA 22203-1714

Rome Laboratory (C3AB)
525 Brooks Road
Griffiss AFB NY 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-94-24

11. SUPPLEMENTARY NOTES
*Additional Sponsor - Joint National Intelligence Development Staff (JNIDS)

Rome Laboratory Project Engineer: Richard T. Slavinski/C3AB/(315) 330-2805

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Under this contract, the MIT Media Laboratory pursued three coordinated streams of research: the development of graphically intelligent tools and principles to support the interactive creation of symbolic information landscapes, the integration of such landscapes with pictorially convincing virtual environments, and the enabling of multi-modal natural language communication with the virtual environment display and its contents via combinations of sppech, manual gesture, and gaze. These streams will converge in year three of the overall project in the form of an ultra-high definition, seamlessly tiled wall-sized display (datawall).

14. SUBJECT TERMS	15. NUMBER OF PAGES
Symbolic information landscapes, multi-modal interaction, virtual environments, large format displays, intelligent agents	72
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

EXECUTIVE SUMMARY

The work under this contract reflects the first year activities under three coordinated streams of investigation:

- the development of graphically intelligent tools and principles to support the interactive creation of *symbolic information landscapes*;
- the *integration* of such landscapes with pictorially convincing virtual environments;
- the enabling of *multi-modal natural language communication* with the virtual environment display and its contents via combinations of speech, manual gesture, and gaze.

These streams will converge in year three of the overall project in the context of an ultra-high definition, seamlessly tiled wall-sized display (DataWall).

* * * * * *

Creation of symbolic information

Summary of 1st year work

Large Scale High-Resolution Display:

- We upgraded hardware of the Large High Resolution Display Prototype (6K x 2K display) to use faster and more powerful hardware. This step provided a system acceleration of over 250% from the original configuration

Selective Information Filtering:

- We successfully ported blur/transparency tools to the 6K x 2K display prototype
- We created a simulation of 3-D landscape which included both text and graphics.

Adaptive Graphics:

The previous model of adaptive text had fixed luminosity difference; blurred rectangle background. This last year we added:

- user-controlled luminosity difference
- user-controlled hue difference
- drop-shadow and transparent rectangle background
- a specialized model to maintain perceptual color, compensating for the interactive effect of the background color (Albers)

Intelligent Design Tools:

- We developed "AIDE", a system that automatically lays out statistical chart-like graphics for complex multi-dimensional information. AIDE uses artificial intelligence (case-based reasoning) to design the graphical presentation and is capable of designing a number of different static and dynamic graphs, allowing the user to quickly and effectively browse and analyze large datasets.
- We developed a distributed model of dynamic design where individual symbolic representation is considered as an active module which can determine its own graphical behavior in time. The investigation included:

- development of a model of design process for expert designers.

- implementation of a simple automatic layout system based on the model.

- use of the memory-based technique to represent hard-to-articulate design decisions.

• We investigated a task level interface for the multi-layer map prototype by integrating the domain knowledge in a form of information-seeking goal hierarchies.

• We built a prototype of interactive three-dimensional symbolic information display.

MEDIAte: An Intelligent Authoring Environment for Information tools

The goal of this work is to research and prototype a unified, graphically intelligent multiple-media information authoring environment that will assist the analyst in the complex tasks of gathering, analyzing and presenting information. The approach utilizes:

1. spatial parsing and generation using relational grammars,

2. programming by demonstration,

3. object oriented knowledge-based domain representation and case-based reasoning.

The current challenge is to bring the two worlds of AI and Graphics together into a single seamless integrated environment.

This work for ARPA/JNIDS had its formal kickoff meeting, and a plan has been developed to acquire a first example data-set.

Projected work in year 2

- We shall be investigating various technologies to support the prospective DataWall. Possible technologies include: the Texas Instruments Inc. "digital micromirror device" (TI DMD); deformable membrane mirror technology; liquid crystal display technology.

- We intend to integrate a subset of Graphical Intelligence with a subset of Dynamic Graphical Presentation in the 6Kx2K display (and any other display prototype that may provide an intermediate to our projected DataWall).

- We shall be integrating subsets from Symbolic Information with Virtual Environment and Multi-modal Communication. This shall form the initial step of integrating the current three streams of research into a common path.

- We shall be developing new interface tools to accommodate and support this integration.

- In our ARPA/JNIDS work:

  Under *Domain and Design knowledge representation and reasoning* (TYRO): We shall digitize and otherwise capture an example analyst's data set. We shall build interfaces to Framer for an analyst's example domain, plus a small analyst's presentation library. Using the Framer interface, we shall represent the knowledge in a small example domain, then re-purpose the example with inferred hyperlinks.

  Under *Instructability in Information Tools* (MONDRIAN): We shall extend programming by example methodology to include dynamic video sequences Also, we shall develop techniques for editing and debugging programs defined by example. We shall experiment with alternative generalization strategies, and produce input to a visual language parser by example.

Under *Automatic Layout* (Logic of Layout): We shall create relational grammars for specific JNIDS applications and incorporate more lexical types for multimedia documents. We shall experiment with new grammars (e.g., temporal grammars), and implement techniques for exploring design alternatives generated by the grammar.

Under *Integration:* We shall design an architecture that will accept MEDIAte components. We shall: extend intelligent and advanced graphical tools and include hooks to other relevant research projects; determine platform capabilities and compatibility; build small experimental integrated test beds.

Integration of symbolic information with virtual environment

Summary of 1st year work:

- We expanded and enhanced the "3D" virtual environment system, with particular attention devoted towards making the system more efficient, while retaining the ability to rapidly prototype applications within the system. A dependency network has been designed and implemented in the existing TCL-based system so that actions will automatically occur when certain conditions are satisfied. We have incorporated the scheme language as a new front end to the system, with the advantages of being both interpretive and fast.

- We have implemented a distributed system used to construct both virtual actors and the virtual world they inhabit. The completed "world program", which is comprised of a set of "active objects", distributes itself over a network of workstations and supercomputers at run-time. Using this system we have successfully built and demonstrated two virtual environments:

1) One environment wherein autonomous actors moving asynchronously through the space and actively sensing their environment. The actors can: locate the position of other actors, tap into world graphics and sound and use this sensor information to modulate their own behaviors and appearance. This scenario provides a fertile ground for investigating issues involved in sensor delay, behavior modulation of a given virtual actor, and questions involved with having multiple autonomous actors in the same virtual space.

2) The other environment is an implementation of a well known, published planning problem, which we have used to validate the performance of the reactive planner, and shows off how virtual actors "sense" the virtual world.

- We have developed intelligent agents for navigation and presentation in virtual environments, including an "intelligent camera" capable of path planning. If the operator wishes to move from the current position A in the VE to some target location B, the intelligent camera should compute a smooth, collision free path from point A to point B which always maintains the appropriate visual context, and which moves smoothly along a physically realizable path. The intelligent camera and path planning systems function in a photo-realistic, interactive "virtual museum", which includes a number of sculptures and paintings.

Projected work in year 2:

Physically based models of humans and vehicles

- In Year 2, we will develop a representation of motion goals which is based solely on the inputs to our dynamics simulators: mass, inertia, accelerations, forces, torques, kinematic and dynamic constraints. This will allow us to represent the coordinated motor behavior of figures and the motion of mechanisms and vehicles using a single vocabulary based on Newtonian mechanics.

Spatial and temporal registration

- We will extend and elaborate texture mapping methods to ensure that symbolic presentations generated by the Visible Language Workshop — especially animated presentations — be properly registered with respect to 3D visual imagery.

Networking

- In Year 2, we will characterize the levels of representation of computational models for command and control tasks, including identifying and quantifying bandwidth and synchronization requirements. Again, we intend to use our developing VE software and hardware testbed as an experimental platform.

Multi-modal natural language

Summary of 1st year work:

- We developed a revised distributed processing network, in the context of a newly-installed 6-ft diagonal back-projection display where user stands, not sits, in interaction with screen. In this setting we have integrated a new connected speech system (HARK by BBN Systems, Inc.). This system, at our request, produces timing information as to the temporal location of each detected word in the connected speech stream, and sends this information to our distributed processing network.

- We have further refined our "gestlet" characterization of the kinds of two-handed co-verbal gesture in normal human-human interaction. We have defined a representation syntax for objects in a knowledge base to enable mapping of hand posture to object orientation. We developed a "gestlet" gesture-feature detector, including a module which segments and characterizes gesture data. We defined and developed a set of primitive features for use with our coverbal gesture recognition scheme.

- We developed a top-level graphics language (dubbed "MVERSE") for elaborating 3-D graphical scenes, e.g., maps, and for populating terrain with referenceable objects, e.g., vehicles, buildings, roads, aircraft. The MVERSE system allows for a "picking" function to select out objects, and for cinematic control of camera/viewpoint position.

- We designed a scheme that allows flexible construction of interface agents capable of a wide variety of facial gestures, eye movements as well as crude synchronization with synthesized speech. The system is being implemented in the X-window system, allowing it to run on any UNIX machine.

- We developed a test environment to derive formula of gaze estimation, including calculating where an on-screen agent is looking and where the user perceives this agent to be looking.

- We have specified initial requirements for the facial animation of our on-screen "agents," coupled with input interpretation, information abstraction, and response generation of multiple modes. Also, we have developed an initial implementation of a hierarchical method of response control and feature abstraction

- We developed a new method for color coding in our Gestalt Perception simulator, with subjective-based color space used to increase the accuracy of the color recognition. This module lets the computer "perceive" objects on the graphics display in ways in which a human observer is likely, under Gestalt principles of perception, to see them grouped, and will help the machine interpret statements by the user which refer to groupings of displayed objects.

Projected work in year 2:

In year two, we shall continue our multi-modal natural dialogue software planning and generation effort. In particular, we shall develop the *time dimension* of the user/machine dialogue situation.

- We shall develop and prototype a comprehensive scheme to have the machine encode dialog "history" of its interactions with the user so that the machine will be able to respond to user reference to past events or situations (e.g., "Go back to when the armored units entered from the left...").

- We plan to extend the above scheme to handle encoding of the history of the exchange forward in time to contemplate "anticipations," or "expectations" of aspects of the

dialogue context and situation (e.g., "When the air units enter from the south [do such-and-such]).

- We shall be extending our gestural work to ensure that the "gestlet" scheme is effectively modular, and independent of any particular manual gesture-sensing technology. This step will ensure that our conceptual framework will be valid in the face of advances in manual sensing technologies (whether by "glove" or by an image-processing approach).

- We shall be more exactly specifying the projected behaviors of our on-screen "agents," in particular defining the conceptual boundary between their socio-linguistic expertise in dealing with the human user and any domain or subject-matter expertise they may possess.

INTRODUCTION

This report attempts to summarize detailed research progress documented in various publications, reports, articles, and/or academic theses which are identified herein.

Purpose of the project:

The purpose of this project is to research and develop in prototype a command and control setting wherein the commander/decision-maker communicates in *concurrent speech, gesture, and gaze* with a rich *symbolic information landscape* integrated with the *virtual environment*, and rendered on an ultra-high definition, seamlessly tiled wall-sized display. The project will also research and prototype a unified, graphically intelligent multiple-media information authoring environment.

Project personnel:

Personnel at the MIT Media Laboratory directly involved in the conduct of this research are:

Dr. Richard A. Bolt, Senior Research Scientist and member of the Media Lab's Perceptual Computing Group.

Prof. Muriel R. Cooper, Professor of Visual Studies and Director of the Lab's Visual Language Workshop.

Mr. Ronald L. MacNeil, Principal Research Associate at the Media Laboratory and co-founder with Prof. Muriel Cooper of the Visible Language Workshop.

Dr. David L. Zeltzer, Principal research Scientist at the Research Laboratory of Electronics, MIT.

CREATION OF SYMBOLIC INFORMATION

Relevant Personnel:

Work under these four sub-topics of managing visually complexity was completed under the supervision of **Prof. Muriel R. Cooper**, Professor of Visual Studies and Director of the Lab's Visual Language Workshop, and **Mr. Ronald L. MacNeil**, a Principal Research Associate at the Media Laboratory and co-founder with Prof. Cooper of the Visible Language Workshop.

* * * * * * *

Large Scale High-Resolution Display

System Upgrade

The first year of work on the 6Kx2K Display Prototype involved a number of changes to the hardware and software to improve its performance and functionality. The hardware was upgraded to the current configuration of 3 Metheus 5000 Frame Buffers driven by an IBM RS/6000. This new hardware increased the speed of the system over the previous hardware by about 250%.

The change to the new hardware required a reworking of some of the basic window system software, BadWindows version 2.0. Many of the changes were required by specific characteristics of the IBM operating system and additional function calls for the new Frame Buffers. In addition, BadWindows itself was extended to include new algorithms for adaptive text and focus. Routines were added to allow for the display of blurred rectangles and images. Also, an entire suite of routines was added to allow for "adaptive text", a series of techniques designed to ensure the legibility of text in dynamic designs. There are a number of techniques which have been added, including:

- routines to remove the high spatial frequencies of the background, either by a transparent rectangle or a transparent blurred rectangle. These rectangles can either be a fixed color, or the average color of the bounding box of the text itself
- routines to maintain contrast differences between the text and the background
- routines to maintain hue differences between the text and the background
- routines to compensate for the interactive effect of colors, allowing the user to choose a specific referent color for a piece of text

Application Improvements - World Map Display

The application that displays the large cloudless image of the Earth has been upgraded to include new images. The new map image has better registration. In addition, a new image that includes elevation information has been created.

The new hardware has off-screen memory, which our older system did not. This advance allowed us to write a number of routines to access this memory, allowing small, quick, double-buffered animations. These routines were used to upgrade the previously slow display of cloud images over the cloudless map image.

Application Design -- Aide: Automated Intelligent Design Expert

Summary

A large portion of the work done on the 6Kx2K display included developing the system called AIDE. AIDE is a system that takes application independent data and produces chart graphics from the data, regardless of its content. It assists users of data who wish to design graphics by combining artificial intelligence techniques with current automated design software paradigms.

AIDE extends previous research in four important ways. First, it combines case-based reasoning with traditional rule-based algorithms for designing graphics. Second, because of their cognitive importance, the rule-based components of AIDE have special design knowledge for temporal and locative information, allowing the system to present this type of information in unique ways. Thirdly, like other automatic layout systems, AIDE is capable of producing traditional static graphics. In addition, however, AIDE can also design and present dynamic graphics. Finally, AIDE runs on the 6Kx2K display environment. This, combined with a unique set of design skills and techniques, allows the system to produce graphical images of a higher quality than found previously.

The research presented here contributes to the fields of artificial intelligence and automated graphic design by effectively combining them in one system. AIDE, the resulting prototype application, can be used to generate a number of different high-quality graphical representations from application independent data and offers potential benefits for anyone who needs to analyze or present data.

Overview

We can consider information to be, among other things, relevant or important aspects of data. That is to say that each set of data contains trends, patterns, and relationships among the elements and their characteristics (the lack of an easy-to-describe trend or pattern being a pattern in and of itself). These trends and patterns often tell important stories about the data in question and are needed to solve real problems. The ability to quickly and efficiently discover such information buried in a database is valuable and grows more difficult with its size and complexity.

Data can be analyzed in many different ways. However, the most powerful way to analyze a database as a whole and to get a good overall view of the information within a

database, especially if the user has no a priori knowledge of the database, is through the use of a graphical presentation (e.g., a chart, diagram, or map).

It is easier to recognize patterns, determine trends, or discover other information by using our perceptual skills for recognizing shapes, colors, and positional arrangement of objects than it is to parse a table of textual information. Graphical images are often more inviting than tables of data and allow the user an opportunity to investigate and examine data in a way just not available with tables. In addition, graphics can often condense data, allowing a view to analyze thousands or millions of data in one visual experience.

For hundreds of years people have been making such charts and diagrams from data. However, the variations on graphical forms are limitless. Choosing the appropriate graphical form for a particular set of data is no easy task. Each aspect of the design—the color of the marks, the width of the lines, the size of the labels—all communicate information and play a role in the effectiveness of the graphical presentation.

Trained graphic designers are specialists in this area, and are often given the responsibility of turning tables of data into powerful images that illustrate the underlying information in the database. This process of turning data into images involves choosing the format of the graphic, choosing the colors and symbols that comprise the graph, choosing the fonts, and hundreds of other decisions that on the surface might seem simple, but are in reality extremely complex—for each decision has a direct impact on the legibility, expressivity, and effectiveness of the final graphical image. Since the need to go from data to image is so common, it is hardly cost-effective or time-effective to hire a trained designer every time a user needs a graphic. As a result, the ability to visualize data has been given to the general user through production software, spreadsheets, and statistical tools.

There are two major drawbacks to the current ways of putting this power in the users' hands. First, the graphical intelligence of these systems is severely limited. The user must often make many design choices, including which form to use (scatterplot, pie chart, bar chart, etc.), which characteristics should go on the x axis, which on the y axis, what the tick marks should be like, and so on. The user is often not a trained designer, and hence, is not equipped with the skills needed to produce effective graphics.

The second problem is the graphical power of such programs is limited. This means that even if the user makes reasonable design choices, the graphical quality of the images that can be created is not very high. In most cases, the system just doesn't provide enough control over the individual design aspects of the image, such as the size of the marks, the colors, and so on.

What are possible solutions to these problems? Providing more control over the various design aspects would solve the second problem, but would make the first even more acute. However, if we could abstract the process of turning data into graphs and generate a set of rules for this process, these rules could be codified in software and a virtual designer could perform the job of a trained graphic designer. Current research in this area has met with limited, although significant, success. One of the biggest problems is that the design process is too complex and large to be codified in a set of simple rules. It does appear that designers follow general guidelines and styles when designing. However, these styles and guidelines are so complex, it seems unlikely that they could be delineated in a comprehensive list of strict conditional rules.

An alternative theory about the process of design is that designers borrow ideas from other designs—that is to say they create new designs based on old ones. Consider a particular diagram that communicates information about some specific set of data effectively and is easy to understand. If the data changes slightly (i.e., if the data is about horses, maybe the height of a horse changes), it would not be useful to completely

redesign the graphic. Similarly, if a few of the values in the dataset change, it would not make sense to completely redesign the graphic. The alternative is merely to modify the original visualization to reflect the new state of the data. As such, these new images could be considered to be "based" on the original. Design often happens this way—adapting other graphics to the state of the "new" data. The more different the new data from the old, the more redesign is needed. At some point, when the new data is significantly different from the original, a completely new design is required.

It is possible to use an artificial intelligence technique called case-based reasoning to model this behavior. AIDE is a system that uses case-based reasoning to automate part of the process of graphic design. AIDE takes a set of application-independent data as input and has a library of different data sets and their graphical representations. It compares the new data to the examples in the library, and based on input from the user, attempts to create a new design based on one of the examples. As an assistant, it does not merely take in data and produce graphics, but assists the user in the process of generating statistical graphics from data. This means that the user is an integral part of the design process.

How does the system work? The first half of the process utilizes the artificial intelligence technique called case-based reasoning. The analysis module takes the input database and produces an abstract representation of the data, called the *shape*, by performing some traditional statistical analyses of the data and their characteristics. This analysis includes sorting the characteristics into nominal, ordinal, and quantitative categories. In addition, AIDE extends this paradigm by treating two aspects of characteristics as special. These aspects are locative and temporal, and are singled out because of their special cognitive value. People have a natural tendency to organize things with respect to places and time. Hence, any system that tries to design like a human being must have knowledge of these aspects, and appropriate design rules to visually represent this data.

Once the system creates this abstract "shape" description of the input database, it then compares that shape to a library of example graphs. The rating module compares the shape of the input data to each of the examples' shapes to assess each as a possible basis for the visual representation of the input database. AIDE visually displays a subset of the examples along with their ratings and in doing so, provides the user with guidance about how to proceed. At this point, the user chooses an example and the system passes the input data and the chosen example to the adaptation module that modifies the chosen example to fit the input database.

Once the example is adapted, AIDE then passes the adapted example and the input data to the designer component. This component lays out a large, full-color visual representation of the data based on the adapted encoding and provides the user with an easy and effective way to discover the information buried within the data. The user can then choose another example from the library, modify the graphic manually, or in relevant situations, ask the system to try to "extend" the graphic to convey more information, through the use of a rule-based design component similar to those found in other work. Once a user is satisfied with a graphic, he can add the new image to the example library for future use.

AIDE extends previous research in a number of important ways: 1) It combines the artificial intelligence technique known as case-based reasoning with traditional rule-based automated layout techniques; 2) Because of their special cognitive value, the rule-based designer component of AIDE treats temporal and locative information as special and contains special design rules to represent these characteristics; 3) Just as other automated layout systems, AIDE can design and present static images, bu: the system is also capable of producing dynamic graphics; and 4) AIDE contains a number of sophisticated new design skills that, combined with its unique display environment, allow the system to create graphics of a higher quality than in previous work.

Design of intelligent multi-layered information display

New design paradigm: a distributed model of dynamic design

The purpose of the first year investigation has been (1) to develop a method to represent graphical behavior of visual design for a medium in which both information and interaction are dynamic, and (2) to implement an experimental prototype system based on the method. We have explored and developed a conceptual model of dynamic design: a distributed model of dynamic design where individual symbolic representation is considered as a module that can determine its own graphical behavior in time. The method explored in this research is not a normative one, rather it is intended (1) to provide a framework for the development of automatic design systems for dynamic media and (2) to server as a conceptual model with which a design can think.

We consider a design as a system and the visual designer's role is to design its graphical behaviors, or method of designing. In a case of a computer-based information display, the system is a computer program which can perform designing according to the dynamic change of information and user's intention.

The system consists of a collection of small "designers" where each "designer" is responsible for presenting a particular information. We call this small designer a "design-module" in order to distinguish it from a human designer. A design-module itself is a system that can modify its graphical behavior as the situation changes and can cooperate with other design-modules. The design-module consists of (1) constraints with associated situations and (2) a mechanism that determines action selection.

A representation of a design-module's graphical behavior has been developed based on a informal observation of an expert designer's method of specifying dynamic behavior of design elements. We have determined two basic levels in the specification of graphical

behavior of individual design-modules: frame, and action selection scheme. Frame determines a boundary of the formal aspect of design. Action selection scheme determines the final design based on the dynamic contextual change.

A simple automatic layout system based on the distributed model

An experimental automatic layout system has been developed and tested with simple layout problems. The system is developed using a simulated multi-processing environment on a UNIX workstation. The system can solve a simple layout problem based on constraints (frame) and memory-base (action selection scheme).

The basic problem solving process is the following: First, a design-module observes global situation, such as the user's interest and the agent's current importance, and determines a set of constraints that are applicable. It then determines a hypothetical solution (rough design). Second, each design-module takes "design actions" based on the hypothetical solution and its local situations, such as a background color and neighboring objects, in order to find a final solution. To find a hypothetical solution we use a simple constraint propagation technique with a constraint heuristic search technique [Fox et al. 89] [Sycara 91]. We have applied texture measures to the attribute ordering in the search, and that strategy has significantly reduced backtracking in the search process.

Memory-based learning and reasoning

Memory-based learning techniques have been used to model perception based design decisions, such as visual balance and rhythm decisions. Since perceptual design decisions are difficult for a designer to articulate, neither the procedural programming nor the knowledge-based approach are appropriate. We have used the memory-based learning technique based on [Maes and Kozierok 93] and [Stanfill and Waltz 86], and implemented a system that learns a designer's design action selection scheme by observing the editing processes. The system keeps a record of the designer's past design

decisions in the form of situation-action pairs, and a new situation is compared against them to determine the best possible design action to perform. The situation is represented as a set of features, such as number of alignments in a frame and distance to closest object, and currently supported design actions are spatial actions (move-up/down/left/right, and stay) and transparency (increase and decrease).

The results with simple layout problems have also shown that the memory-based learning technique is effective for acquiring a perceptual design decision scheme. The system could (1) learn design actions for particular design styles, (2) find some unexpected opportunities to improve the design, and (3) deal with some unexpected situations. We conclude that the memory-based approach is fairly suited to model the design decisions that are mostly based on perceptual feedback. However, one drawback of the memory-based reasoning is its computational speed. This may become a bottleneck when it is applied to dynamic design systems.

A task level interface for the multi-layer map composing system

As an extension of a previously developed multi-layer map composing system, we have investigated the integration of domain knowledge in the system in order to provide a user with a task level interface. A plan-based representation has been examined to represent the domain knowledge. The representation consists of goals (information seeking/providing goals) and associated plans (information seeking/providing plans). A plan consists of a set of subgoals that accomplish a non-primitive goal. A primitive goal is a goal that can be directly executed. A plan consists of a precondition-list, a conflict-list and an effect-list. The effect-list contains a set of goals that are achieved by executing the plan. The precondition-list contains a set of goals that must be achieved in order to accomplish goals in an effect-list. The conflict-list contains a set of goals that is semantically irrelevant or formally conflicting.

Provided information seeking goals by a user, the system can determine priorities of information using the activation network technique. The system injects activation energy to the goals that match a user's current information seeking goal. Then, goal modules that have a high activation level spread the activation energy to subgoals and higher level goals as well as the negative energy to conflicting goals.

We have implemented a simple prototype of an interactive map of Cambridge using this plan-based representation. The system accept a set of goals using a simple textual interface, and then it can automatically compose a map by adjusting translucency of the individual layers. We conclude that (1) the plan-based representation is effective in structuring the mapping information and (2) the activation network technique is fairly successful in finding relevant information according to the user's request.

Interactive three dimensional symbolic information display.

We have implemented an experimental software tool that explores the use of interactive three dimensional stereo graphics as a medium for symbolic communication. In particular, we have investigated three dimensional (infinitely) large space to represent complex information in typography. We have used geographic information as a domain and built a prototype of the world map system that allows a user to zoom in and out of a world view to Cambridge and MIT Media Lab. This experiment has shown that the use of three dimensional space is effective (1) to maintain the context when browsing in a large and deep data space and (2) to provide an overall structure of complex information. The experiment has also provided us with both design and technical issues that need to be further explored: typography in three dimensional space; use of three dimensional motion for symbolic communication, and as a control device to manage the display; implementation of three dimensional anti-aliased text etc.

MEDIAte: An Intelligent Authoring Environment for Information Tools

Agents

In this project, we built an "Agent for Dynamic Information Retrieval from a very Large Database. Very large means that the information space within which the agent operates is too large for the user to get an overview of the available data. The user, rather, tries to focus on certain interesting aspects. The interest of the user will be shifting over time, and the agent dynamically adjusts to this shifting. The project runs on the 6000x2000 pixel display and uses 2MB text of interview transcripts as the database. The concept of agents makes it easy to separate parts of an application and run them on different machines

Sound

We had a slow start with finding the best way to establish the server-client relationship. We are trying to use sound extensively to find out what functions one would like to use We think the main use for sound in the VLW is to "enhance graphics and dynamics in the interface." Sound offers an extra channel to draw the user's attention, and while there is the possibility that sound in some instances may be purely redundant, in other instances sound plus graphics can lead to *data reinforcement*. In any event, one of the topics to explore is: What part of information should be expressed through sound and what part through graphics?

Framer

Framer can be used to interactively program a user sound interface through commands or design it with a mouse. Further, it allows for applications where the user interacts by modifying the appearance of the interface .

We can use FRAMER to design and build an interface for an interactive library of sounds. Sounds play as the cursor moves over them. The sounds can be described and sorted in terms of qualities such as shrillness, noisiness, animal-ness, wetness, or whatever qualities the designer finds descriptive. FRAMER stores knowledge about the appearance and location of sounds in the interface, plus the various qualities they possess relative to other sounds, as determined by the user. The goal is to build a very reactive interface, that enables the user to find an appropriate sound easily as well as making it easy to assign sounds to qualities and vice-versa.

For example, we can lay out a random selection of sounds in a rectangular pattern. Moving the mouse plays the sound the cursor overlays. The goal was to experiment with continuous sound. We are increasingly convinced that sound can be a major contribution and enhancement to a user interface. In this research we used opaque non-overlapping rectangles, each having to incorporate only one sound playing at one time. The next step would be to use transparency and have multiple sounds playing at one time.

Relational Grammars for Interactive Design

This work investigates an approach to interactive design tasks based on Relational Grammars. Relational Grammars extend traditional one-dimensional string languages to higher dimensions through user-supplied domain relations. Design support takes the form of graphic inferences on partial input during design interaction with the user. Because of the nature of the rule definitions, design elements can be roughly "sketched" without concern for their specific details. During the interaction process, the system installs constraints, adds default attributes, and/or builds higher-level composite structures out of the original input. An example scenario of the use of this formalism is presented supporting the task of *document design*. A general algorithm for the integration of grammar-based parsing in an interactive design environment uses an interactive agenda;

this agenda presents pending rule firings—alternative actions that may be invoked—and allows the designer to select the most appropriate action.

Additional work has focused on extending the bottom up, interactive approach by investigating the use of predictive parsing techniques with relational grammars to support the design of structured artifacts. In this paradigm, relational grammars are used to offer continuations of design alternatives. A prototypical editor for the design of flowchart diagrams has been created. This editor incorporates both a bottom up algorithm as well as a predictive algorithm. The bottom up approach allows the grammar rules to either fire automatically, or to be delayed on the interactive agenda controlled by the designer's actions. The predictive algorithm provides continuation support during the design process. This mechanism provides the ability to identify elements of the design that are incomplete and offers a set of possible continuations from these elements. The graphic continuations are in terms of both lexical categories and the relationships of these categories to the unfinished elements. Both the bottom-up and predictive approaches install graphic constraints enforcing layout conventions specified by the grammars.

Future research directions include the use of relational languages for the acquisition of new grammar rules, the automatic generation of graphic form in dynamic media, and the mixed mode of interaction with interpretation, generation and prediction all serving the design process.

References/Publications/Theses:

"6000x2000 Display Prototype", *Visible Language Workshop*, Media Laboratory, MIT, Cambridge, MA, USA, 1991.

Bertin, J., *Semiology of Graphics*, W. J. Berg translation, University of Wisconsin Press, Madison, WI, USA, 1967, 1983.

Buchanan, B. G. and D. C. Wilkins (eds.), *Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems*, Morgan Kaufmann, San Mateo, CA, USA, 1993.

Christensen, J., Marks, J., and Shieber, S., "Algorithms for Cartographic Label Placement", *Proceedings of the American Congress on Surveying and Mapping*, Volume 1, New Orleans, LA, USA, 1983.

Cleveland, W. S. and R. McGill, "Graphical perception: Theory, experimentation and application to the development of graphical methods", *Journal of The American Statistics Association*, Volume 79, 1984.

Fox, Mark S., Sadeh, Norman, and Baykan, Can. "Constrained Heuristic Search." In Proc:. *Eleventh Int. Joint Conf. Artificial Intell.* pp. 309-315, 1989.

Ishizaki, S., et. al., "Adjusting simultaneous contrast effect for dynamic information display", *Proceedings of the 8th Joint Conference on Color Technology*, pp. 81-84, 1991. (in Japanese)

Mackinlay, J., "Automating the Design of Graphical Presentations of Relational Information", *ACM Transactions of Graphics*, Volume 5, Number 2, USA, 1986.

Maes, Pattie and Kozierok, Robyn. "Learning Interface Agents." In Proc. AAAI '93. 19993. Stanfill, C. and Waltz, D. "Toward Memory-Based Reasoning". *Communications of the ACM* 29(12): 1213-1228. 1986.

Riesbeck, C. and R. Schank, *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1989.

Sycara, Katia, Roth, Steven F., Sadeh, Norman, Fox, Mark S. "Distributed Constrained Heuristic Search." *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 6, (November/December 1991): pp. 1146-1461. 1991.

Tufte, E. R., *Envisioning Information*, Graphics Press, Cheshire, CT, USA, [illegible].

Tufte, E. R., *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT, USA, 1983.

INTEGRATION OF SYMBOLIC INFORMATION WITH VIRTUAL ENVIRONMENT

Relevant Personnel:

Work in this area was completed under the direction of **Dr. David L. Zeltzer**, Principal Research Scientist at the Research Laboratory of Electronics, MIT.

* * * * * * *

3D

"3D" is large software system we have designed that serves as a testbed for implementing virtual environment (VE) systems. It is essentially a tool kit for creating custom VE applications that use multi-modal, interactive computer graphics and simulation techniques. 3D runs on a variety of vendor platforms, e.g., Silicon Graphics, Hewlett Packard, Sun Microsystems, Apple Computer, and NeXT. We use 3D exclusively to support the VE research we are conducting. Details of the 3D system can be found in [2].

3D has been under development for some time—for example, work on the graphics and rendering modules began in 1985. Here, we will describe the basic functionality of the system, and note those components that have been developed in the past year, with ARPA/Rome Labs support. There are six broad areas we will briefly discuss: the object database subsystem; the software and hardware renderers; the developers' TCL command line interface; the extensive library of built-in subroutines; distributed, inter-process communication; and, finally, the graphical user interface.

Object Database

While 3D is not a modeling system, it provides complete support for reading and writing geometrical models, and for accessing and modifying any or all of the conventional

graphical object attributes, such as transformation matrixes, smoothness and shading parameters, color, textures and, as of this year, object fillmodes, i.e., whether text, for example, is displayed in outline or filled-in.

There are eight primary types built into 3D, and the language can easily be extended to include new data types as demanded by a particular virtual world application. The object database subsystem has been designed and implemented such that objects, collections of objects, and scripts describing object attributes are organized hierarchically, much like a Unix file system. In fact, the syntax of object commands and their arguments closely follows the Unix file system command syntax. This was done to minimize the learning curve for new users, who, it is assumed, are programmers familiar with Unix-based workstation environments. Directory trees can be made using the "mkdir" command, the state of autonomous agents in the virtual world can be stored hierarchically, and arbitrary combinations of 3D types can be created and stored for easy access in directories.

Software and Hardware Rendering

The 3D rendering subsystem has been designed to take advantage of existing graphics hardware subsystems on particular computing platforms. At the same time, an extensive software library of rendering routines enables quality rendering in software, of course, at the expense of real-time response. In addition to the A-buffer renderer, this year ray tracing and radiosity rendering routines were installed.

3D allows users and developers to access any and all of the conventional graphical and geometric attributes of objects to be rendered, including all of the usual camera parameters, such as viewpoint, view normal and field of view.

Interpreted Language with TCL Command Line Interface

3D provides a complete language for programming actions within a virtual environment, and rapid prototyping of virtual worlds was a design goal from the outset. For this reason, 3D provides an interactive, text-based interface which has been implemented with the TCL [5] interpreter. This year, a performance profiler was added so that various aspects of system performance can be monitored and evaluated. Also new this year, we have implemented and installed several libraries callable from the TCL front-end, including keyframe animation tools, quaternion manipulation routines for controlling rotational degrees of freedom, and constraint-based camera control routines, which will be discussed further in the section on intelligent camera control.

Built-in Support

3D maintains a large library of useful, built-in modeling and simulation tools. These include routines for describing kinematic chains and for computing inverse kinematics; routines for common mathematical functions including the standard linear algebra computations; finite element modeling and simulation is supported, as is dynamic simulation of physical systems; and, finally, collision detection among polyhedral objects is also a built-in support function. New this year are routines for computing inter-object visibility relationships (i.e., "Is object1 in view from the camera position?" "Can object2 be seen from object3?"). Also recently implemented and installed have been routines for spline fitting and interpolation.

Distributed, Inter-process Communication

The computational cost of VE simulation, multi-modal device handling, and multi-modal displays means that distributed, concurrent processing is a requirement, which 3D fully supports. Processes may be started up on arbitrary hosts across the network, and communication with those processes may be maintained using several mechanisms. This

includes using standard, high-level Unix network protocols such as pipes and sockets. We have recently implemented inter-process communication with arbitrary, distributed TK processes, where TK is Ousterhout's recent extension of the TCL system. Using these distributed communication mechanisms, we have implemented communication with a remote server that controls VPL DataGloves and returns real-time hand configuration, position and orientation; a server that generates realistic sound effects on demand; a server that manages a voice recognition system; a server that manages an MIT- built force-feedback joystick; and new this year, a server that computes path planning algorithms for the intelligent camera interface we will discuss below.

Graphical User Interface

The TCL language was originally written to be integrated with the widget library of a window system, and can serve two purposes in such a context. These are to configure the actions of an application's interface, and to design the appearance of that interface. TCL is used for both purposes by 3D. The widget set used is OSF/Motif. Ousterhout's recent enhancement to TCL, TK, provides an event-driven, interpretable graphical interface tool kit, and we have recently installed TK as part of 3D.

WavesWorld

Complex systems are built of many interacting components and assemblies of components. A VE system, for example, may consist of a collection of distributed computing resources and I/O devices, coordinated and controlled by any number of system-level software modules [1, 8].These functions are performed by the 3D system.

In addition, we expect that any VE system will support a set of computational models and simulation processes that make up the virtual world, each of which may be arbitrarily complex. For example, virtual actors in a VE must have a set of behaviors to perform, whether the actor represents a robot capable of only a few simple actions, whether the

actor is a synthetic human possessing a wide range of adaptive motor skills, or whether the actor represents an abstract entity such as a military unit in a simulated military operation.

The mechanism for selecting and sequencing skills to generate the routine behaviors of a virtual actor must link perception with action, in a process we call "motor planning." In earlier work we have described the architecture for a skill network we have implemented to model the routine behaviors of virtual actors, and we have also characterized the kinds of behaviors the skill network can capture, and the sorts of behavior it cannot model [6]. For each actor we need to construct a skill network, which will enable it to respond to simulated events and human input adaptively, and in real time.

This year we have designed and implemented a development environment, which we call WavesWorld, for designing, building and debugging distributed VEs and virtual actors. This system includes tools for assembling a collection of motor skills into a skill network, as well as runtime tools for controlling and visualizing the output of the skill network. WavesWorld makes use of and is completely compatible with 3D, and in some cases, such as inter-process communication, it extends the capabilities of 3D. Further details can be found in [7].

The intent of WavesWorld is to allow a user to quickly build a graphically-simulated character—a virtual actor—that can act autonomously in a networked virtual environment. WavesWorld uses a reactive planning algorithm which was outlined in earlier work [6], later independently elaborated by Maes [4], and which has been significantly extended for dealing with asynchronous and parallel execution [3]. A comprehensive sensing structure has also been added to the planner, complementing the high-level controls the reactive planner provides by allowing time varying sampling rates to be integrated into the perception mechanisms. In addition, this system integrates computational economics at the lowest level, so that all processes in WavesWorld are bid

at run-time. Finally, a networked, multi-modal, interactive development environment that supports debugging as well as design is essential to building complex VE systems. Therefore, WavesWorld has been seamlessly integrated into the NeXTSTEP development environment, which acts as a direct-manipulation, front-end to a large set of heterogeneous computing resources.

When approaching the task of designing a development environment for a VE and the virtual actors that inhabit it, the problems of distributed computing and parallelism come to the forefront. We want our system to be able to run in parallel as much as possible, and we want to take advantage of distributed computational resources over the network, but how do we present the information such that the developer is not overwhelmed?

In our system, we have developed system level tools for managing and distributing computation, as well as tools for organizing, steering, and visualizing the ongoing computations. We have developed our own process abstraction, which we call an "active object", which is the level at which we can distribute computation and allow parallel execution.

Computational actors in WavesWorld are implemented as collections of active objects, which are distributed, object-oriented, message-passing entities. Each of these processes maintains its own read-eval-print loop so that active objects can send complete programs to each other, and thus embed code inside of other active objects. Active objects are flexible enough to trade off computational power for communication bandwidth on the fly. In other words, active objects have facilities for determining, at run-time, if it is cheaper to:

- send another process a message, polling it for information, or,

- download some piece of code (i.e. a set of messages) which will run locally in the remote process and communicate the information to the sending process when appropriate.

An active object is a single, sequential process running on some computational resource. It comes on-line somewhere on the network and makes itself available for communication with other processes that speaks its language. After initializing itself and making itself available for communication, an active object drops into an endless loop, checking for new connections and evaluating the messages from previously established connections.

Since an actor's skill network maintains no explicit world model, all of the data about its relationship to the virtual world comes to the actor via its sensor agents and receptors [6]. Active objects have facilities for defining sensor agents and receptors. Sensor agents each compute some property of the VE that has meaning for its actor. Sensors are executable code that measure some Boolean proposition using information derived from receptors.

A receptor is a code fragment used by a sensor agent: it gathers the information in the VE that is necessary for the sensor to compute its own value. Receptors are executable code (i.e., a set of valid messages that an active object responds to) that are embedded in some other active object. The receptors have a sampling frequency and lifespan associated with them. Whenever a receptor is evaluated, if its current value is different than the last time it was evaluated, a message is sent off to the active object which embedded it. Receptors can be shared by many active objects at once, and their frequencies can be modulated over time.

In addition to supporting sensor agents and receptors, active objects have facilities for acting as a host for goal agents and skill agents. Goal agents represent a fleeting or permanent desire that some sensor be measured "true". The presence of a goal agent

implies the existence of a corresponding sensor agent, somewhere in the actor, capable of sensing when the goal has been accomplished.

The ability to act as a host for any number of goal, sensor, and skill agents is built into the core ActiveObject superclass. This allows a virtual actor's various goal, sensor and skill agents to be running asynchronously on a heterogeneous set of computational resources. Any active object which acts as a host for any agents is referred to as an AgentManager.

While the computation involved in sensing and acting in the virtual environment is distributed over a potentially wide-area network, the actual planning algorithm for a given virtual actor runs in a single active object. The various agents register with an instance of this subclass of ActiveObject called Planner. The Planner instance executes the planning algorithm, sending messages to the various skill agents to tell them to begin executing, and updating its local information concerning the various agents as messages stream in. If any agents die (e.g., an agent's host machine crashes) during the course of a Planner's life, it deletes them from the skill network, and updates the various interconnections of activation flow.

In addition to a Planner and some set of AgentManagers, a virtual actor needs some shared locus of state, both internal (e.g., energy level, likes and dislikes, etc.) and external (geometric, kinematic, and dynamic properties associated with the actor). This is implemented as an active object called the BodyManager.

In summary, a virtual actor is a collection of active objects and agents:

- a Planner
- a BodyManager
- some number of AgentManagers, each of which is managing some set of agents:

- goal agents
- skill agents
- sensor agents, each of which has some number of receptors injected in other active objects.

To a given virtual actor, the virtual environment is everything other than itself. In addition to other virtual actors, there are several active objects with which the component parts of a virtual actor will connect and exchange messages. The most obvious is the EnvironmentManager, which is the active object that acts as the shared locus of activity, managing the data not associated explicitly with anyone actor (e.g. the environment, props, laws of physics, etc.). An EnvironmentManager has the ability to render the VE directly, but it can also delegate the rendering to other active objects.

Using WavesWorld, we have successfully built and demonstrated two virtual environments, a "rave," and a simulated robot in a workcell. A "rave" is a dance floor with music sources, and 3 video screens where videos (Twin Peaks, Terminator 2, Brainstorm, Fantasia, etc.) are playing. In the space are "ravers," autonomous actors moving asynchronously through the space. They are actively sensing their environment, and can find out such things as the positions of other actors in relation to themselves, what music is playing (including several different attributes about the music such as lyrics, beats per second, etc.), and what videos are playing on the screens. The virtual actors use this sensor information to drive their behaviors, such as modulating the kind and manner in which they dance, the texture mapping of selected video images to their own body, or moving closer to other actors that they "like." This scenario provides a fertile ground for investigating issues involved in sensor delay, behavior modulation of a given virtual actor, and questions involved with having multiple autonomous actors in the same virtual space.

The second virtual world, called sanderWorld, is interesting because it is an implementation of a well known, published planning problem, which we have used to validate the performance of the reactive planner. In addition, it shows off some of the novel and interesting features of the system, such as the way in which virtual actors "sense" the virtual world. The scenario consists of a single virtual actor: a two- handed robot which has the dual tasks of spray painting itself and sanding a board. There are a few props in the world: a vise, a sprayer, a board.

The various agents are written in our message passing language, which is an extended version of TCL [5]. In general, a given agent is only a few lines of code, as the underlying AgentManager provides the facilities the agent needs to compute itself. Once the objects have been inspected and connected using the graphical interface, the user can test the robot in its virtual environment. We use a NeXT machine as the interface host, and an SGI SkyWriter as a back-end graphics engine. We also use various computers scattered around the network to run some number of active objects which are not intimately tied to certain hardware resources (e.g. sound/image input/output).

By telling the planner to spread activation (by pressing the sendActivation pulse button), our virtual actor eventually decides to pick up the sander, place the board in the vise, sand it, and then paint itself. As the robot makes up its "mind", the various agents send messages to the Voice active object, telling it to say things (i.e., "I want the board sanded," "I picked up the sprayer," "I am painted," etc.). This multi-modal output (the virtual environment is also being rendered in real-time on the SGI) helps the developer debug the world being built.

We are also developing automatic graphing and layout objects which can help the developer visualize the various interconnections between agents and active objects. One difficulty in building and debugging VE systems is visualizing running programs and understanding the information flow and connectivity between the component parts. The

system of objects we are developing allows the developer to move at various levels of detail and abstraction. They can concern themselves with the hosts the active objects are running on, what agents are contained in which active objects, which agents are connected to which others, where activation is flowing, and so on. This is an ongoing area of development.

The Virtual Museum

This year we have developed intelligent agents for navigation and presentation in virtual environments for command and control. In particular, we have been developing navigation aids in the form of an "intelligent camera" that is capable of automatic path planning. If the user wishes to move from the current position A in the VE to some target location B, the intelligent camera will compute a collision-free path from point A to point B which always maintains the appropriate visual context, and which moves smoothly along a physically-realizable path. The intelligent camera and path planning systems function in a photo-realistic, interactive "virtual museum", which includes a number of sculptures and paintings. What distinguishes this system from other VE systems is the emphasis on task level performance specification as opposed to direct interaction techniques. This is designed to aid the user in a variety of visually-guided-tasks in a complex, unfamiliar VE.

There have been several systems designed that permit visualizations of space through interactive walkthroughs. Most of these systems usually allow control of the viewpoint and gaze of the observer through some mechanical device. There have been treadmill interfaces with steeringbars, joystick interfaces, virtual joysticks implemented using mice or trackballs. Some systems permit total immersion through the use of head-mounted or boom-mounted displays, while others use projection screens or windows on a desktop. Few of these systems concentrate on specific problems encountered when humans try to move freely in the virtual world.

The virtual museum provides a research tool for understanding how to aid the human user in performing the following tasks within the VE:

- orientation
- navigation from point to point
- exploration of unknown areas
- presentation to external observers

By analyzing the tasks themselves, we have developed a more flexible system that allows higher-level interaction with the VE. For example, instead of interactively flying through a VE in an effort to find some object—which may be a non-trivial task in a complex and unfamiliar space—the human participant should be able to specify which object or objects are of interest, and the VE system should actively assist the participant in arriving at the desired goal smoothly, and without becoming disoriented.

The domain that we have chosen to implement is a visually rich one—that of an art museum. The museum contains both two- and three-dimensional objects laid out in many different rooms. We chose the museum domain because it is a kind of spatial information space within which we can formulate a task level description fairly easily.

Here are some of the tasks that we have considered in the museum:

- Show me the museum as a whole.
- Give me an idea of what paintings are in the museum.
- Take me on a tour that shows me this type of painting.
- Take me on a tour that shows me these specific paintings.

- Show how to get to a specific painting.
- Take me there along the path.
- Show me a painting or sculpture instantly.
- Let me save and restore different places that I've been so I can get there again.
- Show me where I am and which direction I'm looking within the museum.
- Let me interact with that directly.
- Let me walk around a sculpture, or conveniently observe a single object from different viewpoints.
- Let me create a visualization of the museum for someone else.
- Let me interactively move through the museum using a conventional walkthrough technique and compare that with other ways of moving about.

Orientation

Perhaps the first and most straightforward task that we investigated was orientation within the museum. Because the museum has a fixed layout that is known *a priori*,, we can provide the user with an inset 2D map view of the VE. As the user moves anywhere in this environment, the representation is continually updated, showing the user's position, gaze direction, and field of view. Thus, visual context is always maintained so that the user always knows where they are in the VE and their directions of gaze and motion, which may be different. But the map view also serves as an input device. By clicking in this overhead view, one can either move immediately to a new position, change the direction of gaze or the field of view, or the user can ask the system to plan a path to some indicated, new position.

Navigation

In addition to the orientation problem, the system interface also provides assistance with navigation. The user can specify a positional goal in the museum using various input methods, and the system will either immediately move the user's viewpoint to match the indicated location and field of view, or the system will plan a smooth, collision-free path to the location. The user at any point can save or retrieve a position and gaze direction, allowing the user to build up a suite of known landmarks or favorite positions. Also a list of the paintings in the museum is provided and the user can select from this list in order to specify the indicated position and gaze vector.

Exploration

There are several modes of exploration that have been considered. First, the user may indicate some spatial region of interest in the museum, and the system interface will construct a guided tour of paintings found in that area. In a similar fashion the user might specify a list of paintings located somewhere in the museum, but whose positions are not known to the user. The system will then plan a smooth, collision-free path that is the shortest and most direct route that visits each of the paintings.

Finally, there are the "low level" modes of exploration. These are the modes that allow direct control of the degrees of freedom of the synthetic camera. Again, the user has a number of options. If the user wishes to observe a single object, the view can be constrained so that all the motion of the camera will take place relative to the specified object. In this fashion, it is easy to generate many different views of an object or to "walk" around the object. In addition, individual degrees of freedom can be manipulated so that the point of view can be moved along a single, view-centered axis. These can be indicated using familiar cinematic terms: craning, panning, tilting, rolling and trucking.

Lastly, the user can combine movement along several degrees of freedom using a single, interactive input tool in the form of a "virtual joystick".

Presentation

Since the system provides many different automatic view-generation modes(e.g., the "guided tour" and the automatic path planning to an indicated location), a great deal of attention has been devoted to having the system generate paths that make visual sense to the viewer.

First, the system constrains the movement of the camera so that it does not rotate too rapidly. Second, interactive camera movement through walls is never allowed, and planned paths always smoothly avoid collisions with walls and objects. This is a constraint that is conceptually very simple, but it is important for maintaining visual context—and it is a constraint that is usually ignored in VE systems. And finally, the system always tries to generate an appropriate and visually consistent direction and field of view. For example, when the camera moves from painting to painting along an automatically-generated path, the view will contain the next painting along the path as soon as possible.

System Design

The system design is structured as modularly as possible. There is a central database and rendering process that interprets inquiries about object positions and renders the 3D scene from a specified point of view. This central process also has the ability to calculate the radiosity values for a particular model so that photorealistic images can be generated. It also supports database inquiries about geometric relationships in the virtual world, such as point-to-point visibility. The virtual museum is implemented using the 3D system, and all the processes are implemented based on the TCL/TK toolkit [5].TCL is extensively used because this interpreted language facilitates rapid prototyping. We have enhanced

TCL by coding some additional features in the C language, which also provides improved performance.

Pathplanning

The pathplanning process actually uses several different algorithms that combine to produce a single path through the museum extremely rapidly. The task is broken down into subtasks that can be performed in parallel, and as much precomputation is performed as possible. The problem of traveling from one point in the museum to another point is decomposed first into finding which doors to travel through. This algorithm is optimized for finding paths that originate or terminate at a doorway, so another algorithm must be used to navigate from one point to another point within a room. This second algorithm can also deal with a dynamic environment.

Finally, in a similar fashion to automatic point-to-point navigation, the "tour planner" will generate smooth and efficient tours, and the interface divides the problem up in several stages. First, an algorithm locates all the rooms that are slated to be visited. Then, an exhaustive search is made of all the paths that connect each room. This is an exponential-.e algorithm, but since there is a relatively low branching factor for each room (as well as a fairly small number of rooms), the algorithm is fast enough to be used interactively. After the rooms have been ordered, the paintings within each room need to be ordered based on the entry and exit door (i.e., the system will first visit the painting which is closest to the door from which the room is entered, and it will always visit last the painting next to the exit door). At this point we have a list of paintings that will be visited in the order specified.

References/Publications/Theses

1. Appino, P.A., et al., An Architecture for Virtual Worlds. *Presence: Teleoperators and Virtual Environments*, March 1992, 1(1), pp. 1-17.

2. Chen, D. and D. Zeltzer. *The 3D Virtual Environment/Dynamic Simulation Language*, Computer Graphics and Animation Group Technical Report, MIT Media Lab, August 1992.

3. Johnson, M. *Build-a-Dude: Action Selection Networks for Computational Autonomous Agents*, M.S. Thesis, February 1991, Massachusetts Institute of Technology.

4. Maes, P., Situated Agents Can Have Goals. *Journal of Robotics and Autonomous Systems*, 1990, 6(1&2), pp. 49-70.

5. Ousterhout, J. K. TCL: An Embeddable Command Language in *Proc. 1990 Winter USENIX Conference*, 1990.

6. Zeltzer, D. and M. Johnson. Motor Planning: Specifying and Controlling the Behavior of Autonomous Animated Agents. *Journal of Visualization and Computer Animation*, April-June 1991, 2(2), pp. 74-80.

7. Zeltzer, D. and M. Johnson. Virtual Actors and Virtual Environments: Defining, Modeling and Reasoning about Motor Skills, in *Interacting with Virtual Environments*, L. MacDonald and J. Vince, eds., John Wiley & Sons, Chichester, England, in press.

8. Zeltzer, D., S. Pieper, and D. Sturman. An Integrated Graphical Simulation Platform in *Proc. Graphics Interface 89*, June 19-23, 1989, pp. 266-274.

MULTIMODAL NATURAL DIALOGUE

Relevant Personnel:

Work in this area was completed under the direction of **Dr. Richard A. Bolt**, Senior Research Scientist and member of the Media Lab's Perceptual Computing Group.

* * * * * * *

Larger screen as gestural arena

Up to now, our work in gesture has been carried out in the presence of a 19-inch workstation screen. The graphic resolution has been 1024 by 1280 pixels, affording good resolution; the speed of our graphics engine has been fast (though not outstandingly so; this we hope to remedy our second, follow-on year). Thus, while the graphics display is by ordinary measures quite adequate, the scale of the screen insofar as it forms the space before which gestural actions are performed is less than adequate.

We have radically re-arranged our laboratory space to install a wall-mounted, six-foot diagonal back-projection display. With our 19-inch desktop workstations monitor, and the user situated at about 20 inches from the screen (the normal "working distance," the angular arena was about 21 degrees. In contrast, with our 6 foot on diagonal wall-mounted screen, and the user about three feet away, we have an angular arena of about 40 degrees. Although the 1024 by 1280 pixels are spread over an area 1300% larger with a concomitant trade off in apparent resolution, in user-action space we have gained a great deal. The user has gained an area for gesture that measures approximately five and one half feet across, and perhaps three feet high. This gain in effective gestural space is the result not only of having a larger screen, but also of being able to *stand before it*. The sense of "surround" is dramatically increased. The "volumetric" sense of space gained is

much greater, this more commodious space inviting two-handed descriptive gestures in addition to any pointing gestures the user may care to make.

This "freeing-up" of the gestural space has enabled us to move boldly into the exploration of two-handed gesture, whereas the scale and space afforded by the 19-inch screen milieu positively discouraged such expansiveness, and at best encouraged merely pointing at objects on display, a gestural act requiring only the most meager of spaces.

In later phases of this project, we shall re-gain the resolution lost in this trade-off by shifting the site on interaction to larger displays (like our 2x6K display, and subsequent successors), and have the best of both worlds in display detail and user action.

Multimodal parsing

Top-down, bottom-up parsing

We have adopted a top-down, bottom up approach to the parsing of our multimodal input in speech, gesture, and gaze. The basic rule is first to look to the speech input to determine whether it is semantically complete in itself. For example, in a tactical planning scenario, the user before the display might say, "Put the green helicopter unit next to the blue tower." If there is only one helicopter unit which is colored green, and only one tower which is colored blue, then all of the relevant information to carry out the user intention is present in the utterance itself.

The user may have made some kinds of gestures, and in fact have looked here or there on the display, but the present strategy is to not interpret it, nor act upon it. The simplifying assumption, for the present, is that the person primarily means what they say, and that accompanying, and possibly contradictory, actions in gesture and gaze, are "noise."

At the current stage of our work, the interesting case is where the input in speech is semantically incomplete, for example, as in "Move those units over this way." There is

information missing, namely: 1) which units; b) which way to move them. The strategy where the meaning from speech is incomplete is to search the situation which in our case is to look at the accompanying actions in hands and eyes.

First, we need to determine what is meant by "...those units". We need to establish the referent. At the level of words, it involves a plural reference; more than one unit is meant. We have to examine what happened in eyes, and/or hands, at the time the phrase "...those units" was uttered.

Enabling Plural Reference: the Gestalt Perception Simulator

Using deitic (pointing) gesture to select a single item on display is straightforward enough: it is the item that the user is pointing and/or looking at when they saw something like "What is the status of that (pointing) unit...??". There can be minor complications, such as when the user is pointing at one thing, but looking at another. One strategy here is to have the system act upon the plausible assumption that manual pointing is more "effortful" than is glancing at something, and decide in favor of the hand's signal. There are a number of further potential complications and subtleties involved in resolving reference to a single item; we shall not go into them here, but rather focus on the case of enabling plural reference in an easy, straightforward manner.

When someone wants to select from a collection of items on display a certain subset, the visual features of the collection of objects may be such to facilitate reference. The desired items may be clustered together, of the same general size, shape, or color, or be arranged in some kind of regular patter. That being the case, a ready way to reference then is not to point at them one after the other saying "that ...that one...and that...", and so forth, but rather sweep one's hand through the locale of the items and say "...those...". It is the *look* of belonging together that prompts the speaker to reference the items in question in this

way; it is the way in which the items are arranged on the display that both prompts and guides the "path" of the sweeping gesture used to indicate them.

The person listening—and looking—at such a referential act is able to understand the reference because they, too, share a similar sense of how things "belong" together; that is, they share the same kinds of gestalt-like perceptions of groupings.

Our aim here is to endow the machine with a sense of gestural principles of perception, so that when the user sweeps their hand amidst a collection of things on display and says, e.g., "Delete those...", the machine will have a similar appreciation of the collection of items as decomposible on the basis of groupings, color, size, shape, and so on.

We have during this past year made refinements to our Gestalt Perception simulator, including a new method of color coding that uses a subjective-based color space to increase the accuracy of the color recognition. This is all part of a strategy to let the computer "perceive" the appearance and groupings of items on display in a manner in which a human observer is like to perceive them. This ability of the machine ought to be particularly useful where the user is, for whatever reason, unfamiliar with the set of items on display and is thus at a disadvantage to refer to them on a semantic basis (by name, e.g., "...all units of the 401st...), or perhaps by having some common involvement, e.g., all having just participated in a common maneuver, and thus linked by mutual involvement, as in "... delete all the units involved in the last attack...").

Timing information in speech

There is a tendency for a speaker who is also making a gesture to lead ever so slightly with the gesture, the words following a split second later (McNeill, 1992). In any event, we need to know when, in real-time, the user spoke that phrase relative to any actions by

hand or eye. In our current speech recognition work, we are using a beta test site version of the HARK™ speech recognition tool by BBN Systems, Inc.[1] HARK™ is a speaker-independent, connected speech recognition system. At our request, a special adaptation was made in the system to output timing information relative to the internal clock of the HARK™ system's host processor (an SGI Iris Indigo). Since all the other processors on our system which handle gestural input (via DataGloves™) or gaze input (via an ISCAN™ head-mounted eyetracker) have internal clocks which we can synchronize each with one another, we have the essential timing information to synchronized events in all three modes of speech, gesture and gaze.

Resolving multimodal reference

We have developed a demonstration scenario that incorporates gesture and speech (gaze temporarily left out) in a series of actions which involve the following types of gesture:

- deitic - a "pointing" gesture
- iconic - where the hand acts like, or represents the thing referenced

Suppose the display bears a simple cut-away view of a room of a house. There is a wall on the left, on the right, and a wall to the rear. The color of the wall on the right is currently red, and the user would like to make it be green instead. The user says: "Color that wall green...". Now, our approach is that the machine should not necessarily be looking for a "template" pointing gesture to find out which wall the user meant to have a change in color. That is, the machine should not be looking at the data from the glove sensor and try to process that data against a "template" representing the canonical

---

[1]We acknowledge the kind assistance of Dr. John Makhoul of BBN in making HARK™ in beta test site form available to us for our work.

pointing gesture: the index finger pointer extended, the remaining fingers curled, the thumb closed about the curled fingers. The user may be pointing in the classical, canonical sense; or, they may not. We, or the machine, does not know, cannot know, how the user may act in any particular situation. Nor—importantly— should the machine insist that the user gesture in any particular way. To insist upon canonical gestures, to have a set of "templates" representing conventional gestures with which to screen current input seems to us a very limiting approach.

Rather, our strategy is to determine what is in doubt, what information is missing, using a) the input; b) the structure of the visual display. In our example, the uncertainty concerns which wall. There are only three walls: the one to the left, the rear wall, the one to the right. This is the uncertainly we have to resolve. What we next do is to go down into the "gestlet" data—the raw data output by our DataGloves condensed and processed to the level of elemental shape and elemental attitudes and action - and see if there is any evidence there to distinguish among our three possibilities.

This evidence need not be a canonical posture, attitude, or action of the hand. We are looking, rather, for the slightest bit of evidence to favor one hypothesis over the other. Suppose, at the appropriate time window—on, or about when the user uttered "...that...", the user jerked their hand a bit to the right and ventured a slight swing of the eyes to the right, and back again. The evidence from the hands is small, fuzzy; it is an "hint" of directionality to the right, but not a large, definitive indication. Similarly with the eyes: the action of the eyes is suggestive of an indication to the right, but not an absolute indication. However, when we consider the weight of the *combined* evidence–a twitch of the hand to the right, accompanied by an ever so slight swing of the eyes to the right—all considered in the context of the limited uncertainty we have to resolve (either left, right, or rear wall), then a very reasonable inference is that it is the right hand wall that is intended.

The inferential, top-down, bottom-up approach allows a number of things. First, it does not limit the kinds of gestures indication to just those that mirror a conventional gesture. In fact, it is not gestures per se that we are trying to recognize, but rather we are trying to resolve questions, set largely by the content of the user's utterance in the context of the situation as determined by the contents of the graphical display. Second, this approach let's the user "be themselves." It does not demand any particular set of gestures or stylized forms of action. For instance, in this last example, the machine does not insist that the user "point" is any particular way, as long as they give some kind of gestural act from which can be wrung out one of three general directions: left, right, or back. There needs to be *some* indication, but no *particular form* of it. Third, this approach may well go a long way to handle "cross-cultural" differences in gestural styles, where a person of one background, for example, is inclined to gesture directionality via a nod or swing of the head, a person of another background favor a shift of the eyes, another some gesture by hand. It also may begin to address intra-individual shifts in style; while it is natural to assume interpersonal and/or cross-cultural differs in gestural style, we may encounter changes or shifts in gestural style within the same person, across topics or even within the same session of interaction with the machine. Whatever the manner, or change in manner exhibited by the user of the machine, given this kind of "anti-template," interpretive approach is flexible to meet the situation.

Agents

We have begun to specify the appearance and actions of our planned on-screen agents. These agents will have graphical "eyes," that establish eye contact with the user, and serve to orchestrate the interaction of the agent and the user, just as eye contact and eye breaks orchestrate human person-to person dialogue. An agent will also have graphical arms and hands with which to express gestures and will talk to the user via synthesized speech.

In our prospective scenarios involving such on-screen interface agents, the agents will be repositories both of interactive dialog expertise and subject specific knowledge. The former sort of knowledge is represented by the rudimentary but powerful ways in which people deal with each other using speech, gesture, and gaze. For example, the agents will look back and forth between the user and the things the user is talking about, thus establishing a sense that they (representing the machine intelligence at the interface) in fact "follow" what the user is dialoguing about.

On the output side, when the agent is telling the user about some aspect of some item or some aspect of the situation on display, the agent will look back and forth between what they are talking about, the direction of their gaze representing which area of sector of the display screen is relevant. They will, through the systems ability to track the eye of the user, have a sense of whether the user is looking in the appropriate spot of the display as it (the agent) talks about some thing or other. Should the user not seem to be paying attention to the appropriate spot, a reasonable inference is that the user is not (for whatever reason) following the discussion, and the agent may choose to (diplomatically) remind the user where to direct their attention. Similarly, the agent will have a practical knowledge of where and how to gesture when it is presenting or discussing some aspect of the situation on display.

The agents will possess inter-agent socio-linguistic skills. That is, they will have knowledge of rudimentary behaviors regarding how they should interact with one another in the presence of the user to form part of an intelligent, intelligible dialogue. The behaviors will include a wide variety of facial animations through with to establish and maintain patterns of attention and connection with the user. To this end, we have developed this last year an initial implementation of a hierarchical method of response control and (facial) feature abstraction, to be extended and amplified in our projects' second year.

References/Publications/Theses

Koons, David B., Carlton J. Sparrell & Kristinn R. Thorisson. Integrating simultaneous input from speech, gaze, and hand gestures. In: M. Maybury (ed.), Intelligent Multi-media interfaces. Menlo Park, CA: AAAI Press. In press.

McNeill, David. *Hand and mind: what gestures reveal about thought.* Chicago: University of Chicago Press, 1992.

Sparrell, Carlton J. *Coverbal iconic gesture in human-computer interaction.* Unpublished master's thesis, MIT, 1993.

DISTRIBUTION LIST

addresses	number of copies
MR. RICHARD T. SLAVINSKI RL/C3AB BLDG #3 525 BROOKS ROAD GRIFFISS AFB NY 13441-4505	25
DR. R. BOLT MIT MEDA LAB 20 AMES STREET 3 CAMBRIDGE MA 02129	5
RL/SUL TECHNICAL LIBRARY 26 ELECTRONIC PKY GRIFFISS AFB NY 13441-4514	1
ADMINISTRATOR DEFENSE TECHNICAL INFO CENTER DTIC-FDAC CAMERON STATION BUILDING 5 ALEXANDRIA VA 22304-6145	2
ADVANCED RESEARCH PROJECTS AGENCY 3701 NORTH FAIRFAX DRIVE ARLINGTON VA 22203-1714	1
NAVAL WARFARE ASSESSMENT CENTER GIDEP OPERATIONS CENTER/CODE QA-50 ATTN: E RICHARDS CORONA CA 91719-5000	1
HQ ACC/DPIY ATTN: MAJ. DIVINE LANGLEY AFB VA 23665-5575	1
WRIGHT LABORATORY/AAAI-4 WRIGHT-PATTERSON AFB OH 45433-6543	1

WRIGHT LABORATORY/AAAI-2 1
ATTN: MR FRANKLIN HUTSON
WRIGHT-PATTERSON AFB OH 45433-6543

AFIT/LDEE 1
BUILDING 642, AREA B
WRIGHT-PATTERSON AFB OH 45433-6583

WRIGHT LABORATORY/MTEL 1
WRIGHT-PATTERSON AFB OH 45433

AAMRL/HE 1
WRIGHT-PATTERSON AFB OH 45433-6573

AUL/LSE 1
BLDG 1405
MAXWELL AFB AL 36112-5564

US ARMY STRATEGIC DEF 1
CSSD-IM-PA
PO BOX 1500
HUNTSVILLE AL 35807-3801

COMMANDING OFFICER 1
NAVAL AVIONICS CENTER
LIBRARY D/765
INDIANAPOLIS IN 46219-2189

CMDR 1
NAVAL WEAPONS CENTER
TECHNICAL LIBRARY/C3431
CHINA LAKE CA 93555-6001

SPACE & NAVAL WARFARE SYSTEMS COMM
WASHINGTON DC 20363-5100

1

CDR, U.S. ARMY MISSILE COMMAND
REDSTONE SCIENTIFIC INFO CENTER
AMSMI-RD-CS-R/ILL DOCUMENTS
REDSTONE ARSENAL AL 35898-5241

2

ADVISORY GROUP ON ELECTRON DEVICES
ATTN: DOCUMENTS
2011 CRYSTAL DRIVE, SUITE 307
ARLINGTON VA 22202

2

LOS ALAMOS NATIONAL LABORATORY
REPORT LIBRARY
MS 5000
LOS ALAMOS NM 87544

1

AEDC LIBRARY
TECH FILES/MS-100
ARNOLD AFB TN 37389

1

COMMANDER/USAISC
ATTN: ASOP-DO-TL
BLDG 61801
FT HUACHUCA AZ 85613-5000

1

AIR WEATHER SERVICE TECHNICAL LIB
FL 4414
SCOTT AFB IL 62225-5458

1

AFIWC/MSO
102 HALL BLVD STE 315
SAN ANTONIO TX 78243-7016

1

SOFTWARE ENGINEERING INST (SEI) 1
TECHNICAL LIBRARY
5000 FORBES AVE
PITTSBURGH PA 15213

DIRECTOR NSA/CSS 1
W157
9800 SAVAGE ROAD
FORT MEADE MD 21055-6000

NSA 1
E323/HC
SA82 DOOR 22
FORT MEADE MD 21055-6000

NSA 1
ATTN: D. ALLEY
DIV X911
9800 SAVAGE ROAD
FT MEADE MD 20755-6000

DOD 1
R31
9800 SAVAGE ROAD
FT. MEADE MD 20755-6000

DIRNSA 1
R509
9800 SAVAGE ROAD
FT MEADE MD 20775

DIRECTOR 1
NSA/CSS
R08/R & E BLDG
FORT GEORGE G. MEADE MD 20755-6000

ESC/IC 1
50 GRIFFISS STREET
HANSCOM AFB MA 01731-1619

ESC/AV 1
20 SCHILLING CIRCLE
HANSCOM AFB MA 01731-2814

FL 2807/RESEARCH LIBRARY 1
DL AA/SULL
HANSCOM AFB MA 01731-5000

TECHNICAL REPORTS CENTER 1
MAIL DROP D130
BURLINGTON ROAD
BEDFORD MA 01731

DEFENSE TECHNOLOGY SEC ADMIN (DTSA) 1
ATTN: STTD/PATRICK SULLIVAN
400 ARMY NAVY DRIVE
SUITE 300
ARLINGTON VA 22202

AL/LRG 1
ATTN: MR. M. YOUNG
WRIGHT-PATTERSON AFB OH 45433-6503

AL/CFHV 1
ATTN: DR. B. TSOU
WRIGHT-PATTERSON AFB OH 45433-6503

AL/HED 1
ATTN: MAJ M. R. MCFARREN
WRIGHT-PATTERSON AFB OH 45433-6503

WL/XPK/AAA-2 1
ATTN: DR. D. HOPPER
WRIGHT-PATTERSON AFB OH 45433-6503

AFIT/ENG 1
ATTN: MAJ P. AMBURN
WRIGHT-PATTERSON AFB OH 45433-6503

USA ETL 1
ATTN: MR. R. JOY
CEETL-CA-D
FT BELVOIR VA 22060

SPAWAR 1
ATTN: MR. J. PUCCI
CODE 3241D
2511 JEFFERSON DAVIS HIGHWAY
WASH DC 20363-5100

ATZL-CDC-B 1
ATTN: CAPT CHARLES ALLEN III
FT LEAVENWORTH KS 66027-5300

NUSC 1
ATTN: MR. L. CABRAL
NEWPORT RI 02841-5047

NOSC (CODE 414) 1
ATTN: MR. P. SOLTAN
271 CATALINA BLVD
SAN DIEGO CA 92151-5000

NTSC (CODE 251) 1
ATTN: MR. D. BREGLIA
12350 RESEARCH PARKWAY
ORLANDO FL 32826-3224

DR. M. NILAN 1
4-214 CENTER FOR SCIENCE &
TECHNOLOGY
SYRACUSE NY 13244-4100

THE MITRE CORP 1
ATTN: DR. H. VERON
MS E073
BURLINGOTN RD
BEDFORD MA 01750

NASA 1
ATTN: DR. J. ROBERTSON
LANGLEY RESEARCH CENTER
MAIL CODE 152E
HAMPTON VA 23665-5225

DECISION SCIENCES & ENGINEERING 1
SYSTEMS
ATTN: DR. A. WALLACE
5025 CII/RPI
TROY NY 12180-3590

NAVAL OCEAN SYSTEMS CENTER 1
ATTN: GLEN OSGA, PHD
CODE 444
SAN DIEGO CA 92152

ARI FORT LEAVENWORTH 1
ATTN: MAJ ROB REYENGA
P. O. BOX 3407
FT LEAVENWORTH KS 66027-0347

ARMY RESEARCH INSTITUTE 1
ATTN: SHARON RIEDEL, PHD
P. O. BOX 3407
FT LEAVENWORTH KS 66027-0347

ARI FIELD UNIT 1
ATTN: JON J. FALLESEN
P. O. BOX 3407
FORT LEAVENWORTH KS 66027-0347

RL/ESOP 1
ATTN: MR. JOSEPH HORNER
HANSCOM AFB MA 01731

JOINT NAT'L INTEL DEVELOP STAFF 1
ATTN: CAPT JOHN HILBING
4600 SILVER HILL ROAD
WASH DC 20389

GEORGE MASON UNIVERSITY 1
ATTN: MS LEE EHRHART
CENTER OF EXCELLENCE IN C3I
SCHOOL OF INFO TECH & ENGRG
FAIRFAX VA 22030

MISSION

OF

ROME LABORATORY

Mission. The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

a. Conducts vigorous research, development and test programs in all applicable technologies;

b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;

c. Provides a full range of technical support to Air Force Materiel Command product centers and other Air Force organizations;

d. Promotes transfer of technology to the private sector;

e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.