A Virtual Environment System for Mission Planning

David Zeltzer and *Steven Drucker*
Computer Graphics and Animation Group
MIT Media Laboratory
Cambridge, MA 02139

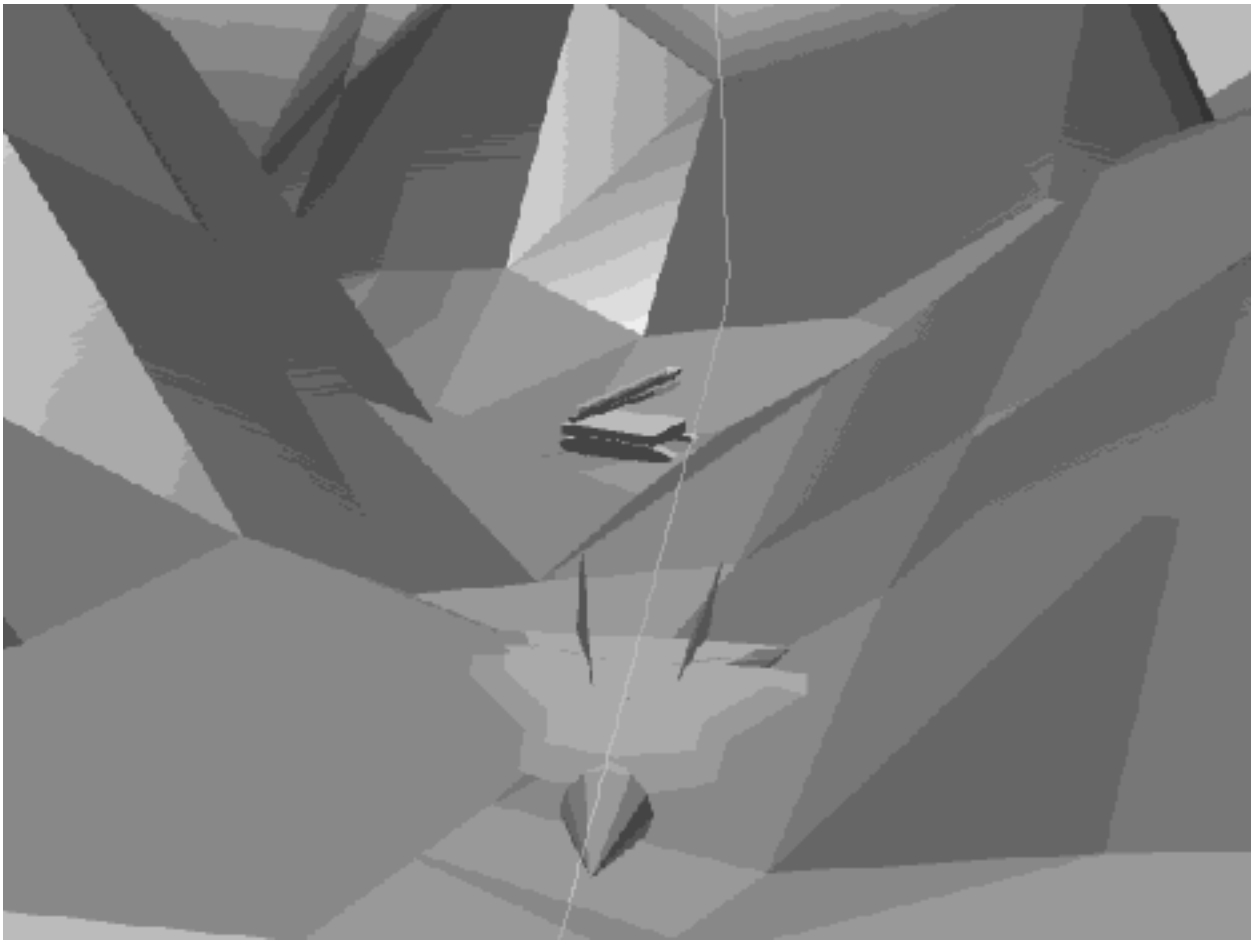

Fig. 1. View along the flight path from an aircraft to a target.

Abstract

A key function of a mission planning system is to enhance and maintain situational awareness of planning personnel and aircrews who will use the system for pre-mission rehearsals and briefings. We have developed a mission planner using *virtual envirnoment* technology. We provide a *task level* interface to computational models of aircraft, terrain, threats and targets, so that users interact directly with these models using voice and gesture recognition, 3D positional input, 3 axis force output, and intelligent camera control.

Presented at the **IMAGE VI Conference** Scottsdale AZ, 14-17 July, 1992

Introduction

A computer-based system for mission planning has a number of objectives. Among them, such a system must:

1) optimize the use of air assets, including manned, remotely-piloted and autonomous vehicles;

2) maximize the probability of mission success as well as aircraft and crew survivability, in part by performing penetration analysis and calculating optimal routes;

3) calculate and output mission data such as fuel and weapon loads, in-flight refueling schedules, and initialize and download mission data to onboard avionics;

4) provide support for mission rehearsal and for briefing aircrews;

5) perform all these functions as rapidly as possible in support of high-intensity, large-scale operations.

Mission planning for tactical operations has been characterized as requiring the "assimilation of large quantities of symbolic and numeric information from a variety of sources under critical time constraints" (Bate and Stanley, 1988). A key function of a mission planning system must be to enhance and maintain the situational awareness of planning personnel and the aircrews who will use the system for pre-mission briefings and rehearsals. Interestingly, one lesson learned in Desert Storm is that aircrews often prefer to study photo imagery rather than maps during mission preparation (Hughes, 1991). Therefore, a mission planning system should support 3D perspective views of photoreal imagery. Finally, while there is a growing body of work on automatic route generation, the problem is characterized by many interacting constraints, so that route-finding algorithms must incorporate heuristics and simplifying assumptions in order to keep the problem size manageable, or to avoid getting stuck in local extrema (Chamberlain and Kirklen, 1988). This means that in any mission planning system — even one in which route generation is partly or fully automatic — it will be important to present easily visualized routing solutions to the human operators, who should find it just as easy to modify and refine them.

For these reasons we have chosen to focus on the human interface to the mission planner we have developed. We assume that some other module may perform optimal route calculations; our system could input these proposed solutions, display them dynamically and in 3D, and make it easy for personnel to interactively modify flight paths, for example, to take advantage of local terrain masking.

Virtual Environments and Task-Level Interaction

Emerging *virtual environment* (VE) technologies offer powerful and productive human interface solutions for command and control problems, and we have set out to demonstrate the utility of VE technology in this domain.

As we have pointed out elsewhere, in addition to the computing hardware and peripherals, a VE consists of three critical components: a set of *computational models*; a *logical interface* that specifies which parameters of these models may be

modified, and when; and a *physical interface* that provides one or more displays (which may be visual, auditory or tactual) and I/O devices by which to communicate the user's intentions to the system (Zeltzer, 1992).

Our work follows a *task-level* analysis of the goals and requirements of a mission planning system, in which workload and stress levels are likely to be very high. It is particularly important in such situations that the computer interface be as transparent as possible, requiring a minimum of computer expertise and programming skills. Operators and aircrews should deal directly with the objects and processes associated with the task, using a vocabulary and sensorimotor skills that are already familiar to them (Zeltzer, 1991).

Unlike a flight simulator, in which interaction is restricted to cockpit tasks, we have designed the system to emphasize task level interaction with the environment: the operator can easily change not only his viewpoint, but all the objects and their positions within the environment. We provide models of aircraft, terrain, threats and targets, and users interact directly with these models — voice recognition for speech input, a VPL DataGlove for positional input and gesture and posture recognition, and a more conventional mouse and keyboard interface are all supported. Finally, we provide a range of sensory displays, including wide field of view (FOV) visual displays, and a force output joystick — a device that can generate force cues for the operator.

The system is implemented within the **bolio** prototype VE system (Zeltzer, Pieper and Sturman, 1989). The **bolio** system maintains a registry of all the objects within the virtual world, and a list of constraints that are activated and satisfied when certain conditions are met. It provides the ability to distribute the system among networked computing platforms, and to incorporate a wide variety of I/O devices. Using the system, processes are assigned to each of the following tasks: a separate process that manages the movements for each aircraft, one process to manage the movement and location of all the targets and threats, a process for controlling and monitoring the DataGlove, a process for controlling and monitoring the force output joystick, a process to manage the menus and mouse input, a process to manage voice input, and the underlying graphical simulation system which displays all the objects in the database and maintains the constraints between the objects. **bolio** is implemented in C on Hewlett-Packard 9000 series turboSRX graphics workstations running the HP-UX operating system, a derivative of AT&T System V Unix™.

Aircraft and Flightpaths

As shown in Fig. 1, when using the mission planner, the operator is presented with a (full color) 3D computer graphic representation of the area of interest; multiple views are available simultaneously. Using 3D input devices, the operator can specify waypoints through which an aircraft should travel, or prespecified waypoints can be read in from an external source. Once defined in either fashion, the aircraft module will generate a flight path which will pass through each of the waypoints, if

possible, based on a simplified aerodynamic model (actually, an A-4 Skyhawk). The system can represent and display an unlimited number of aircraft, each following its own flight path. Motion of the aircraft along flightpaths can be interactively controlled using "VCR" controls that allow the operator to stop action, back up, or fast forward motion as necessary.

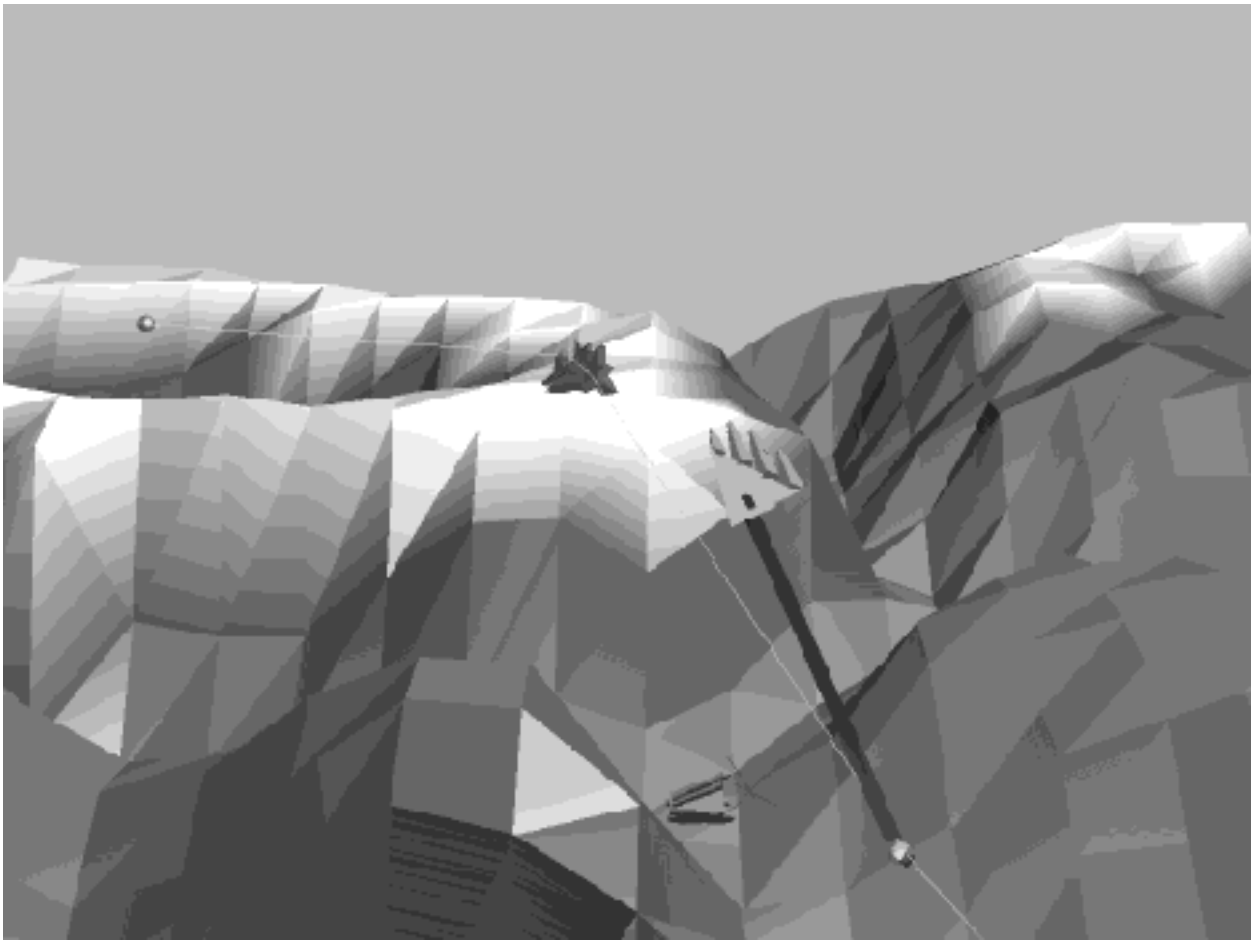

Fig. 2. Using the DataGlove to modify a waypoint.

The aircraft module assumes an initial velocity, and continuously calculates the distance to the next waypoint and the angle of turn necessary to take it to the subsequent waypoint. Based on this information, it performs the necessary maneuvering to maintain the aircraft's position along the flight path. Appropriate state variables are calculated for the entire path including g-load factors and aircraft attitude along the path. G-loads can be displayed as changes in color along the flight path. Since **bolio** supports distributed computation, in order to maintain interactive update rates, aircraft dynamic calculations are offloaded onto a remote platform, in this case, a Hewlett Packard 9000 series 750 workstation.

As shown in Fig. 2, waypoints can be moved interactively using any of the 3D input devices we will discuss shortly. Once a waypoint is moved, the flight path is recalculated and displayed. The user can also specify a

terrain following mode, in which the actual altitude of the waypoints are ignored and the aircraft moves at a specified height above ground level.

The operator can also locate points of interest on the terrain, specify their identity and set the view to originate or terminate at that point, so that lines of sight (LOS) to or from aircraft can be checked. The view can be made to track the motion of any specified aircraft, track a location on the ground from any of the aircraft, or track the motion of a moving object on the ground. This makes it extremely easy to examine LOS between any objects in the system. (Intelligent camera control will be discussed further below). An additional viewing interface includes the use of a head mounted display (HMD) equipped with a motion tracking device which senses an operator's head movements and updates the display appropriately. Not only does this provide motion parallax, an important cue for 3D depth perception, but the HMD has a wide FOV which enhances situational awareness in the virtual world.

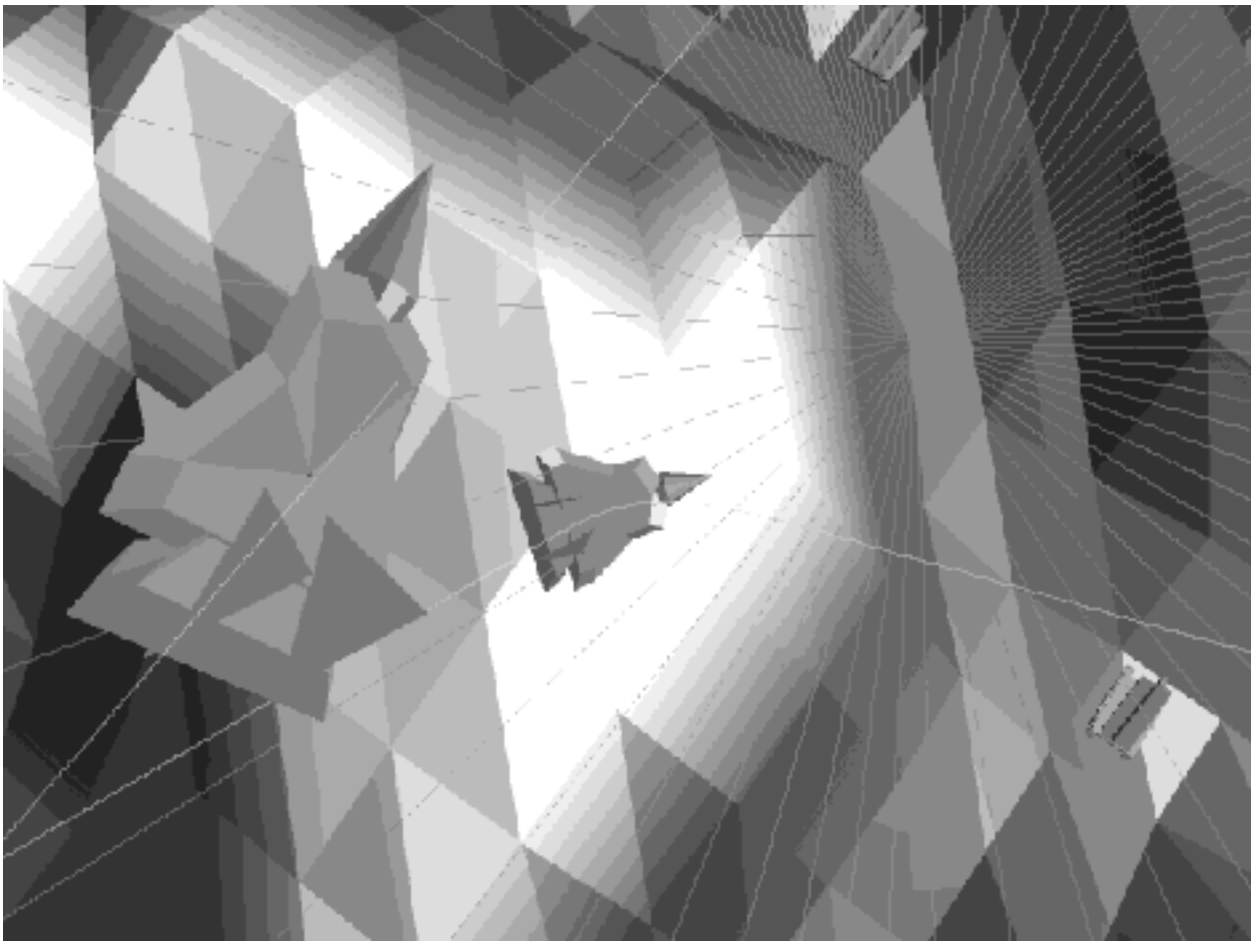

Fig. 3. View from a red A/C toward a blue A/C that has entered terrain masked radar coverage.

Target module/terrain database

Terrain may be derived from aerial imagery using a depth-from-shading algorithm (Pentland, 1990), or terrain objects may be defined and modified interactively using solid modeling tools, as in Figs. 1-4. In principle, terrain information can be derived from other sources, e.g., DMA DTED, but we have not implemented the necessary interfaces.

The terrain model is used by both the aircraft model — in terrain following mode — and by the target module. The target module handles all objects on the ground including threats, radar, stationary structures and moving targets — which are constrained to lie on the terrain surface. Targets can either be loaded in from a database, or specified "on the fly" using any of the 3D input devices.

Using the force output joystick, it is possible to "feel" textured terrain surfaces, or interactively "sculpt" a 3D terrain surface for generating particular scenarios.

Terrain Masked Radar

Terrain masked radar coverage is generated and displayed automatically, and the locations of aircraft with respect to a radar site are continuously tracked. When an aircraft is "visible" to a radar, the color of the aircraft is changed to yellow. See Fig. 3. The algorithms used for generating the radar, and checking the visibility are straightforward. An operator can interactively locate the radar site using any of the 3D input devices, and the terrain is sampled radially outward from this point. The highest terrain elevation in each direction is used as an endpoint, and a connected polyhedron, including the central radar site and all the endpoints, is constructed. This is displayed on the screen in wire frame form so as not to obscure the environment — transparent polygons would look better, but we have not implemented this. To check aircraft visibility, we sample the path between the radar site and the aircraft in question, and compute the vertical angle between the highest terrain feature and the radar site along that path. If at any point the angle between the radar and the sampled terrain value is greater than the angle between the radar and the aircraft, the aircraft must be below the local horizon and is not visible to the radar. This algorithm can easily be run at interactive rates.

Input Devices

There are two primary types of input devices used in the system: those used for specifying discrete command information, and those for specifying continuous three dimensional data. Commands may be input either through a conventional keyboard system, a mouse and menu based system, hand gesture recognition, or a voice recognition system. The mouse/menu/window system is implemented using the X-window system with OSF Motif extensions. The window system supports multiple simultaneous viewpoints, though this is prohibitively slow on our current hardware (Hewlett-Packard 9000 series 800 workstations with turboSRX graphics accelerator). The gesture recognition system uses the VPL DataGlove which incorporates fiber optic cables that sense hand flexion. To specify 3 dimensional input, the

operator can use either the Polhemus Iso-track sensor attached to the DataGlove, the Spatial Systems Spaceball Force/Torque sensor, or a 3-axis force output joystick developed as part of a joint project between the Mechanical Engineering Department at MIT and the Media Lab (Russo, 1990).

The 3-axis force output joystick is really both an input and an output device. It is a conventional joystick that has been equipped with motors and brakes so that it can generate forces on the shaft. Many modes of interaction using the joystick were examined. The user can position targets or threats on the terrain while feeling the texture of the terrain. The targets or threats can be made to repel or attract the joystick giving an overall impression of the environment. The movement of the joystick can be made to follow any of the aircraft as it moved along its flight path. Several of these modes can be combined so that threats can be sensed as the plane was moving along its path. One extension that is not yet implemented is to use the joystick to modify the flightpath while exerting a force to the user proportional to a difficulty involved in having the aircraft follow the newly proposed path.

Voice recognition uses the Articulate Systems Voice Navigator. This is an inexpensive, Macintosh-based system which supports a small vocabulary of about 1200 words. The Macintosh is connected to the main workstation via a serial connection, and acts as a "speech server".

Voice input is processed with a finite state machine so that commands are parsed according to the current context. Available commands are listed on the visual display, and any command that is not appropriate for the current state is ignored. This command "prompting" greatly improves voice interaction. A noise-cancelling microphone is used to alleviate problems with background noise. Currently the voice system is used primarily for view specification, allowing the user to perform a variety of necessary tasks: tracking the movements of any of the objects in the simulation; placing the eyepoint at any object (and keeping it there during movement); requesting overhead and standard views; or adjusting the views based on panning, zooming, or other commands.

An HMD can be used to view the entire scene as an omniscient observer. The user can move along with an object as it moves and the system would track the movements of the user's head and update the view appropriately, generating a feeling of being within the map itself. The wide field of view and the motion parallax, and the stereo display added particularly to this sense of presence.

Intelligent Camera Control

Camera control in the system is designed to be both flexible and powerful. Out-the-window views can be easily directed to any object in the environment, including other aircraft, targets or threats. Views may simply display straightahead out-the-window imagery, or can be made to follow objects as they move through space. At any time, the point of regard and the viewpoint can be swapped. Voice control is incorporated, so all the operations described in this section are available via hands-free operation.

Besides simple viewpoint and point of regard control, camera control based on conventional camera movements are possible. For instance, the camera may pan, truck (camera view point moves toward the point of regard), dolly (camera viewpoint moves perpendicular to the vector connecting the viewpoint and point of regard), crane, and zoom (field of view is increased or decreased). Certain points of view can be saved and retrieved for instantaneous changes to standard reference views.

A more sophisticated level of view control maintains the positions of selected objects at certain points in the frame. For example, rather than generating an out-the-window view of a target, it may be more informative to construct a view external to both the aircraft and the target. This can be seen in Fig. 1, which shows a view from an aircraft to a target, and in Fig. 4, which shows a view from a different target looking back towards another aircraft centered in the frame.

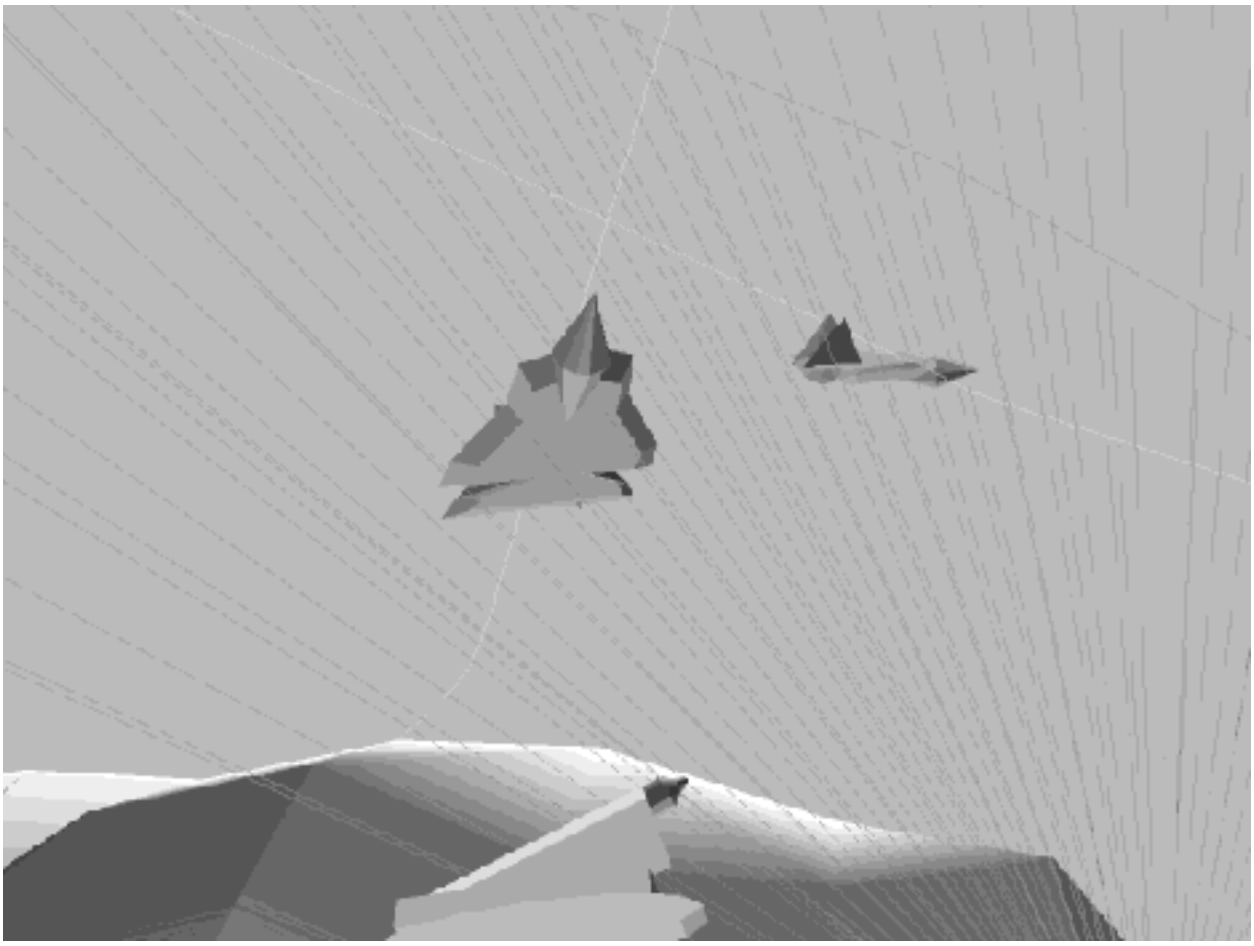

Fig. 4. Centered view showing target, red and blue A/C.

Finally, a system for automatically tracking the closest target or threat is available. This system can be used to merely indicate the closest target via a line between an aircraft and the target, or the point of regard can be automatically changed and tracked so

that the object of regard is *always* the closest target.

Scenario Generation

Mission scenarios can easily be created, stored and retrieved using the system. Waypoints are created using a 3D input device (e.g., DataGlove or force output joystick), and paths through these waypoints can be displayed and edited using a 3D input device. These paths can then be saved in separate files and loaded at a later time. Positions of targets and threats can also be saved and retrieved. “Menu buttons” can be created that load aircraft, threats and targets with a single selection. Complicated scenarios can thus be accomplished by simple scripting without any need for recompilation. Scripting is performed using tcl (tool command language - a public domain front-end scripting language), along with an in-house interface to OSF motif.

Concluding Remarks

While we have not conducted formal evaluations of the system, many visitors, including DoD and Air Force personnel, have seen the system in operation and have commented favorably. We have found that for modifying flightpath waypoints, the combination of gesture recognition with the DataGlove, and positional information from the Polhemus worked quite well. The Spaceball was excellent at modifying the views of the environment by using a “camera in hand” metaphor, which allows the operator to freely move the viewpoint around the virtual world

Voice control has been very encouraging: when tested in a noisy room, recognition was quite good. With the addition of the finite state

machine for command selection, we've had absolutely NO false actions as a result of an occasional “false hit”, i.e., incorreclty recognized command. The worst result is that occasionally one has to repeat a command several times, but more often than not we have had a string of 10 to 40 successful commands in a row.

We are currently in the process of porting the system to faster platforms, and we expect a performance increase of from 4 to 6 times in visual updates due entirely to the faster hardware. In addition, this will allow us to conduct further experiments with the force feedback joystick, since current hardware cannot support realtime force output interaction with dynamic objects, such as flightpaths.

Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency under Rome Laboratories Contract F30602-89-C-0022, and equipment grants from Hewlett-Packard and Apple Computer. Paul Dworkin implemented the force output joystick module, and Amy Pritchett of the MIT Aero and Astro Department implemented the aircraft dynamics module. Special thanks are due to Dick Slavinski of Rome Laboratories for his support and encouragement.

References

Bate, S., and Stanley, K. (1988). Heuristic Route Planning: An Application to Fighter Aircraft. Proc. IEEE 1988 National Aerospace and Electronics Conf, (pp. 1114-20), Dayton OH.

Chamberlain, D. B., and Kirklen, C. R. (1988).Advanced Mission Planning System (AMPS) Employing Terrain and Intelligence Database Support. Proc. IEEE 1988 National Aerospace and Electronics Conf, (pp. 1145-51), Dayton OH.

Hughes, D (1991). Advanced USAF Mission Planning System Will Serve Fighters, Bombers and Transports. Aviation Week & Space Technology, 134(23), pp. 52-57.

Pentland, A. (1990). Linear Shape from Shading. Int'l J. of Computer Vision, Vol. 4, pp. 153-162.

Russo, M.A. (1990). The Design and Implementation of a Three Degree of Freedom Force Output Joystick. M.S. Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Zeltzer, D. (1992)..Autonomy, Interaction and Presence. Presence: Teleoperators and Virtual Environments, 1(1), in press.

Zeltzer, D. (1991). Task Level Graphical Simulation: Abstraction, Representation and Control, in: N. Badler, B. Barsky and D. Zeltzer (Eds.), Making Them Move: Mechanics, Control and Animation of Articulated Figures, (pp.3-33). San mateo CA: Morgan Kaufmann.

Zeltzer, D., Pieper, S., and Sturman D. (1989). An Integrated Graphical Simulation Platform. Proc. Graphics Interface '89 (pp. 266-274), London, Ontario, Canada.