56-16

17

1996

NASA/ASEE SUMMER FACULTY FELLOWSHIP PROGRAM

MARSHALL SPACE FLIGHT CENTER
THE UNIVERSITY OF ALABAMA

INTEGRATION OF THE SHUTTLE RMS/CBM POSITIONING VIRTUAL ENVIRONMENT SIMULATION

Prepared By:	Joseph D. Dumas II, Ph.D.
Academic Rank:	Assistant Professor
Institution and Department:	University of Tennessee at Chattanooga Department of Computer Science and Electrical Engineering
NASA/MSFC:	
Laboratory:	Mission Operations Laboratory
Division:	Training and Crew Systems Division
Branch:	Systems Branch
MSFC Colleagues:	Joseph Hale Richard Dabney

INTRODUCTION

Constructing the International Space Station, or other structures, in space presents a number of problems. In particular, payload restrictions for the Space Shuttle and other launch mechanisms prohibit assembly of large space-based structures on Earth. Instead, a number of smaller modules must be boosted into orbit separately and then assembled to form the final structure. The assembly process is difficult, as docking interfaces such as Common Berthing Mechanisms (CBMs) must be precisely positioned relative to each other to be within the "capture envelope" (approximately ±1 inch and ± 0.3 degrees from the nominal position) and attach properly. In the case of the Space Station, the docking mechanisms are to be positioned robotically by an astronaut using the 55-foot-long Remote Manipulator System (RMS) robot arm. Unfortunately, direct visual or video observation of the placement process is difficult or impossible in many scenarios. One method that has been tested for aligning the CBMs uses a boresighted camera mounted on one CBM to view a standard target on the opposing CBM. While this method might be sufficient to achieve proper positioning with considerable effort, it does not provide a high level of confidence that the mechanisms have been placed within capture range of each other [1]. It also does nothing to address the risk of inadvertent contact between the CBMs, which could result in RMS control software errors. In general, constraining the operator to a single viewpoint with few, if any, depth cues makes the task much more difficult than it would be if the target could be viewed in three-dimensional space from various viewpoints. The *actual* work area could be viewed by an astronaut during EVA; however, it would be extremely impractical to have an astronaut control the RMS while spacewalking. On the other hand, a view of the RMS and CBMs to be positioned in a *virtual* environment aboard the Space Shuttle orbiter or Space Station could provide similar benefits more safely and conveniently with little additional cost.

In order to render and view the RMS and CBMs in a virtual world, the position and orientation of the end effector in three-dimensional space must be known with a high degree of accuracy. A precision video alignment sensor [1] has been developed which can determine the position and orientation of the controlled element relative to the target CBM within approximately one-sixteenth inch and 0.07 angular degrees. Such a sensor could replace or augment the boresighted camera mentioned above. The computer system used to render the virtual world and the position tracking systems which might be used to monitor the user's movements (in order to adjust the viewpoint in virtual space) are small enough to carry to orbit. Thus, such a system would be feasible for use in constructing structures in space.

In order to investigate the properties and limitations of a system providing virtual presence in the vicinity of the end effector of the Shuttle RMS, it was proposed to create a similar virtual environment in the CAVE (Computer Applications and Virtual Environments) laboratory at MSFC. This system would be very similar to the proposed system described above, with the obvious exception that an actual Shuttle RMS could not be used. Fortunately, existing simulation facilities at MSFC include a highly detailed model of the RMS dynamics running on an Alliant FX-8 supercomputer. This model runs in real-time with an input/output loop executing every 40 milliseconds. When provided with actual or simulated six-degree-of-freedom hand controller inputs, the RMS simulation computes the position and orientation of the RMS system in space. The coordinates of the controlled CBM relative to the target, as well as other variables such as RMS arm joint angles, needed to render the visual scene displayed to the operator, are available as simulation outputs updated every 40 milliseconds.

The integration of the "virtual RMS" simulation involved establishing data communications between a number of disparate devices. One key element of the simulation is the set of two hand controllers (identical to those used on the Space Shuttle orbiter) for the RMS. The three-degree-of-freedom controller for the user's left hand is used to input translational (X, Y, and Z axis) positioning commands for the payload on the end effector of the RMS; the right hand controller is a joystick type control used to provide angular orientation commands (roll, pitch, and yaw) for manipulating the payload. A second key element of the virtual environment simulation is the real-time RMS dynamics model described previously. The Alliant supercomputer which runs this model is located in MSFC building 4663, while the rest of the simulation system hardware resides in the CAVE laboratory in building 4610. The third key element of the simulation is a set of three-dimensional graphical models of the RMS, space station modules to be joined (with their CBMs), and surrounding objects such as the orbiter. The purpose of the simulation is for the user to be able to view the process of positioning the controlled CBM from any location in the vicinity of the mechanisms to be joined. This is accomplished by determining the position and orientation of the user's head using a six-degree-of-freedom spatial tracking system (Polhemus Fastrak) and using this information to navigate the user's viewpoint in the three-dimensional virtual environment. The scene is rendered by a high-speed Silicon Graphics Indigo^2 workstation and viewed by the user through a Head-Mounted Display (HMD).

All of the simulation elements just described must be integrated smoothly, and with as little temporal lag as possible, in order to give the user a sense of presence in the virtual world and allow effective control of the payload. At the commencement of the author's Summer Fellowship, procurement of necessary equipment such as the RMS hand controllers, data acquisition board, and SCRAMnet fiber optic data link boards had been accomplished, but no hardware or software integration had been done. The author was tasked with performing the necessary hardware integration; writing the software for interfacing, data communications, and navigation and object manipulation in the virtual environment; and testing and evaluating the simulator system.

APPROACH AND PROGRESS

The basic requirements for integrating the simulation include acquiring user input from the hand controllers, transmitting the hand controller data to the Alliant supercomputer, communicating the RMS state information from the Alliant to the Indigo^2 graphics computer, and sending the user head position data from the Fastrak tracker to the Indigo^2 to determine the viewpoint for the rendered scene. A later enhancement to these requirements was the ability to communicate hand controller inputs to the Indigo^2 as an alternative means of navigating the user's viewpoint.

Several factors complicated the relatively simple scenario just described. The only available interface device capable of translating analog data from the hand controllers into the 16-bit digital values required by the RMS simulation was a National Instruments AT-MIO-16X data acquisition board designed to fit into an IBM PC/AT compatible computer with EISA bus. Communication between the CAVE laboratory and the Alliant supercomputer was to be implemented by a SCRAMnet fiber optic link. The SCRAMnet board is a VMEbus board which is not compatible with the chassis of the PC or the Indigo^2; it could only be installed in another machine, a Silicon Graphics 320 VGX. Thus, the simulation network had to include four computers rather than two: the Alliant, with network name "*cat*"; the Indigo^2, referred to as "*safire*"; the SGI 320 VGX, known as "*cave1*"; and an Intel 486DX-33 based PC called "*rmspc*". All communication between machines other than *cave1* and *cat* was to be done over an Ethernet-based local area network using IP (Internet Protocol). An Internet communication link between *cave1* and *cat* also had to be established for testing and integration of the system pending the installation of the fiber optic link (which has yet to be accomplished at the time of this report).

Besides the hardware differences between the four platforms, development of software was complicated by differences between the various operating systems and language compilers. Both Silicon Graphics machines (*safire* and *cat*) are Unix-based machines; the Indigo^2 (*safire*) runs IRIX 5.3 and ANSI C 3.19 (which were upgraded during the summer causing some delays) while the 320 VGX (*cave1*) runs IRIX 4.0.5 and ANSI C 1.1. The Alliant (*cat*) runs a real-time Unix kernel (Concentrix 5.7.00) and Concentrix C 4.0.17. The 486 PC (*rmspc*) runs MS-DOS 6.22; PC software was developed using Borland's Turbo C++ 3.0. For the most part, C language source code is compatible between the two SGI machines. A few modifications had to be made when coding for the Alliant, but the network function calls adhered to the same RPC (Remote Procedure Call) standard used on the SGI machines. The network function calls for the PC used Novell's LAN WorkPlace for DOS Socket Library, which is quite different from RPC; this complicated programming somewhat.

One of the first tasks in the integration of the simulation was to outfit a personal computer for the task of hand controller data acquisition. The other computers were already in place and connected to the Internet via local area networks. To complete the hardware test bed, however, a suitable PC (Intel 486-based) had to be located and a network interface card and software drivers had to be installed. With the network hardware and drivers and C compiler in place, test programs were developed to acquire and display analog data using the NI-DAQ (National Instruments Data Acquisition) library functions. Subsequently, code was added to broadcast the acquired data to the other machines using UDP (User Datagram Protocol, a connectionless, unreliable messaging protocol for sending data over the Internet). Corresponding code was written for the Silicon Graphics 320 VGX to accept data from the PC and relay it to the other computers.

In testing the UDP-based communications programs, it became evident that messages were arriving out of order on a regular basis, particularly in the case of communications to the Alliant (which is not on the CAVE lab local area network (LAN)). In many cases the exit message from the PC, which is used to terminate the simulation on the other three machines, would be lost and the programs on other machine(s) had to be aborted manually. Considering these problems and the fact that other data communications to and from the Alliant are done using TCP (Transmission Control Protocol, a connection-oriented messaging protocol which is slower but more reliable than UDP), it was decided to rewrite the communications code for all four machines using TCP sockets. Once debugging was completed, the TCP-based programs offered the advantage of easy two-way communication between machines (since sockets are bidirectional) in addition to enhanced reliability due to guaranteed message delivery.

While software development was in progress, a user interface station (the "Virtual Reality Chair") was being designed and constructed by MSFC support personnel. This user station consists of a wooden chair modified with two special arm rests on which the RMS hand controllers are mounted. A wooden chair was chosen in order to minimize the amount of metal in the simulation test bed area; metal objects can interfere with the performance of the Polhemus Fastrak magnetic spatial tracking system used to track the user's head movements. The arm rests can be pivoted to allow the user to easily enter and exit the chair, and allow him/her to adjust the lateral spacing between the controllers. The arm rests can also be adjusted vertically in order to accommodate users of varying heights. Finally, the hand controller mounts are slotted to allow them to be adjusted to various positions closer or farther away from the user. This reach adjustment helps to facilitate the comfortable use of the "VR chair" by operators of varying physical sizes. MSFC and contractor support personnel also assisted with manufacture of the cables used to connect the hand controllers with the data acquisition board and the Fastrak unit with the Indigo^2.

Additional software development was needed in order to extract the RMS state vector and joint angles from the real-time simulation on the Alliant. Bob Linner of MSFC was asked to assist with this part of the project since he is most familiar with the simulation. Delays in completing this part of the overall simulation loop threatened to stall progress on the project. In the interim, a program 'comm50' was written to run on *cat*. It supplies fixed joint angles so a stationary RMS arm can be rendered, and echoes the hand controller data sent to it (after converting the values to floating-point format) so the controllers can be used to "fly" the simulated arm directly. This bypasses the RMS dynamics, reducing the fidelity of the simulation, but allowed development of the communications and graphics routines to proceed.

With communications among the various machines established, additional software routines were needed to create the graphical environment and manipulate the viewpoint and payload on the image generation computer *safire*. The simulation program on *safire* was developed using Sense8's WorldToolKit (WTK), a virtual environment development package which is available for a number of computing platforms including SGI machines. Three-dimensional graphical models of the objects of interest had already been developed by Mark Slone of MSFC. These models were defined as "C" code including SGI Graphics Library (GL) calls; thus, aside from some scaling and minor modifications to suit the application, they could be used directly on the SGI Indigo^2. However, since the models were developed and integrated as GL code rather than WTK objects, WTK sensor objects could not be attached directly to the payload. Instead, software was developed to convert between GL and WTK coordinate systems and manipulate the object representing the payload. Since the user viewpoint is not attached to any graphical entity in the virtual world, it can be manipulated by WTK sensor update functions. Specifically, any of three WTK sensor objects can "fly" the viewpoint: the Polhemus Fastrak, the mouse, or the hand controllers (when set to "COARSE" mode only; "VERNIER" mode manipulates the RMS and its payload). The Fastrak and mouse drivers are supplied with WorldToolKit; the hand controller driver was developed for this project. It is also possible to add other input devices such as a Spaceball to manipulate the viewpoint for the simulation if desired, though the primary method envisioned for this is the use of the hand controllers for gross movements and the head tracker for fine adjustments of position and orientation.

A simulation is initiated by running the simulation coordinator program on the Silicon

Graphics 320 VGX computer (*cave1*). This program, 'simcave1', acts as a TCP/IP server, accepting socket connections from each of the other three computers. Simcave1 first waits for a connection from the data acquisition computer (*rmspc*). This will occur when the user runs the program 'simrmspc' from the MS-DOS prompt. Once the socket to *rmspc* is connected, *cave1* waits for and accepts a similar connection from the Alliant (established by running 'comm50' on *cat*) and finally from the Indigo^2 (the user should run 'simsafire' on *safire*). When all three sockets are connected, *cave1* enters the main program loop, which serves to sequence the order of events and transfer the data required for each frame of the simulation. This continues until the user strikes a key on the PC keyboard, or until an unrecoverable communications error occurs (which is rare). The user keystroke causes a special flag to be set which is passed to *cave1* and thence to the other two computers, causing them to exit their programs.

CONCLUSIONS AND RECOMMENDATIONS

The virtual environment simulation described above was integrated over a 10-week period from May 20 - July 26, 1996. While some delays were encountered in obtaining hardware components and software to interface to the RMS simulation on the Alliant supercomputer, all data communications functions necessary for integration of the simulation were accomplished during the Fellowship period. Successful operation of the virtual environment simulation (minus the RMS dynamics) has been demonstrated. A user in the "VR chair" wearing a head-mounted display experiences an immersive graphical environment which reacts to user navigational and control inputs. Users of the simulator are able to control the positioning of the space station module and its CBM in a manner similar to the boresighted camera approach if desired, by simply observing the target CBM and manipulating the hand controllers. The real advantage of the simulated environment, however, is the ability of the user to adjust the viewpoint position and orientation using head movements and hand controller or mouse commands and thus get a three-dimensional, close-up view of the capture surfaces.

Possible follow-on work would include obtaining or developing more detailed three-dimensional models of the CBM in order to add greater realism to the simulation and give the user a virtual experience even more similar to the actual task. It also would be desirable to more thoroughly evaluate the system's performance (particularly in terms of frame rate and temporal lag) with a view towards improving those parameters and enhancing the user's perception of immersion in the virtual environment. Ultimately, human factors experiments should be designed and conducted to compare the system with the boresighted camera docking approach. Many other behavioral and training experiments might be devised to take advantage of this unique simulation testbed.

REFERENCES

[1] Dabney, R., Hale, J., and Linner, B. (1993). "An Interactive Stereo Graphic Robotic Manipulator Control Interface." Proposal for the 1993 Marshall Space Flight Center Director's Discretionary Fund, August, 1993.