

## AN IMMERSIVE VIRTUAL ENVIRONMENT TRAINING SYSTEM ON THE REAL-TIME MOTION PLATFORM

Jian Chen<sup>\*</sup>, Yung-Chin Fang<sup>\*</sup>, R. Bowen Loftin<sup>\*\*</sup>, Ernst L. Leiss<sup>\*</sup>, Ching-yao Lin<sup>\*</sup> and Simon Su<sup>\*</sup>

<sup>\*</sup>Department of Computer Science

<sup>\*\*</sup>Virginia Modeling Analysis & Simulation Center

University of Houston

Old Dominion University

Houston, Texas 77204

Suffolk, Virginia 23435

{jchen2, coscel, chingyao, ssu}@cs.uh.edu, yfang2@bayou.uh.edu, bloftin@odu.edu

### ABSTRACT

This work is focused on the implementation and integration of virtual environment (VE) technology to support the future Human Mars Mission (HMM) personal training conducted by the National Aeronautics and Space Administration (NASA). We present a six degrees of freedom (DOF) motion platform, or *Flostation*, plus a head mounted display (HMD) based virtual environment training system, where the motion platform is used to simulate rover movement and the HMD coupled with a head tracker and a joystick to support interaction. This research demonstrates that we can achieve not only real-time interaction performance but also high level realism in our virtual environment application. The prototype system was developed on an SGI Onyx equipped with an InfiniteReality II, a two pipe graphics system.

**KEYWORDS** virtual environment, 6-DOF motion platform, training, real-time

### 1. INTRODUCTION

Historically, immersive virtual environment (VE) simulation has been being explored as a training solution for a variety of areas, such as military multi-user interaction [1,2,3], repairing the Hubble Space Telescope [4], shipboard fire fighting [5], harbor and channel ship handling [6], disaster mitigation [7], medical procedures [8] and so on. Just to mention a few, the SIMNET architecture [1] for training and rehearsing battlefield operations has been in use now for many years. Analogous systems, such as NPSNET [2] and the High-Level Architecture [3] are used in research into large-scale distributed mission training systems for war fighting. Many of these efforts gained positive results. They also proved that VE techniques can be used to assess trainees' abilities and skills and to improve trainees' behavior in the real world while reducing

training time, costs and errors. In particular, NASA/Johnson Space Center (JSC) contributed to the first large-scale VE system used in training for its space mission, which was the Hubble Space Telescope (HST) repair and maintenance mission [4]. Its success and effectiveness further broadened and deepened the interests of NASA in exploring VE technologies as a standard training tool.

More recently, it seems that the capabilities of rapid advancement in interactive technologies have been employed in VE for training processes. One of the aspects is their ability to present the trainee with multiple input and output devices, which serve as human-computer interfaces (HCI) to interact with virtual objects and lead to enhance the trainee's performance. In a training system, training tasks are usually goal-driven. Different tasks should meet different goals and have different device components. These devices are combined together to form a simulator, which is often uniquely designed for particular purposes.

The main device of our simulator is a six degrees of freedom (6DOF) motion platform, or *Flostation*, which is primarily designed to replicate real world Mars rover movement in a virtual scenario. In order to enhance the perception of realism, our system was designed to support device manipulation (e.g., manipulating the joystick and tracking head orientation), 'terrain following' navigation (e.g., a trainee can travel around the virtual 3-D Mars terrain freely), real-time rendering of the stereoscopic view on HMD and simulated stereo acoustic effects. The goal is to offer an accessible and cost-effective VE that simulates planned Extravehicular Activities (EVA) for training astronauts who currently receive little or no experiential practices for their assigned tasks. In addition, this work has the potential to save lives and reduce mission errors.

It is problematic to synchronize a user's motion in a real space and a virtual space when multi-device integration is involved. Because in situations where the time lag between these spaces is greater than 100ms, the trainee tends to feel cyber sickness and perceives such a

VE as less interactive and realistic. Also, it is difficult for the trainee to maintain involvement. One of the main strategies to guarantee an adequate software response time for this type of VE simulation is to perform a movement permutation simulation and compute all the possible paths, human delay, frame rate and hardware response time for all possible data points. This approach is very time consuming and lacks flexibility. When changes in the model or device occur, all simulations need to be re-computed. The system described in this paper can meet soft real-time constraints and interaction through multiple devices in the VE using an adaptive soft real time scheme.

## 2. HARDWARE CONFIGURATION

The training system is currently running on an SGI Onyx workstation, which controls the interactive 3-D graphics and coordinates the simulation. It is equipped with 2-way MIPS 195 MHz R10000 processors, each with a 4M L2 cache; the memory sub-system is 4-way interleaved and equipped with 256M of shared RAM. This level of performance allows us to render at fast frame rates. The graphics sub-system is driven by an InfiniteReality II with 16M of texture memory.



Fig. 1 the Motion platform - Flostation

The key element of our simulator is the 6DOF Flostation with a joystick attached on the right/left armrest as illustrated in Fig. 1. It is a Flostation chair mounted on a 6-cylinder motion base platform. The trainee is supported in a modified neutral posture similar to the posture experienced by astronauts in zero gravity. The Flostation base is driven by a 2000-PSI hydraulic system and is capable of generating roll, pitch, yaw,

heave, surge and sway with excellent dynamic response. The input and output signal from the Flostation is processed through a Window PC which also communicates with the Onyx via sockets.

We use a joystick as a navigation interface because it is relatively simple and easily recognized so that the trainee can focus on the training process rather than on mastering complex control commands. The joystick controls the position and orientation of the Flostation through interaction, which can map the control signal from the joystick to the corresponding rover movement, which is either a rotation or a forward or backward movement combined with acceleration or deceleration. The rover movement also reflects terrain following. The corresponding stereo virtual scenario is rendered almost simultaneously.

A Virtual Research VR4 HMD was chosen as video output, which offers mid-resolution active LCD display per eye with separate left and right eye feeds with a 60-degree diagonal field of view and a 48-degree horizontal field of view. A head tracker tracks the user's head movements and extends the field of view to all degrees. So far, only the horizontal rotation of the head was determined. Fig. 2 illustrates the overall system diagram.

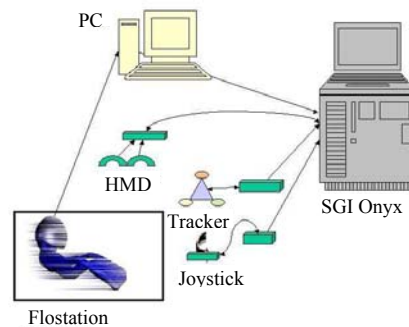


Fig. 2 Overall system diagram

## 3. APPLICATION DESIGN

### (1) Application Architecture

This system includes a set of tightly coupled software modules. The current design, which is illustrated in Fig. 3, consists of two major subsystems: the rendering engine and the Multi-threaded Terminal Emulator

(MTTTY).

The rendering engine running on the Onyx is responsible for rendering interactive 3D scenes, handling user interaction tasks, audio input and output, collision detection during maneuvers and sending events to MTTTY. The rendering engine is written in C/C++, based upon the IRIS Performer library, which supports multiprocessing and provides a high-level graphics API while still allowing direct access to OpenGL and lower level rendering details.

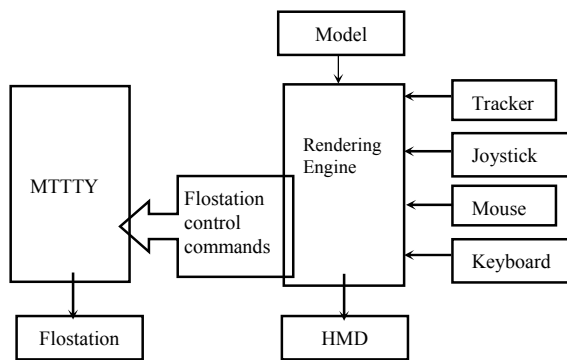


Fig. 3 Application Architecture

MTTTY is the main function running on the PC serving as the decoding engine and the safety guard. It communicates with Onyx, checks the data range, checks the motion platform orientation, converts digital data to analog signals (D/A), and sends signals to the Flostation. When the control PC receives a command, MTTTY will decode, check secure factors (e.g., check the current Flostation position and calibrate the data) and perform possible error recovery. It will either drop the command or forward it to a D/A converter. After the D/A conversion, the analog signal is forwarded to a dual mode controller. Eventually, the dual mode interface card sends an analog command to the Flostation.

Sometimes unacceptable conditions may occur. For example, each actuator extension of the Flostation is only 11.5 inches. There are an actuator scheduling algorithm and a smooth algorithm running to control the step-in Flostation and map the virtual rover movement to the Flostation actuator extension. The actuator extension may exceed the permitted position limits. Under these circumstances, the system will halt and reset the Flostation automatically.

The initial time for the EIA RS-232C port in the control PC is around 250 ms; the initial time for the

motion platform is around 200ms and in the worst case, the response time is around 800 ms. The audio signal has a 44.1KHz sampling rate stereo mode. The vibrator hi-pass cutoff point is 500 Hz; the roll off point is 16 Hz, 32-ohm impedance and 128W of power. The time compensation mechanism is used to take into account communication latency. We therefore synchronize them to obtain interactive frame rates with process pipelining and parallel techniques.

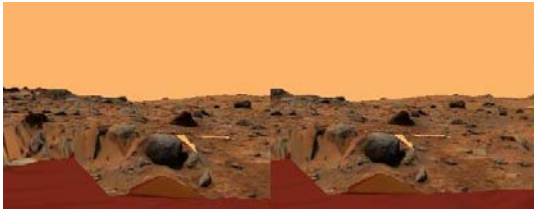
MTTTY contains three threads: a main thread (which initializes the COM port and maintains the user interface), and the other two working threads (which do all serial communication work.) For example, a reader/status thread is created to handle port reading and to monitor status events. This thread issues an overlapped read operation and an overlapped status operation. When one of these operations completes, the information is sent to the user. If a timeout occurs while waiting for an operation to complete, background work for cylinder scheduling is performed to compensate for the miss. Another thread, the writer thread, is created to accept output requests and send them to the COM port by utilizing a work queue. Each output request is placed on a linked list in a private heap and a synchronization event is signaled to indicate newly scheduled work. When the writer thread is available, it will retrieve the output request and perform the output operation. The influence from cylinder scheduling delay and electronic data transfer delay is being measured in this system. However, because it is far less than the mechanical delay, the electronic delay time is ignored.

MTTTY also defines all motion behaviors such as acceleration, damping factor, gravity, and friction. These behaviors interact with the actuator controller. The current version of MTTTY was built and tested under Microsoft Windows95 and Microsoft Windows NT 4.0 (Service Pack 4, Option Pack 4,) using Microsoft Visual Studio C/C++ Version 6.0.

## (2) The Virtual Realistic Model

Data captured by NASA/JSC and Ames space-center during the Mars Pathfinder mission was used in this project. To be convincing, the model had to be rendered in real size although this is always unrealistic and unnecessary. The original Mars model contains more than 200,000 textured polygons, which causes graphics-models loading time over 15 minutes for each subject test. To reduce the graphic system-rendering load, we reduced the number of polygons to around 5000. The

textures from the images taken were mapped to provide an appropriate “look and feel” of being on the real Mars. The virtual scenario for the left and right screens of HMD is shown in Figure 4. Known weather conditions, for example dusts effect, were also considered in the VE creation. Additionally, a rover with the size of 3mx3mx3m was also drawn.



**Fig. 4** The left and the right eyes' view displayed on HMD

#### **4. CONCLUSION AND FUTURE WORKS**

This paper presents a VE system for the NASA/JSC HMM individual training project that permits the trainee to navigate freely through the virtual Mars scenario with a high-fidelity representation of the known data and integration of Flostation, HMD, and tracking devices. In the IEEE Virtual Reality'99 conference, we tested more than 200 subjects by using a 5K polygons model and the average rendering rate was 29 frames per second.

Although we successfully built the multi-device VE, a number of areas for future research work are apparent. First, performing human subject tests for effectiveness quantification should be done in the near future. Second, user interaction techniques for manipulating objects in the Virtual Environment need further study, such as what kind of spatial behavior in the VE is more effective and naturalistic that result in positive transfer to the real world; and how to perceive, understand, and manipulate the spatial information. Last, more natural and intuitive input/output devices can be integrated to provide an enhanced VE training system.

#### **5. ACKNOWLEDGMENTS**

This material is based upon work supported in part by the National Science Foundation under Grant No. NEC95-55682, NASA grant NAG9-985, and funding from the Institute of Somatic Sciences. Any opinions, findings, and conclusions or recommendations expressed

in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

#### **REFERENCES**

- [1] J. Calvin, A. Dickens, B. Gaines, P. Metzger, D. Miller, and D. Owen, “The SIMNET Virtual World architecture,” *Proceedings of the IEEE Virtual Reality Annual International Symposium*, 1993, 450-455.
- [2] M. R. Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham and S. Zeswitz, “NPSNET: A Network Software Architecture of Large-Scale Virtual Environments,” *Presence: Teleoperators and Virtual Environments*, 3(4), 1994, 265-287.
- [3] J. S. Dahmann, J. O. Calvin and R. M. Weatherly, “A reusable architecture for simulations,” *Communications of the ACM(CACM)*, 42(9), 1999, 79-84.
- [4] R. B. Loftin and P. J. Kenney, “Training the Hubble Space Telescope Flight Team,” *IEEE Computer Graphics and Applications*, 15(5), 1995, 31-37.
- [5] L. J. Rosenblum, J. Durbin, L. Sibert, D. Tate, J. Templeman, U. Obeysekare, J. Agrawaal, D. Fasulo, T. Myers, G. Newton, A. Shalev, and T. King, “Shipboard VR: From Damage Control to Design,” *IEEE Computer Graphics and Applications*, 15(6), 1996, 10-13.
- [6] B. G. Witmer, J. H. Bailey, and B. W. Kerr, “Virtual Spaces and Real World Places: Transfer of Route Knowledge,” *International Journal of Human-Computer Studies*, 45, 1996, 413-428.
- [7] S. Tadokoro, H. Kitano, T. Takahashi, and I. Noda, “The RoboCup-Rescue Project: A Robotic Approach to the Disaster Mitigation Problem,” *Proc. IEEE International Conference on Robotics & Automation (ICRA 2000)*, 2000, 4089-4094.
- [8] D. Kaufman and W. Bell, “Teaching and Assessing Clinical Skills Using Virtual Reality,” *Studies in Health Technology Informatics: Medicine Meets Virtual Reality*, 39, 1997, 467-472.