

Design for Interactive Performance in A Virtual Laboratory

Chu P. Wang, Lawrence Koved, and Semyon Dukach

Veridical User Environments
Computer Science Department
IBM T. J. Watson Research Center
P. O. Box 704
Yorktown Heights, N. Y. 10598
(914)789-7530

Abstract

In recent years, a number of research groups have implemented various versions of virtual world concept [2, 4, 6, 7]. A common thread among these virtual worlds is a direct manipulation user interface paradigm based on a glove device with the position and orientation of the hand registered by a tracking device. To explore this paradigm, a new project at IBM Research was started in 1989 to build a virtual laboratory for scientists and engineers. Our first step is to integrate the glove and space tracking devices with the real time graphics on a graphics superworkstation. A simple bouncing ball virtual world has been created to test underlying software and fine tune interactive performance.

Our initial emphasis is placed on understanding the limitations of various system components and getting the best interactive performance from the system. With current state of technology, the glove and tracking devices can generate much more data than the graphics update process can utilize. Both the rendering process and the processes handling the device serial ports are CPU intensive. Our first design problem is how to distribute the processing and match the incoming data rates of input devices with the update rate of the graphics. After a new position from the tracker is received by the graphics, it is displayed only at the next frame update time giving the appearance that the hand image always lags behind the motion of the real hand. Our second design problem is to use techniques to compensate for this inherent lag time. This abstract describes the specific approaches we use to solve these problems and some useful insight gained in

experimenting with lag time reduction by position prediction.

System Overview

Our virtual laboratory is built on commercially available components. We choose a two processor version Ardent/Titan graphics supercomputer as our graphics platform. [3]. With the Dore' rendering package [1], each processor is capable of rendering a maximum of 20,000 smoothly shaded small polygons/seconds. A DataGlove™(VPL Research) device [8] senses finger gestures and a 3space™ ISOTRACK™(Polhemus Navigation Sciences) tracker [5] is attached to the back of the glove to track position. The glove and track devices are attached to two IBM PC-RT workstations respectively via RS-232c serial ports. The track server and glove server software for the RTs are developed at IBM Research based on vendor supplied device specifications. Both servers and the graphics platform are all connected together via an ethernet.

Integration of Input Devices and Graphics

To streamline message and data communication, we utilized a TCP/IP UDP protocol with data sent only upon request to minimize congestion. Data is sent from the server only on request to minimize congestion. The control loop for the graphics update is designed to send a new request only when the previous frame update is completed. In the mean time, although many more data records are generated by the input devices, only the latest available positions are transmitted at the time of request arrival at the servers. This way, the information flow is dictated by the slower update rate of the graphics and gives rise to stable and smooth motion.

To reduce this inherent lag time associated with graphics frame update described earlier, we have implemented a hand position prediction algorithm. Upon receiving the request at the tracker, instead of forwarding the latest position, we send a predicted position to the graphics update routine. Thus, the image in the next frame will be generated at the predicted position which should be much

closer to the actual position of the hand. The ratio of tracker generated positions to the sampled positions used for prediction is an important design parameter. The larger the ratio is, the more accurate the prediction would be. We want to stress the importance of using a powerful dedicated tracker server to provide the maximum possible input stream of positions. For tracker servers with modest processing power, an alternate method to increase the tracker input data rate is to off-load the RS-232c port handling to a dedicated co-processor.

Results

For the bouncing ball demo, we constructed a simple hand model with cylinders (finger sections), spheres (finger tips) and rectangular block (palm). We pre-assembled five bending positions for each finger. A gesture is formed by choosing one bending position from each finger. We also implemented a simple but fast gesture recognizer that maps the glove input into one of the five pre-defined gestures. In this way, we achieve good rendering speed without losing much functional flexibility and visual realism. In the bouncing ball world, the hand is able to grab, release, catch and throw the ball. The hand imparts an initial velocity to the ball at the point of release. The ball bounces either on the side walls or falls down under gravity and bounces on the ground with some loss of energy. The latter is to ensure that the ball will come to rest after a number of bounces.

The bouncing ball virtual world demo has been operational for a few months. We have been using this demo as a vehicle to fine tune both the hardware and software components and to make extensive measurements. Both software timers and video camera taping are used to correlate times between the real and virtual world events.

The tracker server is currently operating at 20.5 maximum input positions per second with the device serial link speed set at 9600 baud. The average measured update rate for the virtual world update is 9.5 frames per second. The measured lag time is between 160-170 milliseconds without position prediction. The system timer has a resolution of 10 milliseconds. The measured round trip message delay on the ethernet from a server to the graphics workstation is about 12 milliseconds. The prediction scheme works quite well when the hand motion is smooth and even. During rapid acceleration or deceleration, the predicted

position tends to overshoot the actual position and converges in two to three update cycles. This is attributable to the relatively low input stream rate (20 positions/second) of the tracker device.

We are also planning to improve the performance of the position prediction scheme further by doubling the device input data rate and by increasing the timer resolution to 1 millisecond or better. Results will be reported in a future publication.

References

1. Bruce S. Borden, "Graphics Processing on a Graphics Supercomputer", *IEEE Computer Graphics and Applications*, 9(4), pages 56-62, IEEE, July 1989.
2. Frederick P. Brooks, Jr., "Grasping Reality Through Illusion - Interactive Graphics Serving Science", *CHI'88 Proceedings*, pages 1-11, ACM, May 1988.
3. Tom Diede, Carl F. Hagenmaier, Glen S. Miranker, Jonathan J. Rubinstein, and William S. Worley, Jr., "The Titan Graphics Supercomputer Architecture", *Computer*, 21(9), pages 13-30, IEEE, September 1989.
4. Scott Fisher, "The AMES Virtual Environment Workstation (VIEW)", *SIGGRAPH'89 Course 29 Notes*, pages 10-27, ACM, August 1989.
5. Frederick H. Rabb, Ernest Blood, Terry O. Steiner, and Herbert R. Jones, "Magnetic Position and Orientation Tracking System", *IEEE Transaction on Aerospace and Electronic Systems*, AES-15(5), pages 709-718, IEEE, September 1979.
6. David J. Sturman, David Zeltzer, and Steve Pieper, "Hands-on Interaction with Virtual Environments", *SIGGRAPH'89 Course 29 Notes*, pages 5.1-5.14, ACM, August 1989.
7. David Weimer and S. K. Ganapathy, "A Synthetic Visual Environment with Hand Gesturing and Voice input", *CHI'89 Conference Proceedings*, pages 235-140, ACM, April 1989.
8. Thomas G. Zimmerman, Jaron Lanier, Chuck Planchard, Steve Bryson, and Young Harvill, "A Hand Gesture Interface Device", *CHI + GI 1987 Conference Proceedings*, pages 198-192, ACM, April 1987.