Using Cellular Phones to Interact with Virtual Environments

Bent Dalgaard Larsen Jakob Andreas Bærentzen Niels Jørgen Christensen
Technical University of Denmark*

Distributed VR promises to change the way we work and collaborate [Singhal and Zyda 1999]. In this sketch we will extend the accessibility of the virtual world originally developed in [Larsen and Eriksen 1998] by introducing the use of the modern cellular phone as a platform for primitive interfaces to VR applications. We believe that our use of a cellular phone has led to the first completely pocketable platform for VR user interfaces.

While a cellular phone will offer only primitive access to the virtual environment, we expect that being able to access the VE at all times will spawn new and enhance existing applications of VR. For instance, existing online multi-user games may provide limited access through a cellular phone enabling people who either in transit or just not near a PC to participate through a more limited interface. An important question is "how limited?". We believe that a "cellular interaction mode" need not be very restrictive as long as careful use is made of the technology available.

Unfortunately, existing cellular phones have no hardware support for 3D graphics and the screen is much smaller (Typically 176×208 or 208×320 pixels). Furthermore, the only input is from a limited number of push buttons. But a cellular phone has a high-speed TCP/IP connection to any server on the Internet using the current GPRS (General Packet Radio Service) protocol (40.2 kb/s) or the upcoming and much faster UMTS (Universal Mobile Telecommunications System) protocol.

Currently, the producers of cellular phones have chosen no common operating system standards. However, many of these mobile devices support a subset of Java (J2ME—Java 2 Micro Edition). This contains no floating-point math and the only supported network protocol is HTTP which implies that the only way to obtain duplex data streams on the cellular phone is if it continuously polls the server.

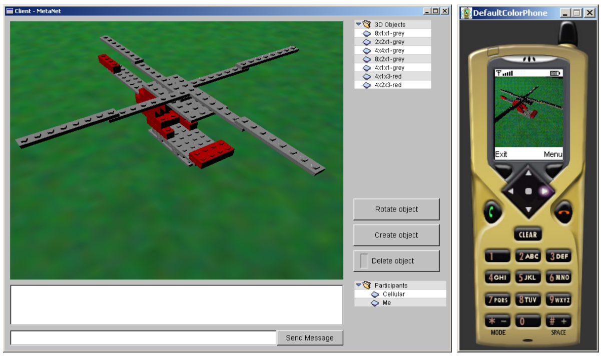

Figure 1: *Interface on the workstation and the cellular device.*

We have chosen to construct a LEGO$^{\text{TM}}$ building application and our task was to enable people to build LEGO$^{\text{TM}}$ structures in collaboration using either a traditional workstation or a cellular phone.

The system is based on a client-server model. The server contains the 3D models and maintains connections to clients. In our case, the clients may be either workstations or cellular devices. In the following discussion, we focus on the case where the client is a cellular device.

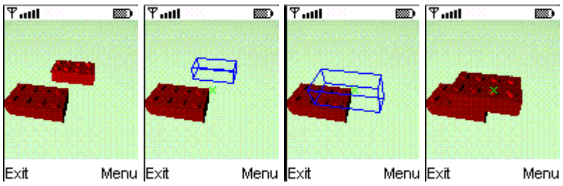

Figure 2: Local modifications on the cellular phone.

The lack of support for a 3D graphics API and floating point arithmetic makes it very important to divide work between the server and the cellular device in an intelligent way. In our system, we have chosen to perform high quality rendering on the server which uploads images to the device on demand. Since LEGO$^{\text{TM}}$ bricks manipulated interactively cannot be rendered locally in real-time, they are rendered in wire-frame. In the absence of floating point numbers, we have implemented a simple facility for fixed point arithmetic.

As the user moves in the environment new images on the server are generated and transferred to the cellular phone. When the user selects a LEGO$^{\text{TM}}$ brick by pointing at it, the server sends a subimage (the changed part of the image) to the user without the selected brick, which is then drawn in real-time wire-frame while the user alters its location and orientation. As the user deselects the brick, another subimage is generated on the server and the high quality image on the mobile phone is then update again.

The user interface is implemented as follows:

- A locator is shown on the display of the phone, which is controlled by the arrow keys.
- When the selection key is pressed, the brick under the locator is selected using a ray-box intersection algorithm.
- The brick is moved using the arrow keys. The arrow keys are used to move the brick in the ground-plane. The height value is found by moving the brick the smallest possible distance from its previous height value, given the constraint that the brick does not intersect other bricks.

Users located in the same VE and connected using a workstation have the same posibilities of manipulating LEGO$^{\text{TM}}$ bricks. Modifications made either through the cellular phone or the workstation are instantly reflected on all devices connected to the virtual environment.

Although this is still work in progress, our conclusion is that mixing locally rendered images on the cellular phone with server rendered images is an appropriate solution for applications such as ours.

This work was supported in part by the STVF project DMM and the Nordunit2 project NETGL.

References

LARSEN, B. D., AND ERIKSEN, M. A. 1998. *Multi-User Interactive Virtual Environments*. Master's thesis, Technical University of Denmark.

SINGHAL, S., AND ZYDA, M. 1999. *Networked Virtual Environments*. Addison Wesley.

*e-mail: (bdl|jab|njc)@imm.dtu.dk