

Application sharing in 3D graphical environments

Robin Stridh, Stefan Seipel

Department of Information Technology, Uppsala University

1 Introduction

In many 3D-applications, there is a desire to reuse existing 2D applications. Both since most currently available application are in 2D, and would be expensive and difficult to convert into 3D applications, but also since some applications that will be used in 3D environments are naturally more suited for 2D, like text editing and web browsing for example.

There is also a desire to share applications between users, so many users can see the same application on different computers. Especially in educational and demonstration scenarios, a common environment has to be shared between users.

We propose a solution to this by implementing a shared, virtual 2D terminal that can be used by several users on different physical computers, but mapped into a virtual 3D environment.

2 Virtual Network Computing - VNC

Virtual Network Computing is a system for sharing applications over the network, developed by people at the AT&T Laboratories Cambridge in 1999. [1] On the computer on which the applications are run, VNC acts as a transparent server which handles all the graphics processing, but instead of displaying the frame buffer on a local monitor, the pixel data is compressed and sent to one or several remote clients that decompresses and displays the data. Input from the client(s) are sent back to the server and the server treats this input as coming from the local mouse or keyboard.

VNC is a widespread, open source system with both server and client software for several platforms including Unix, Linux, Microsoft Windows and MacOS.

Since all the client program has to do is decompress the remote frame buffer data and display in on the local monitor, the client program is quite simple compared to the server.

3 VNC in 3D

The approach taken here is to use the existing server program, `vncserver`, as is, and write new software on the client side, capable of receiving data from the server and use in a 3D environment.

Since the communication protocol is quite simple, and well documented [2], a new library of functions for receiving the pixel data was developed, where the data is handled and represented in a way more suitable for using in interactive 3D applications.

The pixel data from the VNC server can, after being extracted to raw pixel data by these library functions, be used in an arbitrary way by the 3D applications. The most obvious way of using the pixel data from the server is as texel data in a texture, mapped onto a flat polygon in the 3D space.

To improve the usability of these functions, a more specific wrapper structure for the Virtual Reality Toolkit (VRT)[3] has also been implemented. This VRT structure consists of a single quadrilateral polygon, with the pixel data from the VNC server texture mapped onto it, together with functions for automatic update of the frame buffer and functions for mouse and keyboard input.

4 Test Setup

Testing the performance of the 3D VNC client has been made by measuring the update frame rate at the client, and the time between sending of an update request and reception of an update message, for a number of different scenarios. The scenarios have been chosen to test different aspects occurring in normal usage situations at the client side, but are also tested on a few computers with different performance and different network configurations in relation to the VNC server.

5 Test Results

The test results for one test configuration are illustrated in figure 1 and figure 2. Here 1000 samples

were recorded for each test scenario.

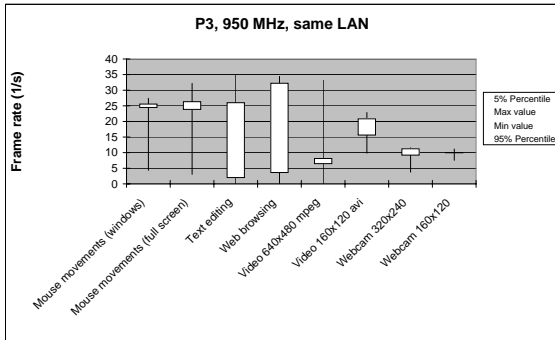


Figure 1: VNC frame rates for different test scenarios

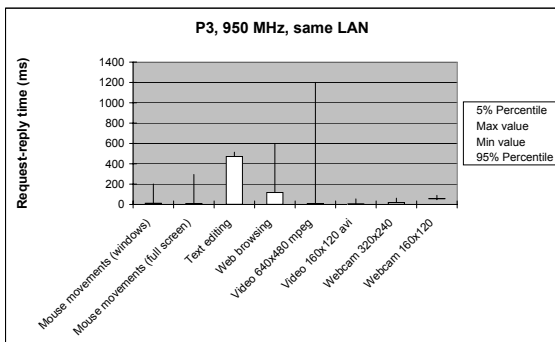


Figure 2: VNC request-reply times for different test scenarios

6 Conclusions

Frame rates for tasks with a high level of interactivity and small amounts of data transmitted, like mouse movements and text editing, turned out to be quite high, about 25 fps. The limiting factor here is the request-reply time, caused by network delays, which will make the interaction slower.

For tasks with little, or no, interaction like video streaming, the frame rates dropped to between 7 and 20 fps, which is still rather high, considering that the image resolution was between 160x120 and 640x480 pixels and the color depth was 32 bits per pixel. Here the request-reply time had little or no effect on the result.

Naturally the frame rates will vary a lot, depending on what type of interaction is performed

References

- [1] Kenneth R. Wood Tristan Richardson, Quentin Stafford-Fraser and Andy Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, January 1998.
- [2] Kenneth R. Wood Tristan Richardson. *The RFB Protocol*, 1998.
- [3] Stefan Seipel. *Virtual Reality Toolkit 1.5 - Programmer's Manual*, 2002.

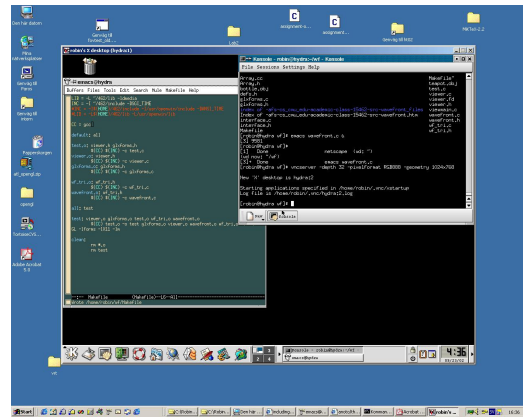


Figure 3: A screenshot of the classic VNC client, vncviewer

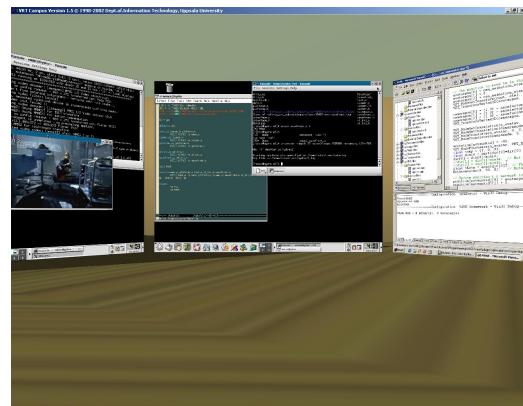


Figure 4: VNC in a 3D environment