Technical bulletin: Remote visualisation with VirtualGL

Dell/Cambridge HPC Solution Centre

Dr Stuart Rankin, Dr Paul Calleja, Dr James Coomer

Introduction

This technical bulletin is a reduced form of the whitepaper entitled “Remote visualisation using open source software & commodity hardware”. It is aimed at High Performance Computing (HPC) users and administrators who wish to develop remote visualisation solutions using open-source software and commodity hardware. The whitepaper provides background reading as well as a command-line configuration guide for those wishing to recreate the solution.

It is commonplace today that HPC users produce large scale multi-gigabyte data sets on a daily basis and that these data sets require interactive post processing with some form of real time 3D or 2D visualisation. The traditional HPC workflow process requires that these data sets be transferred back to the user’s workstation, remote from the HPC data centre over the network. This process has several disadvantages, firstly it requires large I/O transfers out of the HPC data centre which is time consuming, and also it requires that the user has significant local disk storage and a workstation setup with the appropriate visualisation software and hardware.

The remote visualisation procedures described here remove the need to transfer data out of the HPC data centre. The procedure allows the user to logon interactively to the Dell | NVIDIA remote visualisation server within the HPC data centre and access their data sets directly from the HPC file system and then run the visualisation software on the remote visualisation server in the machine room sending the visual output over the network to the users remote PC. The visualisation server consists of a T5500 Dell Precision Workstation equipped with a NVIDIA Quadro FX 5800 configured with an open source software stack facilitating sending of the visual output to the remote user.

By deploying this methodology users can gain direct visual access to their HPC data wherever and whenever they require it. Also it reduces the need to transfer large scale HPC data over the network and removes the need for users to configure complex and expensive visualisation software and hardware on their local machine. In this technical bulletin we summarise work performed with an OS-neutral extension of familiar remote desktop techniques using open source software for remote visualisation.

Whilst traditional X11/GLX over SSH methods may require very high quality end-to-end network connection and additional hardware resources at the local workstation or PC, remote-desktop software limits the required network bandwidth by offloading pixel rendering to the server but is not designed to support 3D applications with hardware acceleration. VirtualGL (VGL) is open source software which addresses these problems by separating off the 3D rendering task and invoking the hardware-accelerated graphics system on the remote server to produce the required images. The resulting (2D) images are then fed to the target (local) X display with the rest of the 2D drawing instructions. The key advantages of this are (i) any 3D image which can be rendered by the server's graphics hardware and software can in principle be rendered by VGL, and (ii) since only 2D commands are ultimately sent to the X display, the destination display need have no native 3D capability at all.

In the following sections we summarise various implementations of VirtualGL in combination with Dell and NVIDIA hardware and other widely available software to demonstrate the effectiveness of commodity systems in supporting remote visualisation solutions.

Test system and software environment

The test system includes a graphics server, client machines and a range of software.

The application server comprised a T5500 Dell Precision Workstation (2x quad core Intel Xeon 5560@2.80GHz, 24GB RAM) equipped with a nVidia Quadro FX 5800 (GT200GL) high-end graphics card and running 64-bit Scientific Linux 5.4. The native X display driven by the FX 5800 has full hardware-accelerated OpenGL/GLX 3D graphics through proprietary nVidia driver software.

Three different types of client systems were used with different network connections to the server, the intention being to emulate three interesting categories of remote use:

- **Departmental.** Intel Core2 6600 @ 2.40GHz, 2GB RAM, Fedora 12 64-bit Linux workstation , nVidia GeForce 9400 GT graphics card. Connected to the server via a 100Mbit/s departmental ethernet
- **Home.** Intel Core2 E6850 @ 3.00GHz, 2GB RAM, Fedora 13 64-bit Linux workstation, nVidia GeForce 8600 GT graphics card. Connected to the server via a 10Mbit/s home broadband cable modem service
- **Laptop.** Intel Core2 Duo T9300@2.50GHz, 2GB RAM, Windows Vista Ultimate SP2 + Cygwin 1.7.5-1 Dell XPS M1530 laptop. Connected to the server via local Eduroam 50 Mbps wireless.

We chose to test VirtualGL with two readily available applications which would both produce complex and rapidly evolving images and at the same time require good interactivity to be usable without pain:

- Celestia (http://www.shatters.net/celestia/) is a free, interactive 3D space simulator. Version 1.5.1 with the default settings was used.
- SmokeParticles is one of the demo programs included in the CUDA GPU Computing SDK code samples from nVidia (http://www.nvidia.com/object/cuda_home_new.html)

Aside from VirtualGL, we also took advantage of other software to optimise transport of the images across the network:

libjpeg-turbo is an improved (SIMD-accelerated) JPEG codec (see http://libjpeg-turbo.virtualgl.org/) for faster compression of the rapidly changing, complex images typically produced by 3D graphics applications. It is open source and freely downloadable.

TurboVNC is a TightVNC variant which is the current preferred accelerated VNC for use with VirtualGL.

The combination of VirtualGL, and TurboVNC gives rise to the data flows as shown in Figure 1.

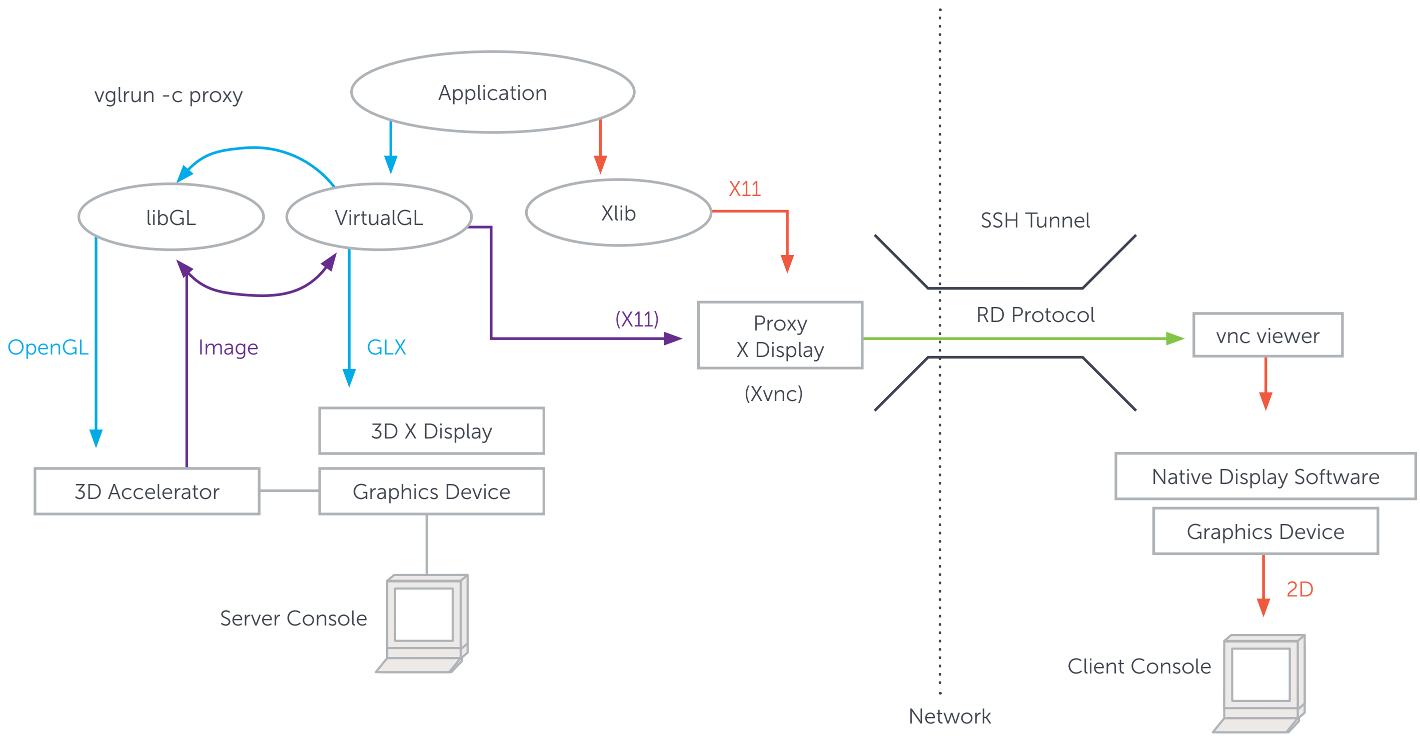

Figure 1. 3D application used remotely via VirtualGL and TurboVNC. The 3D drawing commands are forked off by interposing replacement shared objects in front of libGL.so, and rendered with hardware acceleration by the server graphics card. The rendered images are read back and pushed using X11 commands to the proxy X display. The entire desktop is compressed and sent using a combination of TightVNC and accelerated JPEG to the remote viewer, using an SSH tunnel for security.

Importantly, the quality of VNC and (lossy) JPEG compression applied to the virtual desktop can be altered on the fly – more aggressive levels of compression reduce the network bandwidth required and improve frame rate and interactivity, at the cost of decreased image quality and increased server and client CPU load. Thus visual quality can be sacrificed temporarily for improved responsiveness, and then restored when interactivity is not needed.

Results

Application frames per second (fps) is the rate at which frames representing the entire desktop are delivered to the client, measured using tcbench. In addition to framerates (averages of three tcbench runs), we also give qualitative summaries of the usability for the various JPEG compression settings that can be adjusted on the fly within the TurboVNC. It wasn't possible to use tcbench on the Windows system.

The default geometry of the virtual desktop is 1240x900, with colour depth 24. In the absence of compression, this suggests that each additional frame per second requires 3MB/s additional bandwidth, i.e. a 15 fps minimally acceptable frame rate would already require a gigabit network pipe (without compression).

Test System	SmokeParticles					
	Perceptually lossless JPEG		**Medium quality JPEG**		**Low quality JPEG**	
	Frame rate (fps)	Qualitative	Frame rate (fps)	Qualitative	Frame rate (fps)	Qualitative
Departmental: Linux 100Mbps	28.5	Good image, good interactivity	36	Good (not noticeably worse than previous case) image, good interactivity	38	Noticeably blurry image and good interactivity
Home: Linux 10Mbit/s cable	7.5	Good image, but jerky and poor interactivity	16.5	Good image, good interactivity	25	Blurry image but good interactivity
Wireless Windows Laptop	-	Good image, but jerky and reasonable interactivity	-	Good image, slightly jerky but generally good interactivity	-	Blurry image, sometimes jerky but good interactivity
	Celestia					
	Perceptually lossless JPEG		**Medium quality JPEG**		**Low quality JPEG**	
	Frame rate (fps)	Qualitative	Frame rate (fps)	Qualitative	Frame rate (fps)	Qualitative
Departmental: Linux 100Mbps	21	Good image, but jerky and poor interactivity	28	Good image, good interactivity	27.8	Noticeably blurry image and blurred text but good interactivity
Home: Linux 10Mbit/s cable	7.5	Good image, but jerky and poor interactivity	15	Good image, good interactivity	24	Blurry image and poor text but good interactivity
Wireless Windows Laptop	-	Good image, but very jerky and poor interactivity	-	Good image, jerky motion but acceptable interactivity	-	Blurry image, poor text and irregular motion, but acceptable interactivity

Conclusions

The large data sets resultant from HPC applications frequently require further processing by the owner, who is typically remote from the HPC local network. In many scientific disciplines such post-processing involves 3D graphical techniques, often requiring the services of a high-end graphics card for rendering complex 3D images in real time. The conventional approaches to this are either (1) perform all processing on the owner's local resources, requiring a heavy-duty, non-mobile workstation with adequate storage, processing power, memory and graphics hardware, plus local system support effort; or (2) perform the post-processing on the HPC server with remote graphics rendering taking place on the client workstation through SSH, X11 and GLX, which may work very slowly, or not at all. Similarly remote desktop techniques typically fail to support 3D applications, or if they do, do so only with non-accelerated software rendering, so also at best work slowly (probably unusably so).

The Dell | NVIDIA remote visualisation solution deployed here using the open source software, VirtualGL (VGL) allows a 3D application executing and rendering on a server with hardware-accelerated 3D capabilities to produce a two-dimensional rendered image amenable to the usual SSH/X11 and Linux remote desktop methods. We have demonstrated in this technical bulletin, with full details in an accompanying whitepaper that when used together with a remote desktop package such as TurboVNC, VGL allows low-end clients, using multiple operating systems, to view and operate effectively 3D applications executing on a remote HPC server node equipped with a high-end graphics card.
The improved codec provided by TurboVNC allows the network requirements between server and client to be mitigated dynamically so that a 10Mbit/sec bandwidth connection may be sufficient, offsetting image quality when appropriate against improved interactivity when this is required, and conversely.
We have illustrated this using three systems (two Linux, and one Windows) with network connections simulating departmental, home broadband, and office wireless connectivity.

This allows HPC users "anytime – anywhere" visual access to large scale data sets thereby removing data locality issues and the requirement for high-end user workstations configured with complex visualisation software. Via the Dell |NVIDIA remote visualisation server users can now perform complex 3D visualisation of HPC data on the fly in the airport lounge or coffee shop on their laptop!