

Important issues concerning interactive user interfaces in grid based computational steering systems

Roy S. Kalawsky, Simon P. Nee
East Midlands e-Science Centre, Loughborough University

Abstract

This paper discusses a number of important issues concerning interactive user interfaces in grid based computational steering systems. The goal is to improve the scientist's efficiency and make the system more accessible to users who are not necessarily computer experts. A key step is the identification of critical factors that affect the performance of the user. Whilst it is easy to trivialize the design of these systems, in practice the design of an efficient solution can be very challenging. The more advanced computational steering systems may be regarded as a supervisory control system with high level of interaction and consequently can only be effective if the user is considered during their design. A deeper analysis will show there are many trade-offs that need to be considered in order to achieve optimization from a user's perspective.

1. Introduction

The grid (Foster and Kesselman 1999) has emerged as an extremely important development for future computational based scientific research. Consequently, a new approach to research known as e-Science (Hey and Trefethen 2003) is taking shape with the goal of undertaking large scale scientific research over a distributed computing environment. The grid is ideally suited to support computer simulations in a diverse range of scientific and engineering fields such as condensed matter physics, molecular dynamics and computational fluid dynamics etc. The key advantage is that it facilitates the execution of large scale simulations by harnessing the power of a distributed computing resource. For extremely large scale simulations it is not uncommon to employ many hundreds of processing nodes (perhaps distributed geographically) over which to distribute the computation in order to improve throughput or increase the scale of simulation. A fully featured grid will enable the scientist's application to be farmed out to remote sites with the necessary computing and visualisation resources. Therefore, the scientist has the potential to harness enormous computational/visualisation resources that otherwise would have been prohibitive or inaccessible. However, this distributed capability comes with its own set of challenges. The paper will look at the nature of the interactive user interfaces in the context of a distributed visualisation system as well as some of the related user issues.

2. The Nature of Interactive Visualisation

The complexity and sheer quantity of data generated by today's scientific visualisation tools can make it virtually impossible to process

information completely numerically. The well known adage "The purpose of computing is insight, not numbers" coined by (Hamming 1962) supports this. Even though the human visual sensory channel is the widest information pipeline to the brain and our minds are particularly effective at abstraction, approximation and simplification we can still find it difficult to analyse very large static images. To assist interpretation, scientific visualisation data can be graphically rendered in terms of re-colouring, segmentation, sliced, or iso-surfaced to name but a few. Unfortunately, even these techniques can increase the complexity of the scientific visualisation task if an inappropriate technique is used. The scientific exploration processes can also be assisted by providing new insight through interaction with the visual representation. Over the past decade human-computer interfaces have evolved considerably to the point where user interaction is predominantly a visuo-manual activity as opposed to a text input approach. Quite often with very complicated 3D renderings the information of interest is hidden until the scientist interacts or moves the image. The value of interaction with a scientific visualisation should not be under-estimated. In many grid applications there is a focus on the structure of scientific visual representations. Whilst there is generally less concern with the visual properties of the visual image itself (i.e. brightness, contrast, choice of colours, etc.) greater interest is targeted towards the spatial interpretation given to a vast range of visualisation techniques. For example, iso-surfacing is a fairly standard technique used in scientific visualisation which creates surfaces (3D) of constant scalar values. These contours provide discrimination between different threshold levels in the dataset and are intended

to aid the scientist's perception. Even though this is a widely used technique the contour patterns of an image are well known to provide fairly incomplete evidence for the rich spatial environment that we perceive (Gregory 1993). Also there are many other techniques that require a fair degree of understanding from the scientist in order to extract the features and structures of interest. It should be noted that scale and structure cannot always be determined solely from the visual information alone. However, from a human perspective we are able to update our knowledge through observation of the consequences of interaction. For example, it has been noted that the addition of motion in an image is important to help resolve some of the ambiguities of 3D surface segregation which would otherwise present erroneous depth-order relations (Ittelson 1968). Scientific data, such as medical imagery with moderate to high levels of transparency cannot be viewed effectively by static stereo viewing alone. Interestingly, when stereo is used with user induced motion there is a noticeable enhancement and efficiency in the visual perception of the image (Russell and Miles 1987). Time-varying flow lines (streamlines) consist of particle traces which have been computed for a vector field. When a large number of streamlines are displayed simultaneously it is of great benefit to be able to rotate the image to resolve the perceptual ambiguities that exist in a static representation. These are a few of the many examples that highlight the importance of user interaction (and in particular user induced motion) as a means of aiding scientific visualisation. Consequently, this means that user interaction should not be under-estimated in terms of its contribution to perception of information and it should therefore be considered a key part of any visualisation system. Accordingly, 'knowing-by-seeing' is an active, constructive process utilising considerable non-visual information, some of which is based on prior knowledge. Factors which affect user interaction can also have an impact on the perceptual efficiency of image interpretation.

3. Requirements Analysis

User interaction is central to modern computer based steering systems. The ability to control and manipulate steering parameters interactively is a major contributor to the usability of the system. Early steering systems relied on command line or file based parameter steering. Whilst this approach is reasonably satisfactory for applications that take many hours to execute a single iteration it is not so useful for situations

where the scientist needs to manipulate their data in an interactive fashion. For more responsive steering systems it is necessary to be able to update visualisation viewpoints at least 10Hz. Unfortunately, it would be impractical to expect all parts of the simulation environment to be capable of updating at this rate. Indeed, in the case of extremely large scale simulations a single simulation loop may still take many hours to execute. However, this does not imply that all parts of the simulation need to be tied to this long lead time computation. By separating the visualisation from the simulation loop it is possible to increase the responsiveness of the user's interaction. The exact system configuration has to take account of the capabilities of the resources available to the scientist. It has already been mentioned that a large scale grid application could indeed be distributed geographically over a high performance network. At one extreme the whole simulation environment, including visualisation could be remoted from the user who may have extremely limited access capability.

We are primarily interested in this paper with 'Big Science' applications where large scale simulations (up to tera-scale) are employed to yield the necessary visualisation data. Such large scale visualisations can be so demanding that a grid based solution is the only viable solution if results are to be computed in a reasonable time. However, as soon as a distributed compute-visualisation architecture is considered this raises a number of significant user interaction issues. The first user interaction issue is concerned with the effects of time delay between user input and corresponding output from a given simulation. Depending on the overall system architecture the interaction time delay can be anything from fractions of a second to several minutes or in extreme cases several hours. General advice regarding what constitutes an ideal response time in an interactive system has largely been the same for the past three decades (Card and Mackinlay 1991), (Miller 1968) and can be summarized as shown in Table 1.

Various techniques can be used to reduce these time delays by compressing the data that is sent over the network thus leading to a reduction in network bandwidth. However, this is not always commensurate with a guaranteed decrease in time delay or latency. Herein lies the second problem and that is the effect that such data compression techniques have on the user's interaction. It is a reasonable assumption that

any operation on a data stream has the potential to add a delay into the overall system pipeline, or possibly worse, to introduce a degree of distortion into the original dataset. The real question is how to trade off between frame rate and compression.

Response time	Comments
0.1s	This is about the limit for the user to feel that the system is responding instantaneously to their input. Normally no feedback would be necessary since the user is able to see the results of their interaction without any noticeable delay.
1.0s	This is about the limit for the user's flow of thought to stay uninterrupted. However, the user will often report a sense of losing contact with their data. Typically, user feedback would not need to be provided between 0.1 – 1.0 second.
10s	This represents the limit for keeping the user's attention focused on the interface. For longer response times, users will be easily distracted and will often try and perform other tasks while waiting for the task to finish. Ideally, feedback should be given in the form of a progress indicator. If the response time is unpredictable then the progress indicator becomes even more important. (Myers 1985)

Table 1: Effect of response time on user interaction

4. Implications of grid architectures on interactive visualisation

Whilst the grid offers the potential for harnessing tremendous computational capacity, the distributed nature of grid resources and indeterminate network performance can introduce human computer interaction issues. Apart from basic user interaction tasks such as resource discovery, job submission, resource management etc., the scientist is also interested in controlling (or steering) their applications by manipulating one or more simulation parameters. Current human-computer interaction theory highlights the need for direct manipulation interfaces rather than 'crude'

command line or file based control methods. Ideally, steering would be done interactively whilst observing the result, presupposing that the response time of the overall simulation can meet the timing requirements – though this is unlikely for very large scale simulations. An in-depth human factors audit which has looked the scientist's workflow has been undertaken (Kalawsky and Nee 2004). This study has captured the key user interaction requirements across a number of scientific applications that clearly indicates that the associated user interaction process is complex and involved, which necessitates a number of stages or steps from 'pre' job submission preparation through to the more highly interactive stages of interactive visualisation. This paper is a discussion on a number of issues that affect user interaction within an interactive computationally steered simulation. In such systems, the response time from initially requesting a change to a given parameter, to seeing a change in the resulting visualization, is highly dependant on the cycle time of the simulation itself and any communication that needs to occur between the various compute and visualisation resources (Refer to Figure 1.). It is not unusual for large scale simulations to take hours or even days to generate a new visualisation dataset. Figure 2 shows that there are actually several interactive loops within typical computational steering systems. Firstly, there is the main simulation loop which generates the data set for visualisation. Secondly, there is the visualisation stage whereby the data is filtered to extract the data that needs to be rendered. Thirdly, there is the visualisation rendering stage where the actual visualisation is created. Figure 3 illustrates the order of magnitude of response time between the different stages. The response time of the simulation loop of most large scale scientific applications is almost certainly not going to be real-time (with current technology) since the computation could take hours. Consequently, if the user requires to manipulate some of the initial conditions then a full re-run of the simulation may be necessary. Filtering the resulting data set to create the dataset for visualisation could be achieved in a much shorter timeframe but this obviously depends on the type of pre-processing required. Once a visualisation data set has been produced the scientist often wants to explore interactively the resulting visualisation to observe areas of interest. This exploration may involve operations such as image translation, rotation or navigation through the 3D data set. At the level

of the visualization it is also possible that further pre-processing operations may be required such as 3D iso-surfacing or volumetric renderings. These tasks may not require a full re-run of the simulation in order to change the iso-surface thresholds. Consequently, the response time will be significantly less than the simulation time, even to the stage where near real-time interaction becomes possible.

Unfortunately, despite rapid advances in visualisation technology not everyone will have the necessary hardware to undertake local interactive visualisation. The nature of certain visualization tasks may mean that the necessary high performance visualization hardware is only available at specialist sites. In these situations various solutions are available allowing the user to connect to remote visualisation hardware such that a visualisation is produced at the remote site and transmitted to the user. Transmission of the remote visualisation can be achieved by compression prior to transmission and subsequently decompressed at the local host end. An alternative technique, though less satisfactory from a network bandwidth perspective, is to send OpenGL commands across the network. If non-real time interaction is acceptable then either of these methods can be made to work though with differing network bandwidth requirements. However, to achieve interaction at interactive rates then other factors need to be considered other than 'just' compression.

5. Separation of visualisation from main compute loop - local versus remote visualisation

Given the desirability to support interactive exploration it is considered good practice to separate visualisation rendering from the main compute/simulation loop since this will not tie the visualisation system response time (during an interactive session) down to the long execution time of the simulation. If the user does not have high-end visualisation systems locally it is then feasible to employ a remote visualisation or distributed system. Various techniques exist for supporting distributed visualisation. The X11 protocol has been very useful for many years and is based on simple 2D XLib calls. Unfortunately, when 3D graphics are processed, the library overhead becomes significant, due to network bandwidth and computational resources at the local and remote host ends. Extensions to OpenGL through GLX have also enabled some remote rendering functionality similar to X11, but this

is still restrictive when it comes to real-time interaction for the same reasons. Other interesting techniques have emerged such as the Virtual Network Computer (VNC). Whilst VNC can work extremely well for some applications it does not support the OpenGL network interface (GLX) and consequently relies on XLib calls. Unfortunately, it does not support interactive high frame rates and so must be ruled out. Visualisation hardware has become more sophisticated in terms of the processing that can be performed within dedicated hardware and this has led to a new technique of remotely generating a compressed image and transmitting this to a user over a network.

6. Visualisation Servicing

A number of very exciting products such as Silicon Graphics VizServer (Ohazama 1999) have emerged to meet the needs of remote visualisation. The basic concept is to produce rendered visualisations at geographically remote sites by compressing a rendered image in a remote visualisation system and transmit the compressed data over a network. The local host runs a lightweight client which decompresses the image and makes this available to the user using fairly modest resources. The concept is that visualisation is performed by a high performance remote visualisation system. Figure 4 illustrates the basic concept and shows the top level processes in the visualisation pipeline (remote – local host). Depending on the compression technique the approach is not ideally suited to users who are connected through low bandwidth networks and require high quality images.

7. Performance trade-offs in Server-Client Remote Visualisation Systems

There is no doubt that server-client visualisation systems offer many scientists access to very powerful visualisation resources. However, despite various claims that the user will obtain relatively fast frame rates for reduced network bandwidth for server-client visualisation systems (compared to simply sending uncompressed visualisation data over the network) this only comes with a performance trade-off in other areas. The user needs to consider a range of inter-related performance issues such as resolution of the image and the compression method being employed. To use server-client visualisation systems effectively it is important to recognize the inter-dependency of the key performance parameters. Parameters

such as desired frame rate, required image size, network bandwidth and compression technique are all inter-related. Unfortunately, it is not possible to achieve high frame rates, low network bandwidth, high image resolution and lossless compression at the same time. The user must trade-off between these parameters against the critical visualisation parameters of their application.

To get an idea of the sort of performance trade-off refer to Table 2 (Fowler 2000) which shows observed frame rate and network bandwidth for an image resolution of 640x480 pixels over a 100Mbps network for Silicon Graphics VizServer. The compression techniques (with the exception of the one non lossless method) used by VizServer are regarded as lossy techniques. This means a certain amount of data is lost in the compression/decompression process. The two standard compression techniques used by VizServer to achieve the higher frame rates are essentially based on the Block Truncation Coding algorithm and offer a compression ratio of 8:1 for Colour Cell Compression (CCC) or 4:1 for Interpolated Cell Compression (ICC). The user needs to trade-off network bandwidth, frame rate of rendered images and image compression technique. What the table does not show is that during certain kinds of user interaction such as roaming around the image the performance can actually be significantly worse because the whole scene needs to be transmitted over the network rather than small changes. An example of where this would occur is if the user panned a very wide field of view image from left to right. Consequently, the user needs to identify the parameters which are important to their application.

Compression technique	Frame rate (Hz)	Network bandwidth (Mbps)
Colour cell compression	12	11.1
Interpolated colour cell	10	18.4
No compression	8	59.0

Table 2: Observed VizServer frame rate versus network bandwidth

Some applications will not be able to tolerate any kind of image compression since data would be lost or interpolated. For example, in medical imaging applications the use of any form of compression is generally regarded as

unacceptable, meaning that only lossless techniques are acceptable. This creates a serious problem for distributed visualisation systems that depend on lossy compression techniques. Much work still needs to be done to develop efficient lossless image compression techniques for grid applications.

For near or real-time response the time taken for image compression-decompression can become an important factor. The user must still be aware of the impact on their data of these techniques – especially any technique which employs compression technology. Dynamic stereo imagery also represents an interesting challenge for image compression systems since it is important to maintain the correct timing relationship between left and right eye images. Some compression techniques are unable to present truly consistent images between the left and right eye and lead to the introduction of strange visual artefacts in the stereo image. An extreme case would be one where the right eye image contained significantly more information than the left eye (this mainly applies to partially overlapped stereo viewing systems). As soon as movement is introduced by the user this can lead to visually perceptual differences between the images, caused by the compression technique.

8. Conclusions and Future Work

This paper has looked at the importance of real- or near real-time user interaction with a scientific visualisation as a means of more accurately interpreting meaning from the dataset. We have discussed the issue of interaction delays introduced by system architecture that could affect image understanding. Such delays could for example result from the response time of the remote visualisation system or even the compression-decompression method. Issues such as where to perform the visualisation rendering (local or remote) have been discussed.

Work has started on looking at alternative lossless compression techniques that incorporate variable or adaptive compression. Popular compression techniques such as wavelet transform and discrete cosine are too computationally intensive to support interactive frame rates. Our aim is to develop a compression technique that uses a lossy compression technique whilst the user is manipulating data and then as soon as the user interacts less with the system then a lossless compression method is substituted to preserve

the image quality. The adaptive technique has the advantage of ensuring the final image is presented at the highest possible resolution with no distortion but has the benefit of supporting interactive visualisation. Future work will look at more advanced semi or fully automatic supervisory control systems where user specified goals are used to drive the computational steering tasks. The role of the user then becomes one of monitoring higher level intermediate or final results of the simulation and then providing a higher level steering input based on goals or simulation targets rather than manipulation of discrete parameters. This kind of higher level simulation intervention could make the user interface much simpler.

9. Acknowledgements

We would like to acknowledge the support of all the application and computer scientists within the RealityGrid project (University College London, University of Manchester Computing, University of Oxford, Loughborough University, Edinburgh Parallel Computing Centre). In particular we would like to acknowledge Peter Coveney (UCL) and Stephen Pickles (University of Manchester Computing) for their encouragement and assistance in assessing real world applications. We also wish to acknowledge the time generously given to us by the many researchers involved in RealityGrid as we undertook the in-depth human factors audit.

This work was carried out under grant GR/R67699 from the EPSRC and within the RealityGrid project.

10. References

Card, S. K., Robertson, G. G., and J. D. Mackinlay (1991). The information visualizer: An information workspace. Proc. ACM CHI'91 Conf, New Orleans, LA.

Foster, I. and C. Kesselman (1999). The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann.

Fowler, J. E. (2000). Evaluation of SGI Vizserver, NSF Engineering Research Center, Mississippi State University: 12.

Gregory, R. L. (1993). Seeing by exploring. Pictorial communication in virtual and real environments. S. R. Ellis, M.K. Kaiser, and A.J. Grunwald, Taylor and Francis.

Hamming, R. W. (1962). Numerical Methods for Scientists and Engineers, McGraw-Hill.

Hey, T. and A. E. Trefethen (2003). The Data Deluge: An e-Science perspective. Grid Computing - Making the Global Infrastructure a Reality. F. Berman, G. Fox and A. J. G. Hey, Wiley.

Ittelson, W. H. (1968). The Ames Demonstrations in Perception. New York, Hafner.

Kalawsky, R. S. and S. P. Nee (2004). e-Science RealityGrid - Interim Human Factors Audit Requirements and Context Analysis, Loughborough University.

Miller, R. B. (1968). Response time in man-computer conversational transactions. Proc. AFIPS Fall Joint Computer Conference.

Myers, B. A. (1985). The importance of percent-done progress indicators for computer-human interfaces. Proc. ACM CHI'85 Conf., San Francisco, CA.

Ohazama, C. (1999). OpenGL Vizserver White Paper, Silicon Graphics Inc.,.

Russell, G. and R. Miles (1987). "Display and perception of 3D space-filling data." Applied Optics **26**(6): 973.

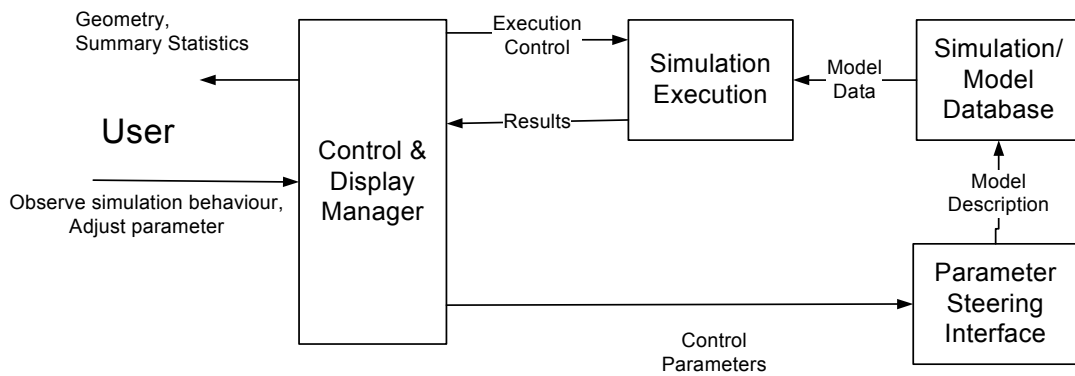


Figure 1: Computational Steering System

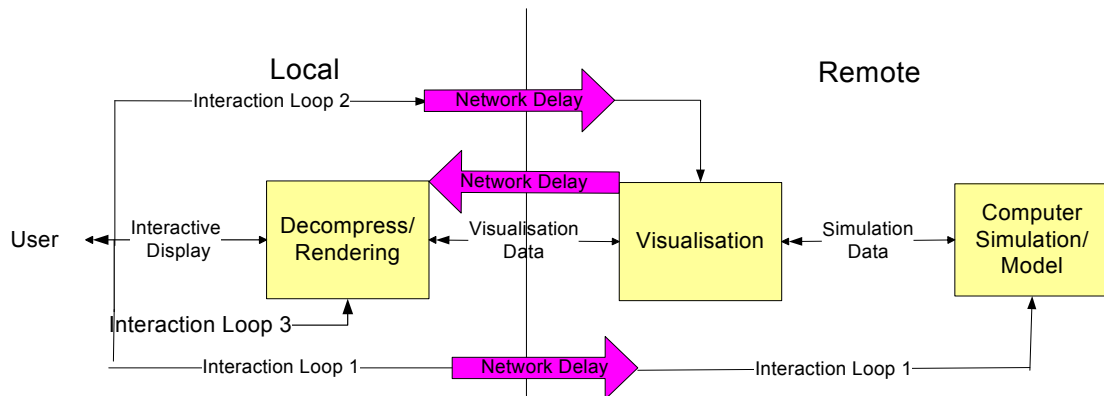


Figure 2: Interactive loops

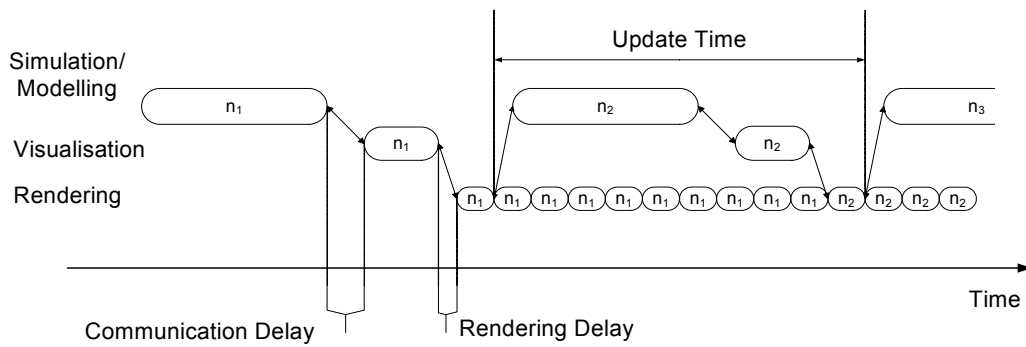


Figure 3: Timing relationships between interactive loops

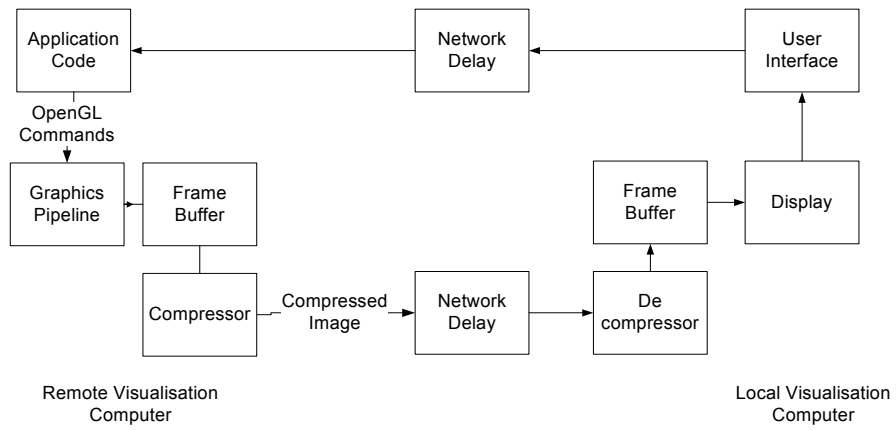


Figure 4: Vizserver