

Putting You In The Picture: Enhancing Visualization With A Virtual Environment

D.R.S. Boyd*
CLRC Rutherford Appleton Laboratory
Chilton, Didcot, Oxon OX11 0QX
United Kingdom

J.R. Gallop†
CLRC Rutherford Appleton Laboratory
Chilton, Didcot, Oxon OX11 0QX
United Kingdom

J.P.R.B. Walton‡
The Numerical Algorithms Group Ltd
Wilkinson House, Jordan Hill Road
Oxford OX2 8DR, United Kingdom

Abstract

We describe recent work aimed at the integration of a visualization package and a virtual reality system, with the emphasis on the re-use and enhancement of commercially available products. The objective of the work is to determine what advantages (if any) are to be gained by combining the visualization package (which is well-suited for the conversion of numerical data into appropriate geometrical representations) with the comparatively novel immersive navigation methods of virtual reality.

Our system—called VIVRE—was designed and built to meet the requirements of users who were to apply it to real industrial problems. They found that the use of the new navigation methods were advantageous for large datasets, and that direct interaction with elements of the visualization (for example, cutting planes) gave new insight into their data.

CR Categories: I.3.4 [Computer Graphics]: Graphics Utilities—Graphics Packages; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality;

Keywords: Visualization systems, virtual reality, user-centered navigation, system evaluation

1 INTRODUCTION

Today’s user seeking understanding of their numerical data can choose from a number of popular visualization packages such as IRIS Explorer [1] or AVS/Express [2]. In systems of this type, the user constructs a network of modules that produces a 3D representation of their data which is viewed in a 2D window on a monitor screen. This *data-centered* interaction mode, in which a camera is manipulated around the 3D representation, can be a powerful way of interacting with some forms of data but another paradigm might, under some circumstances, offer greater flexibility and appear more intuitive. For example, the user might find it better to be freely navigating through the 3D space whilst looking around. This *user-centered* approach is the one which is often employed within virtual environments.

This paper describes the application of this user-centred approach to data visualization by using a virtual environment (VE) to construct a new user interaction interface—called VIVRE—for a data visualization system. The objective of this work is to allow the user to “steer” the visualization task via the VE to more easily gain the insight they are seeking from their data, while retaining the functionality and extensibility of the visualization system.

Unlike previous work in this area, we have concentrated on using commercial visualization and VE systems in order to leverage existing user experience. The project has also been driven by users’ requirements and VIVRE is being applied to their real commercial and industrial problems.

2 INTEGRATING VISUALIZATION AND VIRTUAL ENVIRONMENTS

2.1 Previous work

There have been several attempts to integrate visualization and VE techniques—see [3] for an overview. A number of approaches have been tried:

- taking an existing visualization system and adding improved navigation and interaction capability [4], [5], [6],
- taking a VE system and adding visualization capability [7], [8],
- developing a custom-designed integrated visualization and VE system [9], [10], [11].

Each of these approaches has its strengths and weaknesses. Extending a visualization system exploits existing user expertise in that package but still lacks a system architecture that has been designed from first principles to support real-time interaction. Efforts to add visualization functions to a VE system have been able to take advantage of the fast response to user interactions but have tended to focus on the needs of particular application areas where the functionality required is well-defined and domain-specific. Custom-designed solutions have demonstrated some of the benefits of closer integration but because they are not widely available they lack a large experienced user base within industry.

*D.R.S.Boyd@rl.ac.uk

†Julian.Gallop@rl.ac.uk

‡jeremyw@nag.co.uk

2.2 The VIVRE system

In VIVRE [12], we sought to use two general purpose visualization packages and to couple each of them to an existing, relatively mature VE system. IRIS Explorer [1] and AVS/Express [2] are well-established systems which were already being used by the VIVRE end-users, thus taking advantage of existing investment and experience. Although others could have been chosen, dVISE [13] was selected as the VE system. It is a mature commercial product which provides the required extension mechanisms and supports a variety of user I/O configurations.

By developing the interface between these packages, we were able to provide our end-users with a demonstrator system that enabled them to assess the practical benefits of the user-centered approach to visualization steering for their applications.

Four groups of end-users drove the technical directions for the VIVRE system. Their applications covered a wide range of domain and scale, including the visualization of:

- combustion and gas flow in industrial processes;
- potential leakage paths surrounding underground storage caverns;
- the dynamic operation of a flexible fluid control valve; and
- complex natural microstructures and their properties.

The users required a range of visualization actions (such as moving cutting planes, changing isosurface thresholds, etc.) to be accessible directly from the VE. Other actions such as substituting a different visualization function in the network required the user to interact directly with the visualization system. The users' requirements were amalgamated into a single prioritized list which guided the development work and provided the basis for user evaluation of the VIVRE system.

Building on an existing VE system such as dVISE ensured that a range of hardware devices were supported in VIVRE, from the 2D window plus mouse, through a 6 degree of freedom input device, to stereoscopic projection and immersive head-mounted display.

3 TECHNICAL APPROACH

3.1 Extension mechanisms for the underlying systems

The functionality of both the visualization systems used in this work can be extended by supplementing their module suites (which already include standard visualization *mapper* modules which produce geometry—a cutting plane, say—from numerical data) with new ones. We wrote these for managing data coming from the VE and for associating an object identifier with a piece of geometry.

Figure 1 shows how placing these modules in the visualization network before and after a standard mapper module creates a *geometry strand* which is responsible for a single piece of geometry (a cutting plane or isosurface, for example) in the scene. In addition, we wrote two other modules—one for sending geometry to the VE and one for responding to user change requests from the VE. Only one instance of each of these modules is required in the network (by contrast, there are as many geometry strands as there are pieces of geometry in the scene—see § 3.3, below). Adding them to the

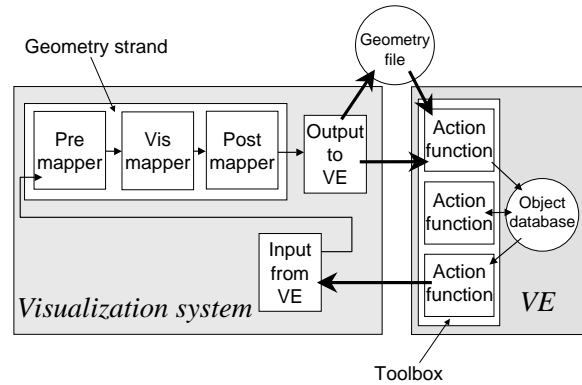


Figure 1: The VIVRE architecture, showing the way in which each piece of geometry is passed from a *geometry strand* in the visualization system (on the left) to the VE (on the right). Its arrival in the VE triggers an *event-based action function* in the VIVRE *toolbox*, which sends a signal to the dVISE object database. Object manipulations in the VE trigger another action function, which passes data to the geometry strand.

network allows each mapper module to respond to requests from the VE (such as a new position for the cutting plane) and for its geometry to be passed to the VE. In this version of VIVRE, this is done by binary transcription via a disk file.

dVISE allows the programmer to associate processing *actions* with events initiated by the user or by objects in the VE. The actions could either be already existing in dVISE (for example, changing the color of the currently selected object), or be developed in VIVRE via the dVISE API. We packaged up these programmer-supplied action functions into a single plug-in which is activated by the dVISE runtime system when required. The dVISE system also provides an interactive, immersive toolbox and a language for programmer-specified toolbox interactions. This allows the user to interact with the visualization without leaving the VE.

One of these action functions in the VIVRE toolbox (see figure 1) is triggered by the arrival of geometry in the VE. The function sends a signal to the dVISE object database, whereupon the geometry is added to the virtual world (see § 3.2, below). In a similar way, manipulating an object in the VE can trigger an action function which passes data (such as a new cutting plane position) to the pre-mapper module in the visualization system.

3.2 Dynamic changes in the virtual world

Although virtual worlds are conventionally crafted with great care in advance, our use in VIVRE requires that it responds to user interactions (for example, changing visualization parameters) resulting in totally new scenes. We therefore pass the initial world to dVISE as a skeleton which contains attachment points for objects to be added or replaced, depending on the multiple objects transmitted from the visualization system. This also requires the user to explicitly acknowledge the passage of new data into the VE system from the visualization system. This has the benefit that the user is not surprised by the appearance of a new

scene in the VE, and does not require the user to leave the VE.

3.3 Multiple objects

In a visualization application, it is common for several different sorts of visualization representations and multiple instances of each to be present in the scene. For example, a set of isosurfaces, cutting planes and an enclosing geometry such as a furnace. In VIVRE, it was envisaged that each could be separately manipulated by the user, and is therefore essential that these be structured in the VE as separate objects.

The structuring of the scene into objects is determined by the visualization network which has parallel geometry strands corresponding to each portion of the network capable of generating geometry (see Figure 1, where a single strand is shown). Each strand contains a VIVRE data preparation module which allows the application developer or user to specify the object name which is to be used by the VE system. Since one of the features of the VE system is to display a hierarchical view of the object collection, user-controlled labelling assists user recognition. VIVRE also attaches a system-defined unique name to the object so that subsequent input may be correctly associated with it, which allows a user change to be passed to the correct module in the visualization system.

3.4 Geometric conversion

Both visualization systems can write geometry in widely recognised formats such as Open Inventor and VRML 2.0 and dVISE provides tools that can read these formats. A problem with this standard method in dVISE is that colors associated with vertices—a common way of expressing variation of color in a visualization scene—are not represented by dVISE as colors to be shaded, but instead as colors which are *due to* shading effects (from—say—a radiosity calculation). This meant that the vertex colors would be ignored by the VE. The solution was to pass the color information as a texture map on the geometry. Texture coordinates are interpreted in dVISE prior to the shading calculation and the texture map embodies the color map.

If no specific optimization is done, dVISE can deliver fast and steady navigation by using an adaptive method based on [14]. Greater control over the loss of detail is possible, but the default method of exercising this in dVISE relies on specific percentages of the scene, which can arbitrarily lose important detail. An experimental solution to this was to use a criterion based on the projected size of features in the scene. Although this method was not available in the dVISE recipe editor, it was possible to implement this using feature-size controls in its lower-level toolset.

3.5 Contextual information for the visualization scene

In addition to controlling the course of the visualization application, the users also required information to be presented in the VE to assist the user's decision making, particularly if the user is immersed. For example, the user may wish to know the current value for an isosurface threshold.

3.6 Control of the visualization system from the virtual environment

Effective visualization requires active control over the visualization processes. Since in this work, the user's interface is via the VE (including the use of immersion where available), VIVRE allows certain information to be returned from the VE to the visualization system, where VIVRE modules identify what has been returned and route it to appropriate places in the visualization network. The types of information returned are:

object identification: this allows the VE to identify an entry in an application database, possibly make changes and recalculate the visualization output.

new position of selected object: the user via the VE can drag an abstract object such as a cut plane and upon a simple signal (such as releasing a button), the identification and coordinates are returned, allowing new information to be recalculated in the visualization network and redelivered to the VE.

return selected parameter value: any parameters—such as an isosurface threshold or an application-dependent parameter—in the visualization network can be registered with the VE. This allows the user to control any such parameter via the VE, which may cause the visualization output to be recalculated.

Other more infrequent operations, such as editing the visualization network, can be handled by direct interaction with the visualization system.

4 EVALUATION RESULTS

The VIVRE system has been implemented under both IRIX and Windows NT. Figure 2 shows a screenshot from a VIVRE session running on a conventional monitor, with the IRIS Explorer network containing the VIVRE and application modules, and the dVISE display. Experiences from the VIVRE users (none of whom had previously worked with a VE) has been largely positive. Although no difference in performance between the original visualization system and the VE could be detected for small datasets, there was some consensus that the VE allowed a comparatively smooth navigation of *significantly large* datasets, to an extent that user actions were translated into navigation in a direct way not normally associated with general purpose visualization systems. Similarly, users reported being able to see advantages of the user-centered approach for visualizations which resulted in a large collection of 3D objects. In addition, the direct interaction with cutting planes in the 3D scene was said to be helpful. The user was able to manipulate a plane's position and orientation in the VE; upon releasing the plane, its new location is passed to the visualization system which recalculates the image to be displayed on the plane.

Not all of the VIVRE users have worked with an immersive system. Of those that tried it, all reported getting better insight into their data, particularly for data where spatial relationships are important, and—once again—where there is a large number of 3D objects.

5 CONCLUSIONS

The work described in this paper has shown that it is technically feasible to link commercial data visualization and VE

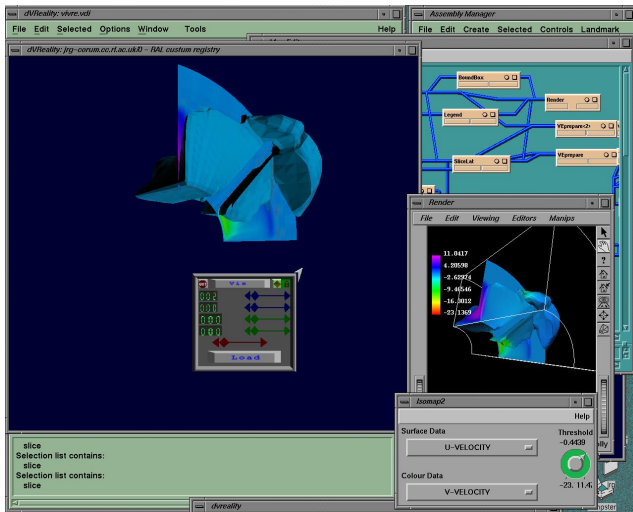


Figure 2: Screen shot from a VIVRE session running under IRIX. Windows from the visualization system (in this case IRIS Explorer) are on the right; the Map Editor containing a group of modules connected together to form an application is at the top. The other windows on the right are control panels for two of the modules (including the Render module, which is displaying the 3D visualization). The large window on the left is the dVISE desktop viewer, which is also displaying the visualization. The input data comes from the simulation of flow around a fluid control valve.

software systems to provide users with an integrated environment which offers access to the capabilities and strengths of both systems. This new environment can exploit the computational performance and graphics capability of high-end HPC systems while also being accessible to PC users. We have developed a prototype system which provides a sufficient range of visualization functions that users are able to develop useful application scenarios. Their evaluation has shown them that the user-centered approach to visualization and navigation can be advantageous for datasets that are large, or which result in a large number of 3D objects. They reported that direct interaction with elements of the visualization such as cutting planes was a helpful addition to their more conventional means of data exploration. The users' scenarios provide the basis for evaluating the degree to which this user-centred experience of navigating and interacting within an immersive VE will offer measurable commercial benefits compared with the data-centric systems used previously.

Future work here includes the extension of the system to include other visualization techniques such as streamlines, the incorporation of time-dependent data and the improvement of color control and the interrogation of intersections between objects. Finally, we note that NAG has said that it intends to incorporate the VIVRE system into a future release of the IRIS Explorer visualization toolkit.

6 Acknowledgements

VIVRE is a demonstrator project in the High Performance Computing and Networking sector of the European Commission 4th Framework Programme. The authors wish to thank the Commission for their support, and the VIVRE part-

ners for their co-operation and contribution to the project—particularly Mark English from Tessella Support Services plc, who ably managed the project and contributed to the design of the system. Finally, JW thanks Silhacene Aid for his invaluable help during the preparation of this paper.

References

- [1] IRIS Explorer. see http://www.nag.co.uk/Welcome_IEC.html.
- [2] AVS/Express. see <http://www.avs.com/products/expovr.htm>.
- [3] J.R. Gallop. Virtual reality—its application to scientific visualization. In *Eurographics 96 Tutorial 3*, 1996.
- [4] W.R. Sherman. Integrating virtual environments into the dataflow paradigm. In *Proceedings of 4th Eurographics Workshop on Visualization in Scientific Computing*, 1993.
- [5] H. Haase, M. Goebel, P. Astheimer, K. Karlsson, F. Shroeder, T. Fruehauf, and R. Zeigler. How scientific visualization can benefit from virtual environments. *CWI Quarterly*, 7(2):159–174, 1994.
- [6] A. Fuhrmann, H. Loeffelmann, D. Schmalsteig, and M. Gervautz. Collaborative visualization in augmented reality. *IEEE Computer Graphics and Applications*, 18(4):54–59, 1998.
- [7] T. Fruehauf and F. Dai. Scientific visualization and virtual prototyping in the product development process. In *Proceedings of the 3rd Eurographics Virtual Environments Workshop*. Springer-Verlag, 1996.
- [8] L. Sastry, D.R.S. Boyd, R.F. Fowler, and V.V.S.S. Sastry. Numerical flow visualization using virtual reality techniques. In *Proceedings of the 8th International Symposium on Flow Visualization*, 1998.
- [9] S. Bryson and C. Levit. The virtual windtunnel: An environment for the exploration of three dimensional unsteady flows. *IEEE Computer Graphics and Applications*, 12(4):25–34, 1992.
- [10] C. Cruz-Neira, D.J. Sandin, and T.A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *Proceedings of SIGGRAPH 93. Computer Graphics Annual Conference Series*, pages 135–142, 1993.
- [11] H. Haase. Symbiosis of virtual reality and scientific visualization system. *Computer Graphics, Proceedings of Eurographics 96*, 15(3):442–451, 1996.
- [12] VIVRE. see <http://www.tessella.co.uk/projects/vivre/vivre.htm>.
- [13] dVISE. see <http://www.ptc.com/products/division/>.
- [14] T. Funkhauser and C. Sequin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Proceedings of SIGGRAPH 93. Computer Graphics Annual Conference Series*, pages 247–254, 1993.