Alternative Archaeological Representations within Virtual Worlds

Jonathan C. Roberts and Nick Ryan

Computing Laboratory, University of Kent, Canterbury, Kent, UK

Tel: *+44 (0)1227 827699*
Fax: *+44 (0)1227 762811*
e-mail: *J.C.Roberts@ukc.ac.uk N.S.Ryan@ukc.ac.uk*

Traditional VR methods allow the user to tour and view the virtual world from different perspectives. Increasingly, more interactive and adaptive worlds are being generated, potentially allowing the user to interact with and affect objects in the virtual world. We describe and compare four *models* of operation that allow the publisher to generate views, with the client manipulating and affecting specific objects in the world. We demonstrate these approaches through a problem in archaeological visualization.

Keywords: visualization, VRML, archaeology.

1. Introduction/Motivation

Ancient buildings and landscapes have proved to be a popular subject for computer based visualizations. From the early solid models of Winchester Old Minster (Reilly, 1989) to the recent 'virtual Stonehenge' (Superscape, 1997), corporate laboratories have used them as a vehicle for demonstrating their latest technologies. Static three-dimensional models and VR techniques are now widely used by architectural historians, archaeologists and museums, and the World Wide Web has become an important means of delivering representations of the past both to other professionals and to the public.

As with modern museum displays, many of the images and virtual worlds found on the Web today appear very realistic. Indeed, they may sometimes be too realistic, particularly when they are based on very incomplete evidence but convey an impression of a firm knowledge of the past. Some archaeologists have rightly criticised the tendency of computer-based visualizations to be driven by the need to demonstrate advanced graphics techniques, rather than by archaeological considerations (Miller & Richards, 1995). Care is needed to strike a balance between the desire to produce realistic images and the need to express the tentative nature of an interpreted form (Ryan, 1996).

Archaeological evidence for a building may consist of little more than a few fragments of wall foundation, perhaps severely disturbed by later human or geological activity. A building may have been altered or rebuilt on several occasions, often with major changes in its appearance and function. Unfortunately, most current visualizations of the past provide a single view, representing not just a snapshot in time but also only one of many possible constructions of the surviving evidence. Much more is needed to capture the full range of both temporal change, (see, for example, Johnson & North, 1997), and possible interpretations (this paper).

Just as an architect might wish to give users access to both 'before' and 'after' views of a new development, archaeologists need a way of presenting alternative interpretations and changes through time. The essence of the problem is to allow the user to both tour and adapt the world, within certain constraints (Wood *et al.*, 1996; Walton & Knight, 1997). The constraints we employ are that there is a point-to-point discovery and adaptation of the archaeological model, rather than group or collaborative interaction (which we leave to a further study) and that the models contain a basic structure, with some components that may be augmented or adapted. For example, if the model contains stairs, then there will always be stairs in the augmented view, but their number or position may be altered.

We have discovered four models of operation; these allow the archaeologist or publisher to generate views of the historic site, with the client manipulating and affecting specific objects in the world.

We present an implementation of these ideas, using VRML (Moving Worlds, 1997), illustrated by models of the roman theatre in Canterbury. This building underwent at least two major phases of construction and several interpretations have been published. By manipulating basic design parameters the end user can experiment with and compare a range of possible reconstructions of the theatre, features difficult to achieve with current virtual tours.

2. The categorization

Over the past few years, we have seen the rapid introduction and development of VRML (Hartman & Wernecke, 1996). From its inception at the 1994 World Wide Web conference, it has developed through the use of the Open Inventor file format as a basis for VRML 1.0, the multiple proposals for VRML 2.0 in 1996, to the finalized VRML 2.0 version in 1996 (Pesce, 1995). VRML 2.0 provides advanced features to allow complex static worlds, a high level of interaction, animation and movement with behaviour modelling and prototyping to create new objects. Both VRML 1 and VRML 2 may be used to generate alternative archaeology views, hence, our alternate-world categorization complements the development of VRML, incorporating and classifying the earlier static models, generated using VRML 1, and the moving-worlds models of VRML 2.

Each of our categories allows the presentation of alternative interpretations of the underlying data and may be implemented using multiple techniques. Each has its different advantages and disadvantages. The publisher needs to present a well rounded-argument, allowing the observer to form an opinion of the data. Appropriate background information must be provided, with multiple interpretations of the world and the assumptions of each archaeologist. Ideally, the viewport on the world would stay constant, allowing the user to obtain a direct comparison between diverse interpretations. Feedback, in the form of legends, keys and annotations, to provide additional information and focus the user on particular aspects of the archaeological data would also be beneficial.

We name the parts in our categorization, with increasing complexity and user control: *Require New*, *Switch Change*, *Functional Change* and *Program Run*. We describe each category, several implementation flavours and their relative advantages and disadvantages. The distinguishing features of this categorization are found in the complexity of implementation and the tightness of the visualization cycle. The visualization cycle represents how quickly, and by whom, the alternate worlds are generated, published and controlled. A loose cycle is represented by the client requesting a new version of the data from the publisher, whereas a tight cycle is produced when the alternate view is processed at the client side.

The visualization cycle is often divided into Filter, Map and Display processes (Haber & McNabb, 1990). For the purpose of this paper we have coerced these into a single 'generate world' process. It can be argued that these processes generate a long visualization cycle, some issues for shortening the visualization cycle are discussed by (Roberts, 1996). Our problem of operating and controlling alternative realizations is similar to the generation of Visualizations with different parameters over the World Wide Web described in (Wood *et al.*, 1996). There, they present different scenarios that transfer responsibility for the creation of the visualization from the publisher to the client. They provide a 'Web server' that takes a visualization request from the user, collated by an HTML form, generating a VRML visualization of the data using IRIS explorer attached through a CGI script. A different realization results in a new request and a new VRML model. This work is coincident with our first model, *Require New*.

2.1 Require New

The archaeologist generates multiple instances of the world, describing a range of possibilities and arrangements of the 'reconstructed' buildings. The client down-loads each model as required (Figure 1).

Examples of this category would typically take the form of a 'tour' in which the archaeologist presents the different interpretations, by developing discrete realizations, interspersed and overlaid with appropriate descriptions, references and annotations.

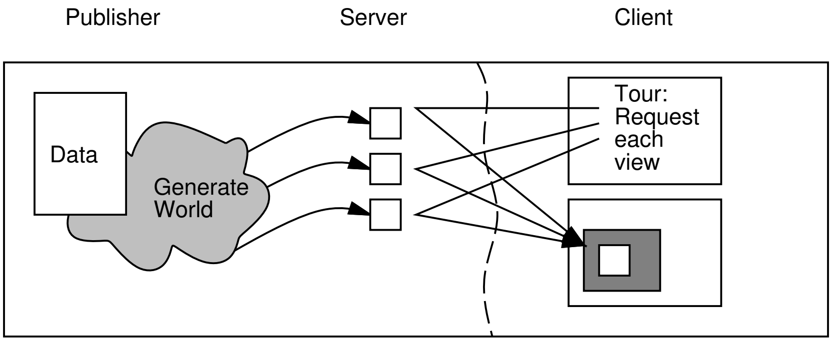

Figure 1: Model 1, Require New

2.1.1 Implementation

There are many flavours of tours so this approach may be implemented using several different techniques. The tour can be a mixture of HTML, images, movies, sound and VRML. A two dimensional version may be generated by using HTML, images and movies, with the user taken through the possibilities by selecting a *next* button to load the subsequent page. The virtual Frog dissection (Robertson *et al.*, 1995) visualizes three dimensional frog data, generating GIF images over the Web through a CGI script.

The three dimensional version uses HTML, text and images as the two-dimensional version but adds VRML worlds. These worlds may be generated dynamically, as in the model of Wood, Brodlie and Wright (1996) which uses an *active* publisher in the form of a CGI script and visualization system to generate and publish the required instances. A solely VRML tour version may be developed, where the user is 'teleported' to another view, (e.g.

the Tenochtitlán city, Hartman & Wernecke, 1996), however, this is just another view of a single realization, not an alternate interpretation.

2.1.2 Advantages

The publisher retains a high level of control over what is presented and the order of presentation. The user can be guided round, and told where and what to look at. The user then only needs to have basic interaction skills and little specialist knowledge. Pictures — GIF, PNG (W3C, 1997) — could be used instead of dynamic VRML 1.0 or VRML 2.0 (moving worlds specification). Models may be relatively small and quick to load, but there is an overhead in loading each new view as it is required.

2.1.3 Disadvantages

A limited or biased viewpoint can be easily, even unconsciously, presented. The user is not so immersed in the three-dimensional world, nor do they have much control over what is being viewed; with the publisher providing the choices for the user. Appreciation of scale and making comparisons between observations and viewpoints are difficult unless the publisher chooses to provide images of the same scale and viewpoint on a single page. Because each separate world visualization containing a different interpretation is complete in itself, the overall download time may be higher than for other categories.

2.2 Switch Change

Here, the publisher generates the multiple interpretations, placing them in one document. The client requests the single document, and may switch between the various configurations and navigate the world, see Figure 2.

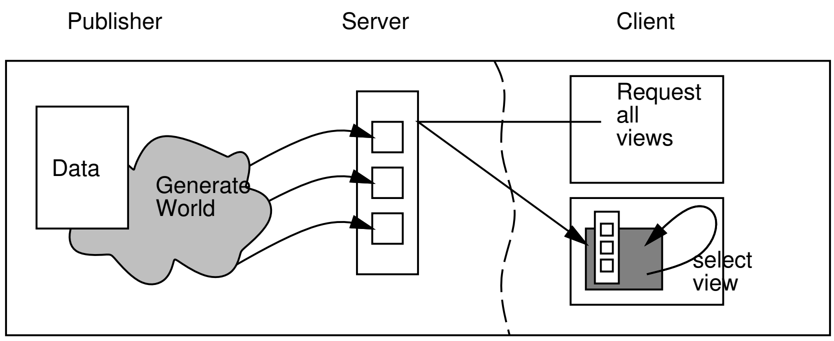

Figure 2: Model 2, Switch Change

2.2.1 Implementation

The multiple models are combined together into one document and embedded in a VRML switch construct, enabling zero or one of the nodes, specified in the choice field, to be displayed. A simple interface may then be used to toggle or select the required view. The user's choice can be generated from a simple sensor (e.g. touch-sensor or time-sensor) through a script node or from an external authoring interface (EAI) using, say, Netscape's LiveConnect (Marrin, 1997).

2.2.2 Advantages

As all of the models are sent together the initial download time is increased. However, each instance does not require a separate description of the environment and other 'paraphernalia', so the overall retrieval time may be reduced. Moreover, if the initial model file uses the 'Inline' construct, then the browser may not load the alternate views until they are required and the overall download time is increased.

The alternate views can be easily compared, because the navigation position of the browser is kept constant when the instances are loaded. This allows particular aspects of the model to be compared, within the same context.

In this model, the data values are fixed by the publisher. This would be useful where a series of distinct interpretations are to be depicted. The EAI method has an additional advantage, in that, the HTML can directly effect the version that is being viewed in the VRML-browser, therefore, relating the world back to the written information, providing useful contextual information.

2.2.3 Disadvantages

The 'switch' version provides little feedback to the user, who may find if difficult to decide which argument fits what world. The EAI version provides extra contextual information, but still does not provide a detailed comparison between interpretations. This feedback problem may be rectified by sending an appropriate user interface (developed in VRML) with a functional model of the site. Nor is it possible to present a continuous change such as varying the height of building. The provision of a user interface and handling continuous change are illustrated by our next category, *Functional Change*.

2.3 Functional Change

The publisher generates a world with objects and characteristics that may be easily changed (see Figure 3). This allows, for example, the number of arches to be incremented or reduced, or the building height to be changed. The values may be altered by a control panel that is encoded and sent with the virtual model. Portions of the world may be relocated, resized, or otherwise altered.

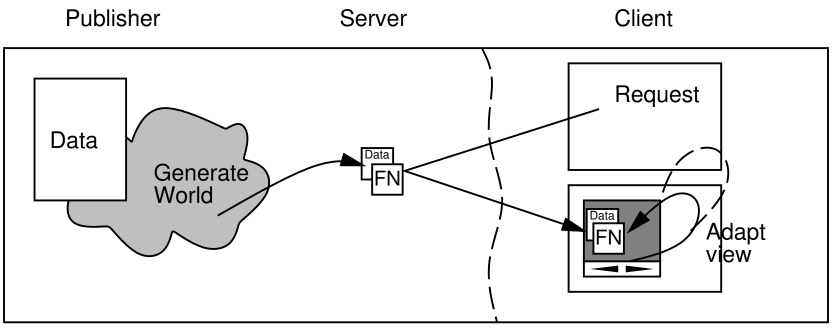

Figure 3: Model 3, Functional Change

2.3.1 Implementation

There are several flavours of implementation for this model. In each case the VRML prototyping mechanism is used to implement the nodes of the world. The nodes are routed to a central 'Calculator' prototype, used to calculate the bounds and values of the dependencies. The Calculator is also connected to the user interface to allow the user to control the value of the parameters.

The first flavour provides a user interface entirely in VRML, the output data streams are ROUTE'd to the Calculator prototype (Figure 4). The calculator prototype allows the calculations of the bounds to be grouped in one place, reducing the number of routes in the model and allowing changes to the dependencies to be altered in one place in the code. The user interface is generated in a Head Up Display (HUD) configuration. The HUD nodes maintain the position of the user interface on the browser screen, independent on the viewing angle or position, thus keeping the controls at the user's finger tips.

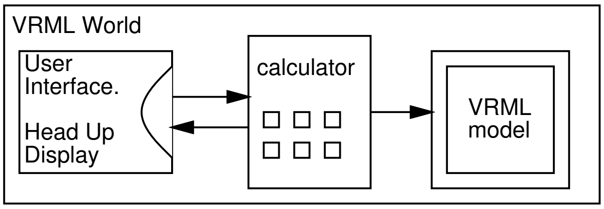

Figure 4: Functional Change - VRML model configuration

A second implementation flavour uses the External Authoring Interface (EAI) to link Java, JavaScript, HTML or other external languages together (Figure 5). A version may be implemented using an HTML/JavaScript form, as the user interface, connected through a Java applet, to calculate the values and dependencies, and linked through Netscape's LiveConnect (Hoque, 1997) external interface to control the VRML world. Likewise, a Java applet with sliders and text inputs could be used to provide the user interface.

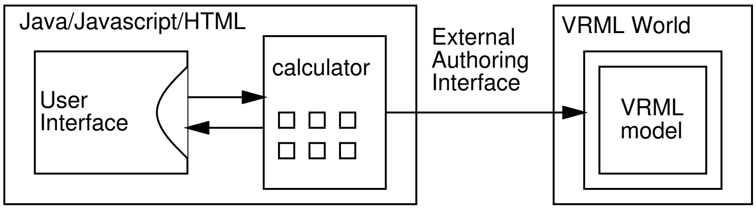

Figure 5: Functional Change - VRML and EAI configuration

Thirdly, a full Java version could be implemented, visualizing the world entirely in a Java applet. The application and user interface (generated using Java scrollbars, for example) could then be honed for the specific subject being viewed. The VRwave browser (Andrews & Pichler, 1997), a successor to VRweb (Pichler *et al.*, 1995) uses OpenGL or Mesa as the rendering library.

2.3.2 Advantages

The first two models provided fixed values selected by the publisher, whereas, this model allows the values to be chosen by the user. Direct feedback is given to the user as they adapt the values, immediately changing the model and updating the dependent variables and values on the HUD.

The model is described by two parts: the user interface and the prototypes. This results in a larger single model, with an increased initial download time. However, no other models are required and the speed of the alterations is dependent on the client hardware browser software. Moreover, as the model and user interface communicate through a calculator node; the calculator can be extended to include a buffer. This buffer need not send every event to the model, only updating the initial and the last user's request. In practice, this is achieved by routing another event (isActive) from the sensor node to the calculator node, describing when the sensor is active and when it has been released.

2.3.3 Disadvantages

The presentation of contextual information that matches values in the model to specific arguments in a written document is difficult to achieve with the integral user-interface version. However, the second and third flavours can be easily connected to an online document, enabling the values to be changed, depicting a particular argument when, say, a new HTML page is loaded.

2.4 Program Run

A version of the data and program is published. The user then runs the program on their local machine to generate different views and models of the displayed world (see Figure 6)

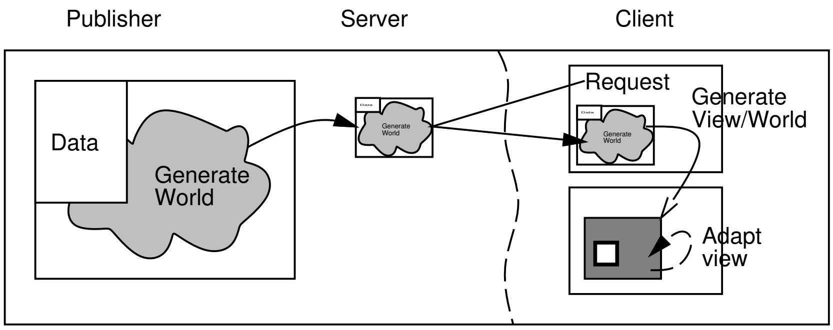

Figure 6: Model 4, Program Run

2.4.1 Implementation

A Java applet may be published and executed on the client machine to generate various views of the world. The applet can also be designed to control and alter the parameters of the displayed world through an EAI interface. Alternatively the applet could generate an appropriate user interface at the same time as generating the world.

2.4.2 Advantages

This category allows the user to both alter and affect the virtual world and to generate a complete new instance of that world. This allows the user to save copies of their work and publish their own instances, possibly accompanied by their own arguments and interpretations. In essence, this category allows a visualization system with appropriate data to be published and operated at the client side.

2.4.3 Disadvantages

The user is given complete control over how the worlds are displayed (within the constraints of the program). Consequently, the publisher has little control over what the user generates and views, so it is difficult for the publisher to present a reasoned argument between alternative views. The size of the program and data would be substantially larger than the single instances generated by the other categories.

3 Implementation/Results

We have implemented versions of the first three categories using VRML 2.0, JavaScript (vrmlscript) and HTML, and using the CosmoPlayer 1.0 browser (Software, 1997). We describe in greater detail an implementation using our third category of models of the Canterbury Roman theatre.

This implementation includes a user interface, a HUD and a prototyped model in a single document. The HUD and model are glued together using the calculator node. This node contains multiple dependencies: for example, by decreasing the width of the seats, so the number of seats and the height of the building increase. These dependencies are shown in Figure 7.

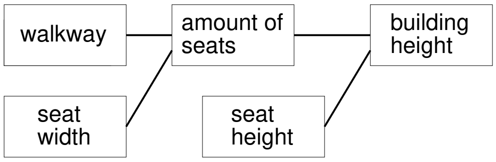

Figure 7: Functional Change - dependencies

We have routed the values taken from the interface sliders through the HUD to affect the dependent values of the model, giving dynamic feedback to the user as the world is changed. The behaviours of the calculator and other nodes are implemented in JavaScript (vrmlscript) which gives a shorter initialisation time than a Java behaviour.

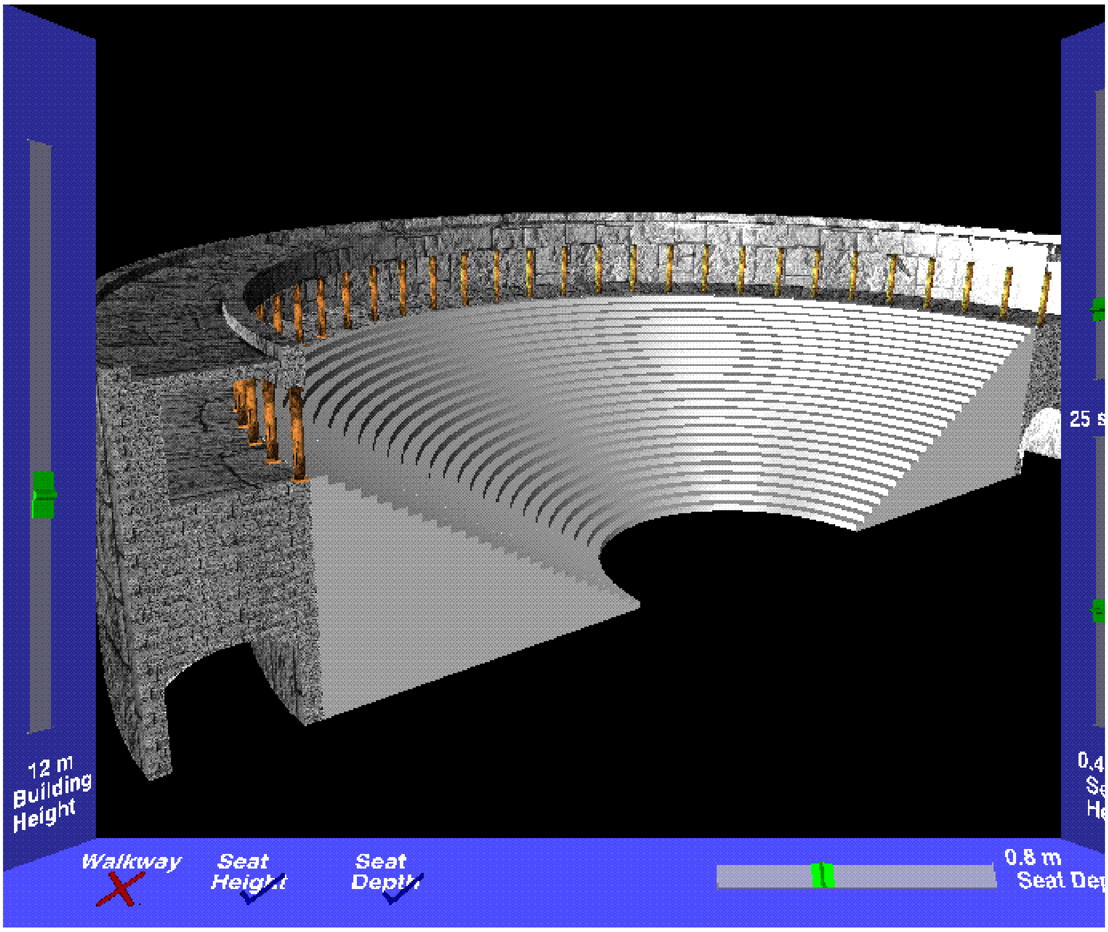

Figure 8: Example of the *Functional Change* implementation with HUD user interface. The model shows a simplified representation of the Canterbury Roman theatre (with stage and surrounding buildings removed). The user may adjust the building height, seat number or dimensions, or add a walkway between the seats to examine a range of possible appearances.

4 Discussion of categories

We have presented four categories describing methods to control (and publish) alternative views or interpretations of archaeological evidence. Alternative names for the categories could include a tour, a flipbook, a method sending the user interface, and a method of publishing the program. Another view of these models is to look at what is being changed in the scene, this being either the behaviour or the model itself. A *static* world is generated if the model (or behaviour) does not alter. A *dynamic* world is generated if the model (or behaviour) is changed. This latter categorization fits well into our schema by providing the following four states:

Category	**Name**	**Model**	**Scene behaviour**
1	Require New	Static	static
2	Switch Change	Dynamic	static
3	Functional Change	Static	dynamic
4	Program Run	Dynamic	dynamic

Examples implementations can be found on our web site at <URL: http://www.cs.ukc.ac.uk/people/staff/nsr/arch/vrsig97/ >

5 Future Work

The work described here began with a problem that arose when one of us attempted to produce a simple model of the Canterbury Roman theatre. Each of the existing published drawings and artists' impressions assumed different dimensions. In some cases, these dimensions resulted in an impossibly steep rake to the seating, resulting in a building that was quite unsuited to human use (Ryan, 1996). Our implementations allow the user to explore a range of more plausible forms. However, several issues remain for future work.

Amongst these are methods of making the comparison of alternatives more precise. For example, by providing a virtual 'ruler' or 'tape measure' with which the user can make detailed comparisons between the different views. We are also interested in addressing the questions of collaborative worlds, in which several users might 'meet' to debate different interpretations.

Our example has concentrated on a single building at a single point in time. Our theatre model could be extended to include other buildings in the area (see, for example, <URL http://www.cs.ukc.ac.uk/people/staff/nsr/arch/visrcant/visrcant.html>). Likewise, we may also apply similar methods to allow the user to examine changes through time and thus gain an understanding of the development of the City over an extended period (see also, Johnson & North, *forthcoming*).

6 Conclusion

We are only human and often end up disagreeing or unable to choose between several alternatives. This is often the case in areas such as archaeology where the original data may be sparse and several alternative interpretations may be equally plausible. Publishers require methods that allow them to present these different viewpoints with associated arguments. Users will be better served by systems that enable them to explore and understand the different viewpoints, interpretations and arguments surrounding the original data.

Many existing VR models enable the user to 'pick up' and 'manipulate' virtual portable objects within a fixed virtual environment. Here, we have applied similar techniques to permit the user to manipulate the virtual environment itself. We have presented an example based on the need for effective presentation of the uncertainty inherent in archaeological interpretation, but the approach may also prove valuable in many other fields such as geology, civil engineering and architecture.

Our discussions of four methods of controlling alternative representations have shown that there are benefits in each approach and that none is an overall 'best'. It is a case of using the correct tool for the job. For example, a novice or non-specialist user needs to be guided around and shown the different ideas and varying viewpoints. However, comparison of particular aspects of the model is difficult to achieve in a simple tour.

Publishing archaeology over the internet is here to stay, (see, for example, the journal *Internet Archaeology* - <URL http://intarch.ac.uk>. Considerable use has already been made of graphical and 3D modelling techniques to present virtual museums and 're-constructions' of the past, but many of these suffer from the problem that viewers may believe what they see, simply because it appears on a computer. We believe that it is important to present alternative views of the data and have shown how these can be programmed into a visualization to allow the user to examine and alter parameters and gain a better understanding.

References

Andrews, K. and Pichler, M. 1997 *The VRwave Home Page.* <URL: http://www.iicm.edu/vrwave>

Haber, R. B. and McNabb, D. A. 1990, "Visualization idioms: A conceptual model for scientific visualization systems", in B. Shriver, G. M. Nielson and L. J. Rosenblum (eds.) *Visualization in Scientific Computing*, IEEE Computer Society Press", pp.74-93.

Hartman, J. and Wernecke, J. 1996, *The VRML 2.0 Handbook*, Addison Wesley Developers Press. <URL: http://vrml.sgi.com/handbook/>

Hoque, R. 1997, Java, Javascript and plug-in interaction using client-side LiveConnect, <URL:http://developer.netscape.com/library/technote/index.html?content=javascript/liveconnect/liveconnect_rh.html>

Johnson, I. and North, M. 1997, "Mapping the fourth dimension: a methodology for recording and visualising cultural features through time", in V. Gaffney, M. van Leusen and S. Exxon (eds.), *Computer Applications and Quantitative Methods in Archaeology: CAA1997*, Tempvs Reparatvm, Oxford, forthcoming.

Lea, R., Matsuda, K. and Miyashita, K. 1996, *Java for 3D and VRML worlds*, New Riders.

Marrin, C. 1997, VRML 2.0 External Authoring Interface Reference, <URL: http://vrml.sgi.com/developer/externalapi.html>

Miller, P. and Richards, J. D. 1995, "The good, the bad, and the downright misleading: archaeological adoption of computer visualisation", in J. Huggett and N. Ryan (eds.), *Computer Applications and Quantitative Methods in Archaeology: CAA1994*, Tempvs Reparatvm, Oxford, pp. 249–254.

Moving Worlds, 1997, The Virtual Reality Modeling Language, ISO/IEC DIS 14772-1, <URL: http://vrml.sgi.com/moving-worlds/spec.DIS>

Pesce, M. 1995, *History of Virtual Reality Modeling Language*, <URL: http://webspace.sgi.com/Archive/VRML-history/index.html >

Pichler, M., Orasche, G., Andrews, K., Grossman, E. and McCahill, M. December 1995. Vrweb: a Multi-System VRML Viewer, in *Proceedings of the First Annual Symposium on the Virtual Reality Modelling Language (VRML '95)*.

Reilly, P. 1989, Data Visualisation in Archaeology, *IBM Systems Journal*, vol. 28, no. 4, pp. 569–579.

Roberts, J. C. 1996, "Interactive Hierarchical Data Investigation using Abstractions – the Waltz Visualization Environment", Computing Laboratory, University of Kent, Canterbury, UK, report no 24-96.

Robertson, D., Johnston, W. and Nip, W. December 1995. Virtual Frog Dissection: Interactive 3D Graphics via the Web, in Proceedings of the Second International WWW Conference '94, <URL: http://george.lbl.gov/ITG.hm.pg.docs/dissect/info.html >

Ryan, N. S. 1996, Computer based visualisation of the past: technical 'realism' and historical credibility, in T. Higgins, P. Main and J. Lang (eds.), *Imaging the Past*, British Museum Occasional Paper No. 114, pp. 95–108.

Software, C. 1997. CosmoPlayer. <URL: http://cosmo.sgi.com/ >

Superscape, 1997, *Superscape/Intel Stonehenge visualisation*,
<URL: http://www.connectedpc.com/cpc/ecs/stonehenge>

W3C, 1996, Portable Network Graphics, <URL: http://www.boutell.com/boutell/png/>

Walton, J. and Knight, D. 1997, "Rock'n'Roll: Using VRML 2.0 for Visualization", *Eighth Eurographics Workshop on Visualization in Scientific Computing*.

Wood, J., Brodlie, K. and Wright, H. 1996, "Visualisation over the World Wide Web and its Application to Environmental Data", *Proceedings Visualization '96*, sponsored by the IEEE Computer Society.