Virtual Reality (2006) 9: 215–225
DOI 10.1007/s10055-005-0008-0

ORIGINAL ARTICLE

Kevin Badni

基于网络历史重建的虚拟现实设计技术

Virtual reality design techniques for web-based historical reconstructions

Received: 12 July 2004 / Accepted: 11 October 2005 / Published online: 11 November 2005

Abstract Due to increases in personal computer power and available bandwidths, 3D worlds are becoming increasingly accessible over the Internet, allowing viewers to freely explore virtual constructed worlds. However, even with advances in computing power and bandwidths, creating realistic 3D worlds which are fast downloading and are pleasurable to navigate around requires a number of design techniques to be employed. The aim of this paper is to describe the design techniques used in a redesign of a web-based virtual reality reconstruction of an historical site describing how different techniques can be used to optimise download times yet retain historical realism. The techniques and processes can be used as a guide by any designer to help create lightweight realistic virtual models.

Keywords VRML · Design methodologies · Visualisation techniques · On-line constraints

1 Introduction

Through the medium of television and film, photo-realistic reconstructions of historical scenes produced using computer imagery is common practice. To create these scenes an intensive number of computer-generated images need to be rendered requiring large time-consuming calculations to be undertaken. The resulting productions are a one-way medium being 'passive' as the viewer cannot interact with them. Virtual reality (VR) on the other hand is an interactive medium, described by Hendersen (1991) as allowing users to partake in realities different from their own, with the main purpose being to provide new experiences that can enrich their normal lives.

As VR is an interactive medium it must by its nature occur in 'real-time'. Real-time as the words describe, is the ability to render a scene immediately rather than allowing the computer to render the scene off-line. The rendering technique of ray tracing used in films and for photo-realistic rendering is known for its ability to create high-quality images, however, it is also known for its long rendering times due to high-computational costs. These costs are directly related to the requirement of the computer to calculate the traversing light rays across the scene, intersecting with each geometric object and applying the correct shading to the visible objects. Due to this high-computational cost ray tracing is perceived almost exclusively as an off-line technique (Waltery et al. 1999). VR models do not have the luxury of using off-line rendering, due to the interactive requirements a computer would need to be able to calculate a vast amount of data in real time to create truly realistic representations.

Graphics pioneer Alvy Ray Smith once said that 'reality' was "80 million polygons per second", meaning that the computer would have to be able to display that many polygons on the screen in real time to achieve fully realistic images (Bolter and Gromala 2003). To put this into perspective a typical high-end PC can render 7 million polygons/s whilst the Playstation 2 which is a dedicated polygon rendering computer can render up to 20 million polygons/s. This comparison can be seen in Table 1.

Traditionally, VR has been based upon a structure of using high-end computers capable of manipulating large numbers of polygons in real time such as the SGI Onyx 2. The performance of a 3D rendering engine running on a computer can be measuring by the number of images it can generate on-screen per second (frames per second—FPS). The higher the FPS the smoother the perception of on screen motion. When a model is being created adding complexity increases the levels of realism, however, increasing the complexity has a negative affect

K. Badni
Department of Design and Technology, Loughborough University, Leicestershire, LE11 3TU, UK
E-mail: k.s.badni@lboro.ac.uk
Tel.: +44-1509-222980
Fax: +44-1509-223999

Table 1 Performance comparison between Onyx2, PC and PlayStation 2

	SGI Onyx 2 with infinite reality graphics (industrial VR engine)	High-end PC with asus 6,800 graphics card	PlayStation 2
CPU	250 MHz+	1 GHz	300 MHz
RAM	up to 16 gigabytes	Gigabyte +	32 MB
Max Polygon Rate	10 million per pipeline/s	7 million/s	20 million/s

on the ability of the computer to render frames in real-time. The quandary faced by VR designers is whether to add more complexity resulting in better realism or to aim for a high FPS rate. Yuan (1999) leans towards FPS stating that responses to real-time actions and the perception of natural movement are as important if not more important than realistic imagery to the users. However, aiming at a purely high FPS rate could result in a scene with a rather simple or naïve appearance (see Fig. 1). The VR designer must therefore try to reach a balance creating realism whilst maintaining an acceptable FPS rate.

This paper looks at a number of different design techniques which can be used to address the problem of how to reduce the polygon count in a VR world yet still retain a high level of realism with a good FPS rate. The practical use of these techniques is demonstrated in a publicly accessible VR world which can act as a guide for anyone wishing to create a realistic lightweight web-based VR models.

2 Web-based VR

Exposure to VR environments used to be restricted to expensive industrial projects or academic computer laboratories due to the initial high-capital outlay and continuing maintenance costs. Fortunately, in May 1994 at the first International World Wide Web Conference in Geneva, Switzerland the VRML mark-up language was introduced by two researchers, Tony Parisi and Mark Pesce. They presented a session on virtual reality, and described their efforts in developing a virtual reality interface for the Web. With the creation of this language a number of dedicated web browsers have been produced which are used to display and control virtual worlds using the VRML mark-up language over the Internet. These browsers are cross platform compatible and interpret the VRML ASCII files (using the file suffix. wrl) displaying a VR model within the browser. These models comprise a number of objects creating a 'world' or can simply be single objects. Davis (1996) compares the functions of a VRML browser to those of a standard HTML browser. The standard web browser interprets HTML code allowing the user to see arranged text and images. VRML browsers work in the same fashion interpreting the .wrl files allowing the user to view arranged 3D models instead.

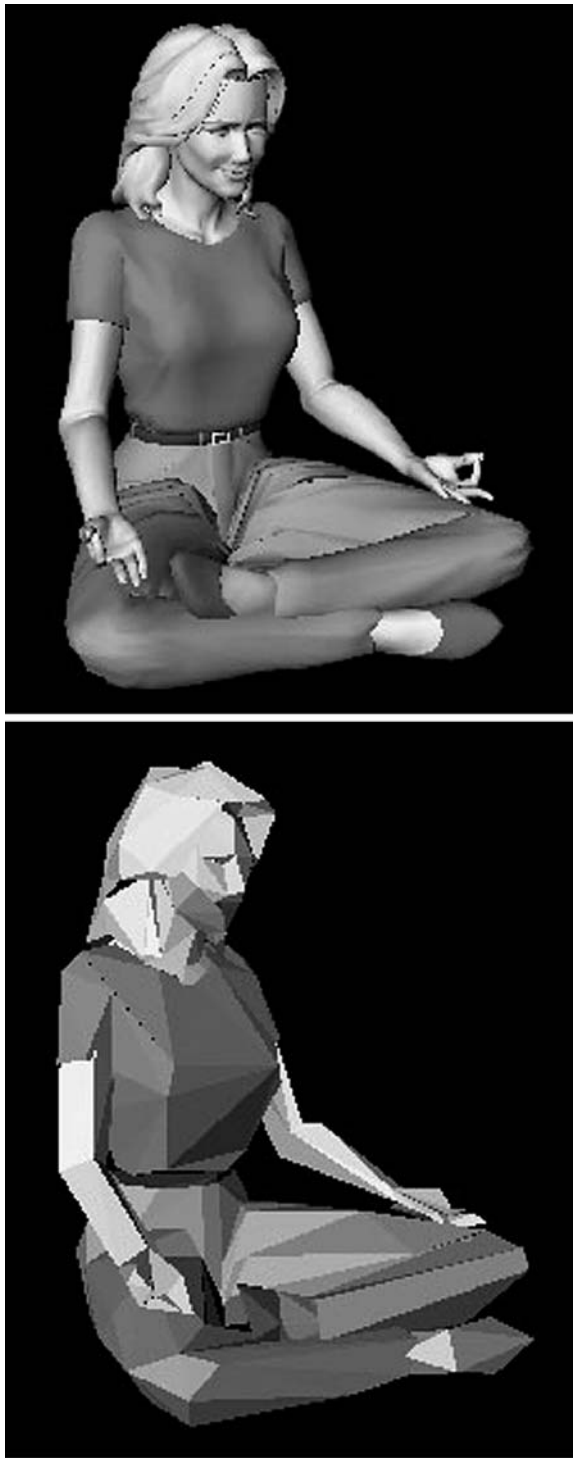

Fig. 1 HiRes model at 33254 polygons compared to LowRes model at 620 polygons

Unlike C++, VRML is not a general purpose programming language, neither is it a script-based language like JavaScript or a page specification language like HTML. VRML is a scene description language based on a subset of Open Inventor from Silicon Graphics that describes the geometry and the internal behaviour of a 3D scene. This definition of a set of

objects is known as nodes, which are arranged in hierarchies called scene graphs. These scene graphs have a top–down arrangement in which nodes that are written earlier in a scene affect later ones. These nodes can be divided by the use of separator nodes. These can be used for example to colour one part of the scene, but prevent the following nodes from being coloured. The techniques described in this paper will manipulate a large variety of these nodes (Fig. 2).

2.1 Performance issues

The case study used in this paper required an increase in the number of features compared to the existing VRML world. Even with high-speed computers VRML designers must use a number of methods to maintain the high-frame rate required in a VRML application. To compound the restrictions faced by VRML designers, web-based models are displayed not on arrays of high-end workstations but on personal home computers, which cannot be guaranteed to have fast processors and specialised graphics cards. The VRML model files must therefore contain the minimum number of polygons as possible so keeping the file size as small as possible. The smaller the file the faster the computer can render polygons and hence the higher the FPS rate. Hartman and Wernecke (1996) recommend that the designer should aim at achieving a minimum frame rate of 10 FPS, which is the frame rate required to simulate visual continuity of moving through the real world.

FPS is affected by the processor speed, graphics card and memory available on the clients PC. As this is unknown, the VRML designer cannot presume that they have the latest graphics card or 1 GB of RAM, and must therefore design the 3D scene to be as computationally light as possible yet still maintain a high level of model realism.

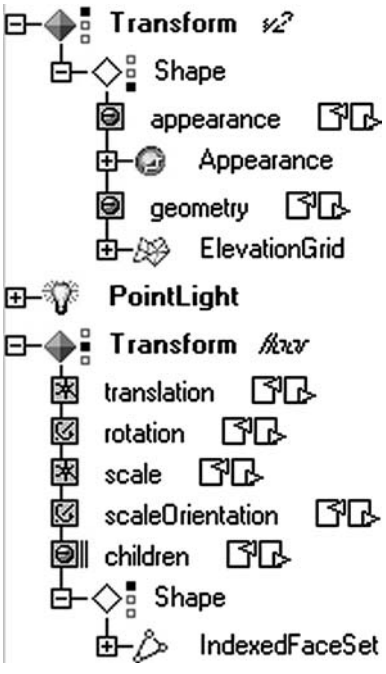

Fig. 2 Graphic representation of VRML nodes

3 Comparative models

Within this paper two VRML models are used to compare design approaches, aesthetics and performance results. The two comparative models were both created for the BBC History Web Site. The BBC History web site has a multimedia section that presents the viewer with a number of interactive VRML models (Fig. 3). The model referred to in this paper can be viewed at this site (BBC History Web Site 2005).

The two computer models created were both of a section of a World War One Trench. The first model was produced in 1999 as one of a number virtual reconstructions produced for the BBCs History website by PERA based in Melton Mowbray, United Kingdom. The second revamped model was created by the author in 2001 to accompany the BBC2 programme 'The Trench' which recreated the experience of the 10th Battalion of the East Yorkshire Regiment in the Autumn of 1916 on the Western Front. When the Trench programme was being considered by the BBC they revisited the original VRML Trench model. Even though it showed the Trench's tactical layout and construction and allowed the viewer to explore the Trench paths, it was felt that the model lacked any realism. The BBC required a new more realistic VRML model. The requirements were for the new model to still effectively show the tactical layout of the Trench and allow the visitors to walk through to the front line, but it must also appear less clinical and include some WW1 artefacts. This was all to be achieved without a dramatic increase in the .wrl file size. The basic modelling and limited content helped to keep the original Trench model size down to only 85 Kb but its compromise was that of realism.

Size is important, a 1 Mb VRML model would only take a few seconds to download on a broadband connection, but via a 14.4 Kb domestic modem it could take up to 10 min. The original Trench model was 85 Kb so took approximately four and a half minutes on a 28.8 Kb modem to download. The new Trench model (NTM) was required to be of a similar size yet must be visually more realistic.

When researching the NTM the author worked closely with an expert historian and used photographs of the Trenches and WW1 artefacts to try and maintain as much accuracy as possible. The issue of inaccurate virtual models available on the Internet is discussed by Kantner (2000) who states that web-based virtual reconstructions have a great potential to be misleading in their presentation and in their interpretation. This is due to the wide variety of backgrounds of both

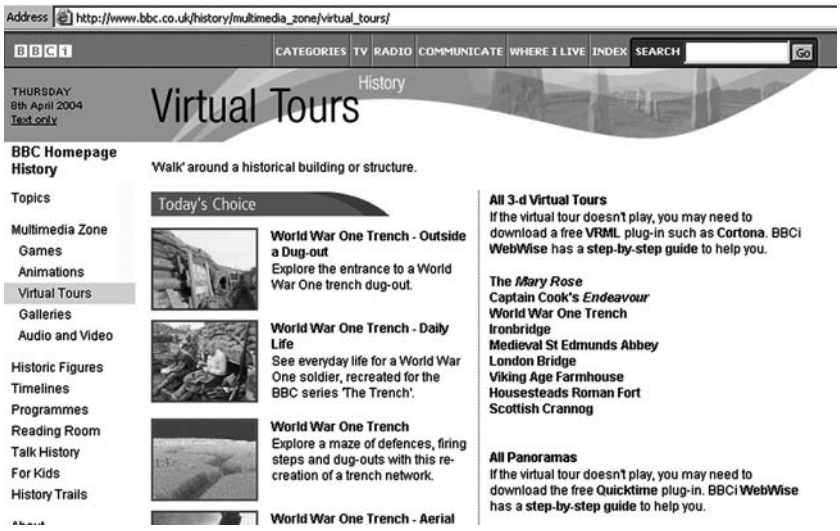

Fig. 3 BBC History Web Site—Virtual Reality tours

modellers and viewers that use the Internet. The VRML design is always a compromise between realism and what can be displayed considering computer software, hardware and bandwidth. As stated by Davis (1996) complete imitation is impossible as it is inevitable that some process of subtraction and modification of representation is always required. To maintain a small file size a number of techniques were deployed by the author to make sure the model was lightweight yet create a more realistic experience, the following sections outline these techniques.

4 Creating VRML worlds

The following design techniques were specifically used by the author to fulfil the BBCs requirements for a more realistic Trench model whilst maintaining a computationally light interactive 3D model. Other design techniques can be used to increase download and rendering speeds. These along with specific software modelling methods and VRML codes are not explained as these are better illustrated with dedicated user manuals (Hartman and Wernecke 1996). However, the techniques used by the author with their associated examples should prove useful to anyone working on VRML projects with the aim of attaining the same goals of producing visually realistic yet computationally light VRML worlds.

There are various ways in which VRML files can be created. The most basic way is to use a text editor, typing in the VRML code directly and then loading the final file into a browser for viewing. This requires a very good understanding of syntax and is not practical if the world being designed is very complex and contains non-primitive shapes.

There are some tools available which are specifically aimed at designing VR worlds, such as Sense8s WorldToolKit (Sense8 2005). The WorldToolKit software has been programmed with the aim of providing all the functionality a developer would need to create complex applications. It is an object-oriented library written in C with more than 1,000 high-level functions for configuring, interacting with, and controlling real-time simulations.

Dedicated 3D modelling packages are also available which allow the VRML designer to create the world graphically on-screen and which output a VRML-compliant file. For example, both 3D Studio Max and Pro-Engineer have options to save 3D model files in the VRML format.

Each VRML designer will have their own working preference. In the case of the author a 3D graphical package was used to create the main model and a simple text editor was utilised to adjust and optimise the final model.

5 VR optimisation techniques used in the NTM

General modelling strategies and VR optimisation techniques are covered in a number of dedicated books (Hartman and Wernecke 1996) on the subject. The following section describes the VR optimisation techniques specifically applied by the author in the NTM.

5.1 Texturing

Textures give objects in the VRML scene a realistic look whilst maintaining a low-polygon count. A texture is a 2D image, usually JPEG or GIF format, which wraps around the selected 3D object in the VRML scene. The 3D object surface then takes on the colour, tone and patterns associated with the 2D images. Texture mapping can be compared to wallpapering a real object. The use of texturing allows the designer to minimise the number of polygons required to represent an object. In

the NTM for example a number of ammunition boxes were used. These were simple cubes but with the use of textures the blank cubes took on the appearance of panelled boxes. Figure 4 below illustrates this approach.

5.2 Billboards

Another form of texturing is as a Billboard. Billboards are single flat layers that rotate around a specified axis and always face the viewer. For example, the designer can import a 2D image of a tree and specify that it rotate around its *Y*-axis (up). Since the tree image is always facing the viewer, they will not be able to see that it is simply a flat image. Billboards can also be designed to rotate around the *X* or *Z*-axis or be screen-aligned. Billboards are useful for optimising the scene, since the VR designer can use 2D objects to simulate 3D objects and thereby save large numbers of polygons. In the NTM a number of British Lee-Enfield 0.303-inch rifles were used in the scene. Rather than modelling each individual part of the rifle, two polygons were created forming a rectangle and a 2D image of a rifle was placed on the rectangle representing the rifle. These Billboards rotated around the *Y*-axis as the viewer walked past them. A fully modelled Lee-Enfield 0.303-inch rifle would have contained many polygons and be very processor intensive, whereas a Billboard only has two polygons and a single texture. Figure 5 illustrates how the Billboards developed.

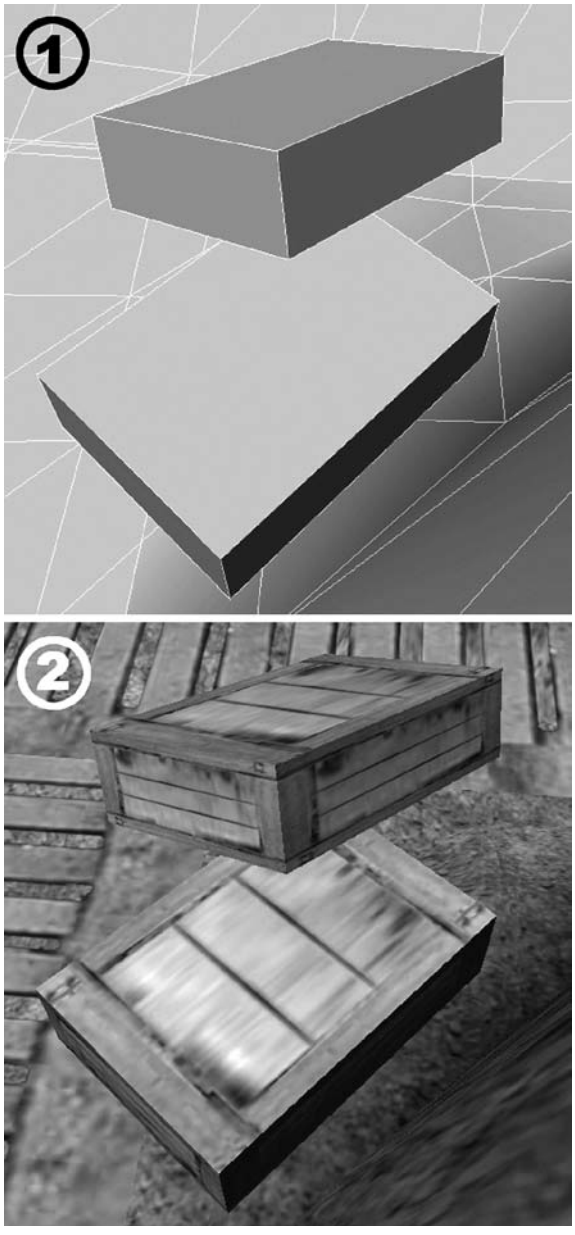

Fig. 4 Texturing. 1—untextured polygons, 2—textured

5.3 Lighting

VRML does not support 'dynamic' lighting, due to the real-time nature of the applications the calculations required to produce correct lighting and shadows is simply too processor intensive. However, VRML does support a number of different lights:

- Directional lights are the simplest type of light. They have a direction but no location.
- Point lights radiate light equally in all directions from a given location.
- Spotlights illuminate light along a primary direction from a given location. These spotlights are analogous to spotlights used in theatres, which produce a cone of light that diverges from the light's location.

Spotlights by their nature take longer to compute than point lights, which take longer to compute than directional lights. Therefore, directional lights should be substituted for point lights and spotlights where possible to assist with the rendering speed. The VRML designer can also specify a headlight. As the name suggests this is a light which acts as a headlight attached to the viewer as they navigate the VRML scene.

Within the NTM two directional lights were used to fully illuminate the scene. A headlight node was not used to avoid saturating the NTM model with light, (see Fig. 6) this also assisted in reducing the processor load on the viewer's PC.

5.4 Distance level of detail

To reduce the number of polygons displayed in the scene a VRML designer can specify the browser to use the level of detail (LOD) node. The distance LOD node specifies the distance at which objects are displayed relative to the viewer within the scene. This is a particularly useful optimisation technique as increases in object detail are only introduced into the scene when required, thus keeping the polygon count as low as

Fig. 5 Billboards. 1. Flat plane no texture. 2. Transparent Gif texture on Billboard. 3. Billboard rotated around the *Y*-axis being viewed from reverse perspective

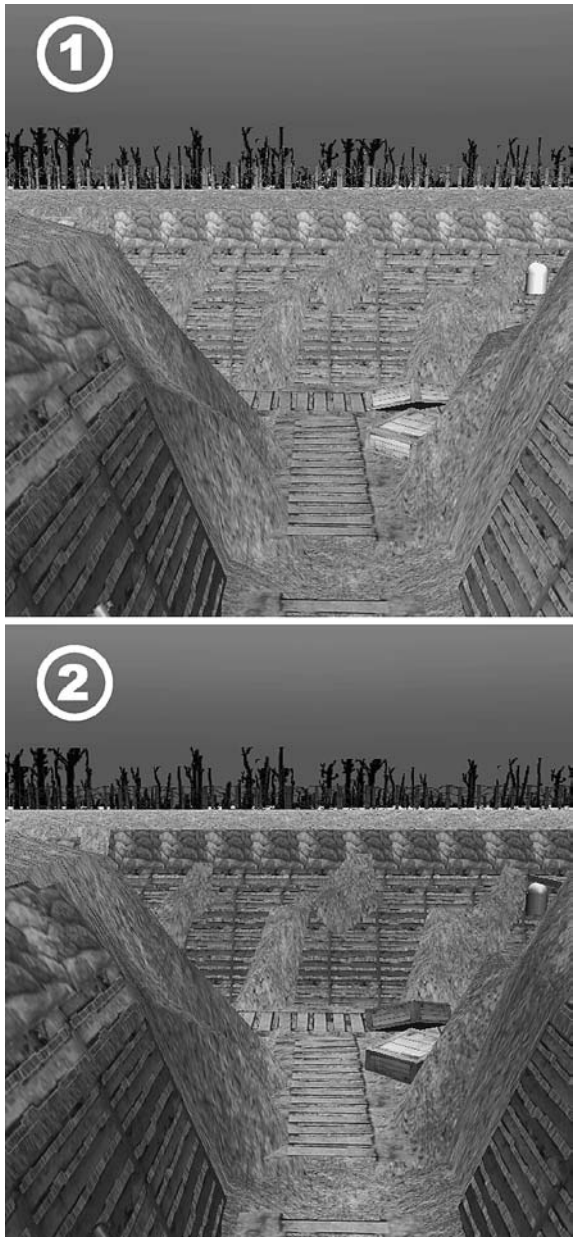

Fig. 6 Lighting. 1. Saturated with headlight node. 2. Two directional lights only

possible for the maximum time. Using LOD node does have a down side, navigation may become rather stop–start, the viewer will travel smoothly whilst a model is in RAM and then there maybe a pause whilst the next LOD is loaded. However, without the use of the LOD node navigation may be slow all the time.

Within the NTM LOD nodes were used not only for complete sections of the Trench but also for the artefacts within the Trench. From a distance the ammunition dump was a single polygon of colour, as the viewer moved closer simple coloured boxes were made visible. As the viewer gained a closer proximity the boxes became fully textured. The LOD node was also used to

remove the tactical overlaid map so it only appeared when the user first entered the VRML world and disappeared as the viewer moved closer to the Trench. Figure 7 below illustrates a three level LOD used in the NTM.

5.5 Cloning

Cloning, or instancing, is an optimising technique used extensively for the creation of artefacts within the NTM. If the same object appears multiple times in a scene, cloning of the object is used instead of copying, as cloning references the original object's geometry instead of creating a whole new object. Cloning produces an identical copy of the object which shares the original object's geometry and appearance information. Whereas, copying produces an identical copy of the object that has its own geometry and appearance information. Clones share the same appearance but can be resized and placed independently. Through cloning each model is downloaded into the PCs RAM only once and used repeatedly, therefore reducing the download time compared to downloading individual copies.

5.6 In-lines

Another download optimisation technique is to use In-Lines. In-Lines are self-contained 3D model files with associated textures and optional lighting and sound attributes. Similar to cloning these In-Lines instances are referenced by the main VRML scene and are used wherever possible to avoid unnecessary downloading. When designing the NTM the entire model was divided into modular sections so individual predefined components could be imported as in-lines. In this way the NTM was assembled like a large jigsaw puzzle, each component being downloaded only once and then used repeatedly. Figure 8 illustrates the construction of the Trench walkways by combining individual In-Lines.

5.7 Elevation grids

One of the main criticisms of the original Trench model was the clinical look of the Trench walls. These walls were perfectly straight in the original model being computationally efficient as each wall panel consisted of only two polygons, unfortunately, this did not represent the actual walls which were hand dug out of fields. Rather than graphically modelling new walls irregular terrain in the NTM was created by utilising the elevation grids node.

In VRML the elevation grid node specifies a computationally efficient rectangular grid. At each intersection within this grid, values are assigned which creates

Fig. 7 Level of detail. 1. At a distance—single coloured polygonal plane. 2. Mid-distance—coloured boxes. 3. Close up—fully textured boxes

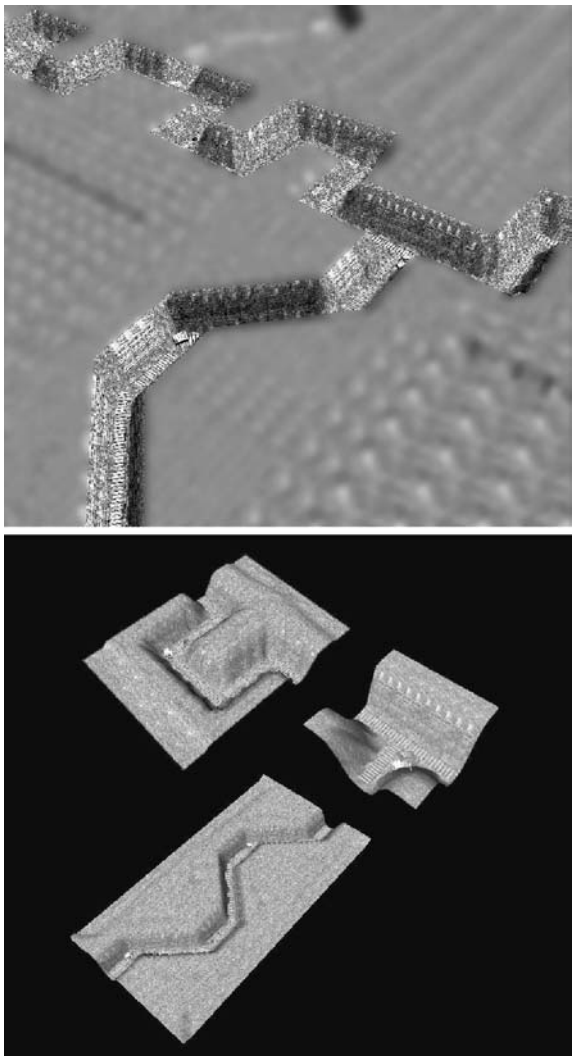

Fig. 8 In-lines. The NTM was constructed on a modular basis using multiple In-lines to reduce the download time

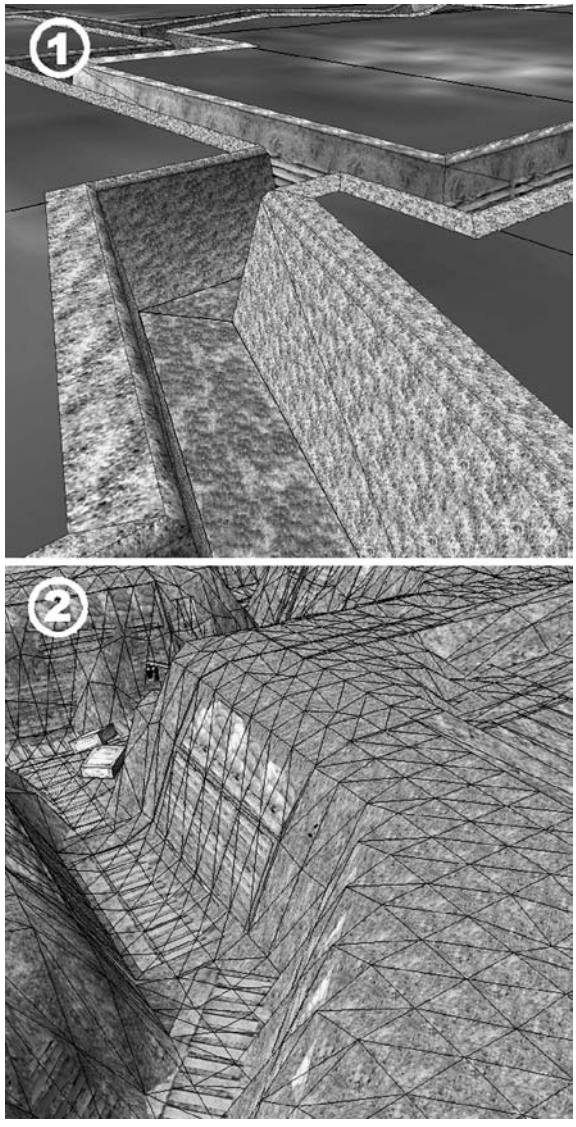

Fig. 9 Elevation grids. Original Trench model was too straight and clinical. NTM used elevation grids to add realism

specific heights above each intersection, so generating a mathematical elevation grid. The grid can be scaled and contain any number of intersections. Once the elevation node has been written the VRML browser then interpolates surfaces between the given vertices. Since, it is unlikely that the terrain surfaces will be planar, each quadrilateral is broken up by the browser into a pair of triangles (see Fig. 9).

5.8 Fog and sound

To add some atmospheric realism, fog and sound nodes were introduced in to the NTM.

The fog node allows the VRML designer to simulate the effects of fog by making objects further away appear less visible. The fog node cannot however create swirling eddies as this would be too computationally intensive. Instead the fog node simply blends the fog colour with everything that is displayed in the browser so simulating a reduced depth of field. Using fog improves rendering performance of the browser as any object beyond the visibility range set within the fog node does not get rendered (Fig. 10).

Within the NTM sound nodes were used to add atmospheric background noises. These sound nodes were placed within the NTM scene using their location fields in the same manor as other 3D objects. When the viewer is at or very close to their location the sound volume is heard at the maximum recorded volume. The VRML designer has more control over the sound node than just position and volume. VRML allows the designer to control where the sound is heard by using two ellipsoids, each of which has the location as a focus. Generally, a smaller ellipsoids is completely contained within a larger other. The smaller ellipsoid defines the region where the sound is heard at maximum recorded

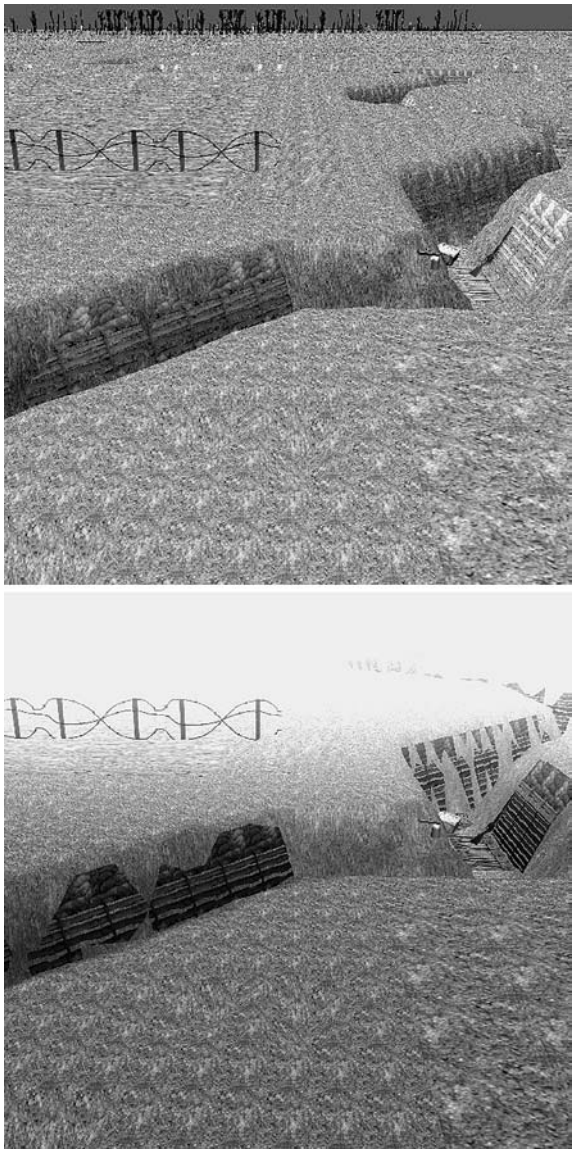

Fig. 10 Fog. NTM without and with fog node

volume, moving within this ellipsoid makes no difference to the sound's volume. The region between the small inner ellipsoid and the larger outer is where sound attenuation takes place. The sound's volume decreases as a function of distance from the inner ellipsoid to the larger outer ellipsoids surface. Once outside of the larger ellipsoid the sound cannot be heard (Fig. 11 illustrates this). Within the NTM the inner ellipsoids were placed at the front line so full volume is heard at the front line. The outer ellipsoids were orientated towards the support trenches so the sound grew louder as the viewer approached the front line. The sounds for the NTM came from the BBCs sound archive. These were high-quality files which resulted in very large file sizes. To maintain a quick download time the sounds were reduced by re-sampling them at a lower rate and changing the stereo sound to mono.

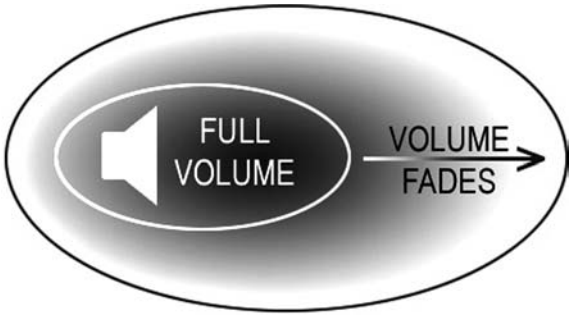

Fig. 11 Sound ellipsoids

5.9 Arranging objects spatially

The Cortona VRML browser which the BBC recommends the client to download, uses a view culling process to decide which objects in the scene should be rendered and which objects should be ignored. Each model within the VRML scene has a bounding box which is the smallest rectangular box that can enclose the object. The browser uses this bounding box combined with the viewing camera's field of view to determine which objects should be rendered. The grouping of objects within a VRML list affects what the browser will render. If the bounding box in the field of view is grouped with another 3D object outside of the browsers field of view it will also try to render both groups. However, if the other 3D object is not grouped with the bounding box in view, the other 3D object will not need to be rendered and consequently the browser rendering speed and FPS will increase. For example in the NTM sections of the communications trenches were not grouped with the reserve trenches as these would not be seen from within the aforementioned Trenches. At a lower level the ammunition dumps were not grouped all together they were individually grouped only with the part of the Trench that they sat in, so helping to maintain a good FPS rate.

5.10 ASCII adjustments

As a VRML file is an ASCII file, removing any unnecessary spaces will reduce the overall file size. In the NTM superfluous new lines, spaces and leading tabs, were removed using a simple text editor. The floating point numbers used were also reduced to a decimal place of three with little noticeable affect on the final model. The final .wrl files were compressed using the patent free gzip utility (Gzip 2005) as the Cortona browser automatically decompresses this type of file. As the files are compressed they can be downloaded more quickly. Before using the gzip utility

Table 2 Comparison of case studies

	Original Trench model	Revamped Trench model
Model statistics		
Polygons	2852	16176
Model size	85 Kb	42 Kb
Sound file size	0 Kb	27 Kb
Texture size	70 Kb	187 Kb
Total size	155 Kb	256 Kb
Download time comparison with identical Internet set-up		
14.4 Kb modem	1 min 28 s	2 min 25 s
28.8 Kb modem	0 min 44 s	1 min 12 s
56 Kb modem	0 min 22 s	0 min 37 s
ISDN (64 Kb)	0 min 19 s	0 min 32 s
ISDN/DSL (128 Kb)	0 min 9 s	0 min 15 s
VRML nodes		
Billboards		√
Clones		√
Directional light	√	√
Elevation grids		√
Fog		√
FPS	avg 9.02	avg 5.65
Headlight	√	
In-lines		√
LOD		√
Point light	√	
Sounds		√
Spotlight		
Textures	√	√

the NTM was 358 Kb afterwards it was reduced to 256 Kb saving almost an entire minute in download time when using a 14.4 Kb modem.

6 Technical comparison of the two trench VRML models

In order to examine the key differences between the two Trench models it is necessary to consider the key technical difficulties. Table 2 compares the two VRML models from a technical perspective. As can be observed the number of polygons used in the NTM was five and a half times more than that of the original Trench model. However, due to the modelling and optimisation strategy implemented the final file size was only one and a half time larger yet contained many more VRML nodes than the original model.

Comparing the visual differences (Fig. 12) highlights how these technical differences are transferred into very different VRML models.

7 Conclusions

3D virtual worlds by their nature are typically large in size and so typically suffer from long download times and performance latency issues. Research by Zari et al (2001) suggests that the amount of time required for a website to download is a significant factor in discerning the satisfaction of the user and the ultimately

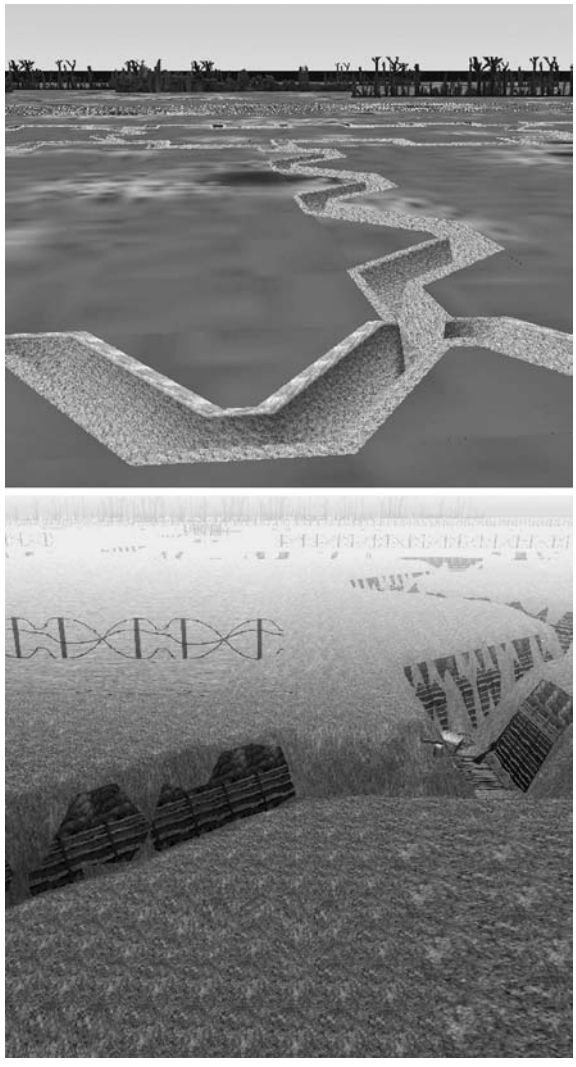

Fig. 12 Original Trench model and new Trench model

the success of a website. Large file sizes also have a negative effect on display latency which as described by Conner and Holden (1997) can induce a number of unpleasant consequences for the user. These include motion sickness and a feeling of lack of control due to the user experiencing unexpected delays when making inputs.

The techniques described in this paper to constrain VRML file sizes and maintain fast rendering speeds are essential to minimise these negative issues. However, simply following these techniques does not guarantee that the resulting VRML worlds will be of a high-visual quality or that the interactivity will make the world an enjoyable area to explore as aesthetic judgements and user guidance are also important aspects of any 3D world. This has been shown in the example of the case study within this paper. The original Trench model which had a smaller file size and higher FPS than the NTM was replaced by the NTM. It was judged by the BBC to be superior in its

historical realism with its use of sound, visual aesthetics and interactive guidance and as such took the place of their existing Trench model.

With continuing increases in computer power it could be presumed that viewing complicated VRML worlds will become quicker and smoother as the processing power and bandwidth access increases. However, the requirement to keep models to manageable size will continue as network traffic is also likely to increase. It is unlikely that there will ever be a situation where 3D scenes actually move too fluidly in the browser, or the user's actions are responded to too quickly. Therefore, the techniques discussed in this paper should remain valid for the foreseeable future.

References

BBC History Web Site (2005) World War One Trench 3-D Virtual Tour. [Online]. Available: http://www.bbc.co.uk/history/war/wwone/launch_vr_trench.shtml [2005, May 13]

Bolter JD, Gromala D (2003) Windows and mirrors, interaction design, digital art and the myth of transparency. MIT Press, Cambridge Massachusetts, pp 40

Conner B, Holden L (1997) Providing a low latency user experience in a high-latency application, Proceedings ACM Interactive 3D graphics '97, April 27–30, 1997, Providence, Rhode Island, United States, pp 45

Davis SB (1996) The DESIGN of virtual environments with particular reference to VRML. Report for the advisory group on computer graphics, Middlesex University, 4:61–65

Gzip (2005) Patent free compression utility. [Online]. Available: http://www.gzip.org/ [2005, May 13]

Hartman J, Wernecke J (1996) The VRML2.0 Handbook, building moving worlds on the Web. Silicon Graphics Inc., ISBN 0-201-47944-3

Henderson J (1991) Designing realities: interactive media, virtual realities, and cyberspace. In: Helsel S, Roth J (eds) Virtual reality, theory, practice and promise. Meckler, pp 66–72

Kantner J (2000) Realism vs reality: creating virtual reconstructions of prehistoric architecture. Virtual reality in archaeology, Archeopress, Oxford, pp 1–5

Sense8 (2005) Graphic based VR modelling software. [Online]. Available: http://sense8.sierraweb.net/ [2005, May 13]

Waltery B, Drettakisy G, Parker S (1999) Interactive rendering using the render cache, rendering techniques '99 (Proceedings of the 10th Eurographics Workshop on Rendering, Springer-Verlag/Wien 10:235–246

Yuan P (1999) Benchmarking mesh simplification algorithms in real-time rendering applications. Presented at western computer graphics symposium SKIGRAPH'99. pp 1

Zari M, Saiedian H, Naeem M (2001) Understanding and reducing web delays. Computer 34 12(Dec.):30–37