Distributed Virtual Environments for Urban Modelling

D. Harvey, D.B. Arnold and A.M. Day

School of Information Systems, The University of East Anglia,
University Plain, Norwich, Norfolk, NR4 7TJ, England

dah/dba/amd@sys.uea.ac.uk

Abstract

We describe many of the considerations required when designing and developing a distributed virtual environment for urban modelling, paying particular attention to its application to virtual heritage. We also describe the possibility of using a specially defined set of scenery and building primitives to reduce the amount of data that needs to be transmitted to recreate the virtual urban environment on the observer's computer.

Keywords: distributed virtual environment, urban modelling, urban model primitives, internetworking, virtual heritage

1. Introduction

The technology and research to deliver usable distributed virtual environments is advancing rapidly through the work of many independent research teams. Unlike a localised virtual environment, a distributed virtual environment offers unique challenges and problems, but also allows a greater flexibility and opens many new possibilities in the area of multi-user environments.

When modelling large urban areas, such as those encountered with some Virtual Heritage projects, the potential applications of distributed environments are immense.

Many possibilities exist for virtual urban environments (distributed or otherwise) in virtual heritage, ranging from non-interactive walk- or fly-throughs of urban sites that are of particular historic interest, to fully interactive "role playing" experiences for entertaining for educating people about life in times past.

There are two main reasons why a project would want to adopt a distributed virtual environment rather than a localised one. Firstly, the volume of model data may be too large for a single machine to store and process. Secondly, this volume of data may be required to be viewed by multiple people at the same time, while being centrally maintained. Obviously, it is inefficient to store and update the model on each observer's computer because there is a large amount of unnecessary data duplication.

This paper outlines some of the ideas and technologies that can be used in creating a distributed virtual environment for displaying historical urban model.

2. Distributed VE for Urban Modelling

When designing a distributed virtual environment, one may fall into the trap of trying to make it too flexible or generic. Attempting to construct the environment from just triangles or quadrilaterals may mean that in a large number of instances, bandwidth is being wasted by transmitting models that could probably be sent as a primitive or separate object defined only once. Consider a simple street, with lamp-posts on either side of it. Each lamp-post is effectively identical, and yet if the environment is stored as just polygons, each one is transmitted separately.

The DEF and USE commands included in VRML[1] solve this problem of object duplication partially, although Bourdakis et al[2] describe that, even with these, VRML as it stands is not wholly applicable to producing large urban models. The VRTP[3] and VRML Streaming[4] Working Groups are investigating more efficient methods of transmitting VRML over the Internet. However, because VRML is designed for displaying generic polygonal structures, the underlying polygons must still be transmitted at some point, even if it is only once.

Transmission (and rendering) times can be reduced by using level of detail[5] techniques, where simpler representations with fewer polygons are used for distant objects (or fast moving objects). However,

Published by the Eurographics Association ISSN 1017-4656

the observer is normally free to move closer to the low-detailed representation, and it must therefore be updated by either transmitting a newer representation of the model, or making use of incremental encoding[6] or similar methods. It is an inefficient use of bandwidth to transmit more than one copy of a part of the model at different levels of detail (incremental encoding eliminates this at the cost of data complexity), although this could be justified for very distant objects.

Of course, polygonal information need not be the only information transmitted. The use of impostors[78], where distant, complex, geometry is replaced by a transparent “billboard” texture, can help to decrease rendering times significantly. However, these 2D or 3D representations of the geometry are not efficient for transmission if they are large. Impostors would then be best used for a distributed virtual environment to represent geometry that would be rendered to the sub-pixel level on the client’s machine - especially buildings close to the horizon.

If the data set is within fixed boundaries, specialised primitives can be defined to decrease data transmission requirements dramatically. With a historical urban model, we estimate that approximately eighty percent of the model will be relatively uninteresting buildings, such as plain terraced houses. These parts of the model are certainly required to add realism to the model, but are of no direct interest to the casual observer. For example, a virtual environment containing just a cathedral - however accurately modelled - would not feel fully immersive because it would lack the surrounding buildings that make up a city.

Our research into traditional English buildings[9] has shown that in most cases, a typical historical building can be defined by very few parameters. While the resulting building will not be a completely accurate representation of the physical building, it will be a fairly close approximation and for these buildings it may not matter.

When this is considered in the context of a distributed virtual urban environment, significant portions of the network bandwidth could be saved by transmitting navigation commands within a known data set defining various traditional building styles. An example might be a building defined thus: Type: Victorian terraced, Style: Gothic, Material: Red brick, Floors: Two, Roof: Gabled, Windows: Double-sash, Front door: Right-aligned. Using a system such as this, an entire building could be transmitted in significantly fewer bytes than would be required to transmit the polygonal information - only these parameters and its position, scale and orientation would be necessary. In addition to this, the building can be rendered to any level of detail (even the interior of the building can be generated, if needed) without requiring any additional use of network bandwidth. When the need arises to transmit “feature presentations” (such as the cathedral, in the example above), the problem has now been simplified because almost all of the network bandwidth is still available and has not been clogged by transmitting relatively uninteresting scenery.

Because less interesting buildings can be described using a finite set of parameters, there is obviously a finite number of buildings that can be generated. In order to give the illusion of a wider range of buildings, various random factors can be introduced, such as minor displacement of windows, doors or chimneys, etc. It is often the subtle things - those that would not be apparent to a casual observer - that help to enhance the illusion of realism a virtual environment creates.

These subtle changes in the various buildings must obviously be generated on the client side of the connection, to allow the encoded buildings to be transmitted, and minimal bandwidth is used. This does mean, however, that two important factors are introduced.

Firstly, the generation of these buildings must now be done at run time, rather than simply referencing an existing primitive. This has the potential to be fairly slow, especially if the interior of the building is to be generated with certain random elements. The generation time can be reduced by moving back to a set of specialised primitives for simpler elements - a window frame instead of the entire front of the house, for example.

Secondly, at the lowest level, the buildings’ random elements would be generated using some form of random number generator. If the generated model is to be identical each session, for all users, the seed value for the random number generation must be fixed by the server, or each building needs to be stored (and transmitted) with its own pre-calculated randomisation parameters.

Another important factor in creating a virtual urban environment is the ground. While it is not one of the first things to be considered in a project, it is still a key factor in generating a realistic environment. Obviously, the simplest method of representing the ground is to use a flat plane. A more sophisticated approach would be to represent the ground as a series of height points, which are then connected by triangles or quadrilaterals before rendering. This representation lends itself well to lossy compression (such as JPEG[10]), especially when one considers the height points can be represented by a grey scale image where the luminosity of the individual pixels is relative to the height of the point. The rendering time for this can be reduced by adding specialised level of detail to the resulting structure[11].

The problem of creating a distributed urban environment can be simplified if the environment is considered - in part, at least - to be a 2½D problem. In ninety percent of cases, the urban environment can be considered to have no concave areas at all: a building can be considered to be an extra entity within the world and can be "superimposed" on top of the landscape, so any concave parts of the building are of no concern. With this in mind, the world can be stored in a simple binary or quad tree, rather than a more complicated octree structure.

For inter-user interaction, two main options are available – peer to peer, where the user's computers communicate amongst themselves, or client/server where all inter-user communication goes through a centralised server. In the context of a distributed virtual environment, this communication could just be the appearance and movement information of the user's avatar within the environment, or it could contain text or multimedia streams. Alternatively, a combination of both client/server and peer to peer may produce the most efficient results. The server need not know about inter-user communication, but it would certainly already know about user's navigation commands, so the client need only transmit the appearance of the user's avatar to the server when it first connects and this would be propagated to other clients when required.

The current trend in other areas of internetworked multimedia development, such as audio or video, is to move from traditional "download and play" formats to far more internet-tuned formats, where data transmission delays can cause problems with real-time media. The audio and video streaming formats from RealNetworks[12] are extremely popular (especially for on-line music shops), and these play in real time, playing silence or lower quality audio when the incoming audio stream arrives too late (due to network delays), rather than pausing the sound and waiting. It is possible that a similar approach could be employed for a distributed virtual environment - the incoming data stream is time-stamped and automatically discarded if the observer has already moved on from that position. However, this would obviously lead to potential gaps appearing in the world during slow transmission periods, which would potentially destroy any feeling of realism or immersion the user may have.

Most large scale distributed virtual environments will most likely use the Internet as the underlying transport medium – especially if the project has some form of commercial sponsorship or goals. The speed of the Internet can be a problem for real time applications - hence the need for time-stamped data in real time audio and video streaming. With users accessing the Internet with many different connection speeds, from slow 14Kbps modems to fast T1 or LAN connections. Obviously, the amount of data that can be transmitted during a real-time walkthough is constrained by the available network speed.

Source	*Throughput*	*Raw Triangles*
56K Modem	12K/second	340
Fast LAN	992K/second	28200

The table above shows the result of a simple test to measure the typical maximum throughput for two different connection speeds. The test program transmitted 100K of data as fast as it could, and measured the time taken to receive. The "raw triangles" column shows roughly how many triangles (with 32 bit precision) could be transmitted per second, in a brute force system. A more realistic application would almost certainly not achieve this throughput as other data would also need to be transmitted, including textures.

3 Conclusions

We have described many of the considerations and problems that may be encountered when designing and developing a distributed virtual environment and outlined many of the constraints that should be considered during system design that could affect the performance of the final system.

4 Acknowledgements

The work reported in this paper has been supported in part by a contribution from the European Union under the ETHOS project (Telematics Applications Programme Project SU1105)

References

1. Virtual Reality Modelling Language, *VRML Consortium*, , 1999

2 V. Bourdakis, *The Future of VRML on Large Urban Models,* UK VRSIG '97 Proceedings

3 VRML Transport Protocol, *VRML Working Group*, , 1999

4 VRML-Streams, V*RML Streaming Working Group*, , 1999

5 C. Wiley, A. T. Campbell, S. Szygenda, D. S. Fussell and F. W. Hudson, *Multiresolution BSP Trees Applied to Terrain, Transparency, and General Objects,* Graphics Interface, pp. 88-96, May 1997.

6 D. Schmalstieg and G. Schaufler, *Incremental Encoding of Polygonal Models,* 30th Hawaii International Conference on System Sciences (HICSS-30), Maui, Hawaii, Jan. 7-10, 1997.

7 F. Sillion, G. Drettakis and B. Bodelet, *Efficent Impostor Manipulation for Real-Time Visualisation of Urban Scenery*, Eurographics 1997.

8 G. Schaufler, *Dynamically Generated Imposters, GI Workshop "Modeling - Virtual Worlds - Distributed Graphics",D.W.Fellner (Ed.), infix Verlag, November 1995, pp 129-135*.

9 J.W. Cooper, D.B. Arnold and A.M. Day, *Rapid Urban Modelling*, Eurographics 1999.

10 Joint Photographic Experts Group, *JPEG Frequently Asked Questions list*, http://www.faqs.org/faqs/jpeg-faq/

11 A. James and A.M. Day, *Accelerated Artificial Landscape Visualisation*, Eurographics 1999.

12 RealNetworks, http://www.realnetworks.com/