Virtual Dublin – A Framework for Real-Time Urban Simulation

John Hamill
Image Synthesis Group
Computer Science Dept.
Trinity College Dublin
Dublin 2, Ireland
John.Hamill@cs.tcd.ie

Carol O'Sullivan
Image Synthesis Group
Computer Science Dept.
Trinity College Dublin
Dublin 2, Ireland
Carol.OSullivan@cs.tcd.ie

ABSTRACT

We present a description of an urban simulation system work in progress. The goal is to create a large-scale immersive simulation of Dublin City that can then be used as a research test bed for various further projects including crowd and traffic simulation, and distributed graphics research. The long-term aim of the project is the evaluation of methods of handling of large data sets efficiently in real-time.

Keywords

Real-Time Animation, Urban Simulation, Occlusion Culling.

1. INTRODUCTION

The simulation of large urban environments in real-time has become a desirable goal in recent years for the fields of computer games, urban planning and architectural visualisation. This work is a description of an ongoing project to create an interactive model of the city of Dublin, Ireland, including all practical features of the city. The project aims to provide the user with a first-person view of the city and allow them to navigate as desired. The city aims to be seamless with no pauses required to load new 'levels' into memory. We also do not wish to restrict the user's movement to specific areas, instead we are investigating and developing methods to create a large-scale environment where every location is accessible and of sufficient detail to satisfy the user.

The level-of-detail required for the system is such that conventional hardware cannot cope with the sheer number of polygons required nor the volume of texture data used without resorting to a battery of techniques to reduce the processing and memory overhead. Fixed level-of-detail methods along with Occlusion Culling and Texture Management are the tools needed to accelerate a high detail virtual world to acceptable levels for user interaction.

This paper describes the background and features of the Virtual Dublin project, the use that is made of various acceleration methods and the future goals of the project.

Journal of WSCG, Vol.11, No.1., ISSN 1213-6972
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.

2. RELATED WORK AND PROJECT BACKGROUND

Simulation of large urban environments involves the rendering of a large number of polygons. In order to perform this rendering at interactive frame rates, there are a number of methods available. Teller and Séquin describe a portal-based method for visibility determination of axially aligned spaces [Tel91], but this would not be suitable for a large scale outdoor environment such as ours. Should we add internal areas to many buildings then this and other portal methods would be utilised.

One possible solution is to use Occlusion Horizons [Möl01] described by Möller et al., but unfortunately, this only operates correctly with continuous vertical occluders. Our city simulation contains several bridges and so a hybrid method would be needed at the least.

Another method, that would be more effective for our purposes, is the one proposed by Coorg and Teller [Coo97] for general-purpose real-time occlusion. Occlude shadow volumes [Bit98] combined with bounding box tests may also serve to provide an efficient occlusion strategy that requires only limited

Figure 1. View from O'Connell Bridge, Dublin

Figure 2. View of Westland Row, Dublin

pre-processing of the world. As part of future work, these and other algorithms will be tested to find the most effective one for our purposes.

The main objective of this research is to create a simulation of Dublin, starting with Trinity College, in which the user can explore the college and its surrounds in detail. The initial system was developed from an existing CAD model created by the College for renovation plans. This model was used as a base to create more detailed buildings that were textured with images acquired of the college buildings as detailed later. Further expansion of the system spread out from this central point, with the addition of major thoroughfares and their surrounding streets.
The urban simulation system we have developed is a Win32 OpenGL application using OpenAL for sound. It offers the user a first-person perspective view of the city environment and allows them to navigate at will. The covered area at present amounts to approximately 5km^2 of Dublin city centre. Control is through the mouse and keyboard and the system handles in a similar fashion to most first-person perspective computer games. Images of the system in operation can be seen in Fig. 1 and Fig. 2.

3. MODEL CREATION AND DATA ACQUISITION

Model Creation

The models used for the virtual city project were created by hand in Discreet 3D Studio Max R3.0 and R4.0. In general, the geometric detailing on the models was kept to a minimum, with the intention of using detailed texture maps instead. Once each model had been created, it was exported and converted to the Alias Wavefront OBJ Format. The models on average contained fewer than 500 polygons and utilised approximately 1Mb of textures each. The maximum reached was approximately 13500 polygons with 6Mb of texture data.

Hardware texture compression, along with paletted textures, was used to help reduce the burden on the 3D accelerator card once the models were loaded into the world.

At present, the models in the system are held as discrete entities that are loaded at runtime from their respective files. A simple text file is used to control the loading and positioning of the models in the world. This system allows for rapid prototyping of new additions to the system (including new object types) without requiring a redesign of a compiled level format. However, this also leads to potential duplication of data between models sharing vertices or textures. Once a finalised system has been established, then a compiled level format will be used which should reduce those wasteful duplications.

Data Acquisition

One of the major issues in a project such as this is the acquisition of accurate data to facilitate the modelling of the environment. While commercially available city maps may provide a starting point, they leave much to be desired when it comes to detail. Since the project involves the user interacting with the world in a first-person perspective mode, there needs to be the illusion that they are really walking down a city street with all its attendant features. These include accurate roads, footpaths, building facades, and pieces of street furniture such as traffic lights and street lamps etc. to give the user a believable world.

To this end, the following methods were used to acquire data for the Virtual City. Street plans collated for the original CAD model formed an initial street map that was extended as required. Individual streets were mapped out through visiting them and noting the locations of buildings and street fixtures.

Textures were acquired using a commercial digital camera. Source buildings were photographed from various angles and ranges. These photographs were

edited and modified to a suitable size and aspect in Adobe Photoshop. The average texture resolution used in the running project was 256x256 pixels, reduced down from high-resolution digital photographs, though some textures for important buildings were 512x512 pixels in size.

Further information on the city layout and the shape of various buildings was obtained by chartering a helicopter to take aerial photographs of the city. This allowed us to model the detailing on the city's roofs, as opposed to having a collection of flat blocks, and enabled us to revise errors in our estimated lengths and angles of streets. The photographs were taken at a low altitude, (approximately 400 meters), which provided a great deal of information about the target locations, including roof texture and geometry references.

Automated methods for urban data capture have been investigated by Teller et al [Tel] and these show great promise for reducing the time and effort required in modeling an urban environment accurately and in detail.

4. CULLING AND COLLISION DETECTION

One of the critical aspects of producing a system such as this, where there are potentially hundreds of objects, is to devise a method for culling out those objects that should not be rendered in the current frame. This section discusses the methods used for culling out occluded objects in the world and handling collisions between the user and the world.

Occlusion Culling

As part of the project, various occlusion algorithms will be evaluated along with polygonal simplification methods to allow a realistic large-scale city to be rendered efficiently and in real-time on domestic hardware. At the present time a custom occlusion algorithm is used in the system along with frustum culling to reduce the load on the graphics hardware. The occlusion culling method was devised to reduce the amount of texture drawing done each frame, given that the bulk of the burden placed on the hardware in our system revolves around the volume of textures we require. In our method, we create an occlusion buffer at a reduced resolution for the screen. To this buffer are drawn simplified versions of the current Potentially Visible Set of buildings (PVS). The models are drawn at a reduced resolution to a back buffer without lighting or texturing and those polygons with alpha-blended textures are ignored. Each building is assigned a specific colour as it is drawn, and when this buffer is scanned for colour components a new Visible Set can be created containing only those buildings that are actually visible in the scene. This final set of buildings is now rendered to the screen using full lighting, texturing and detail.

The drawback to our method is that it requires a complete draw of each object in the view frustum to select those few that should be drawn with full lighting and texturing. Given the low triangle counts of the buildings used and the high cost of overrunning the 3D accelerator's texture memory, this was judged to be an acceptable trade-off. The extra triangles drawn do not approach the limits of modern 3D accelerators, but the memory issue is a hard upper limit.

Simplified geometry could also be used in the occlusion buffer stage as long as the simplified geometry is contained wholly within the true fully-detailed model's geometry. This will give a conservative PVS but would save the drawing of many triangles. Due to the bottlenecks with texture memory, however, this has not been tried with the system at present. Further work will address the viability of hardware occlusion techniques such as the Nvidia NV_OCCLUSION_QUERY [Nvi02] OpenGL Extensions for data sets such as ours. Partitioning methods such as Binary Space Partition (BSP) Trees [Fuc80] or Visibility Octrees [Sao99] as described by Saona-Vázquez et al. shall also be investigated to determine the most effective method of dealing with large outdoor urban scenes.

Collision Detection

Collision detection with the world is performed using an implementation of the Möller Ray-Triangle Intersection Algorithm [Möl97]. The number of potential collisions is reduced using bounding box intersection tests leaving only several hundred ray-triangle tests per frame, which is acceptable for real-time operations. The user's position is modified by collision with the world geometry to allow sliding along walls and other obstacles in a similar fashion to most first person perspective computer games. Partitioning techniques such as BSP trees or Octrees as mentioned previously shall be used to provide a more efficient division of the world geometry as the world expands. This will in turn allow an increase in the number of object that can collide with the world geometry and each other.

5. WORLD EFFECTS

A simple Shadow Algorithm enhances the visual realism of the world. This projects each world model onto the ground plane in the OpenGL Stencil Buffer and then covers the scene with a single alpha-blended quad. This system is simple to implement, but is limited to a single ground plane. When the ground

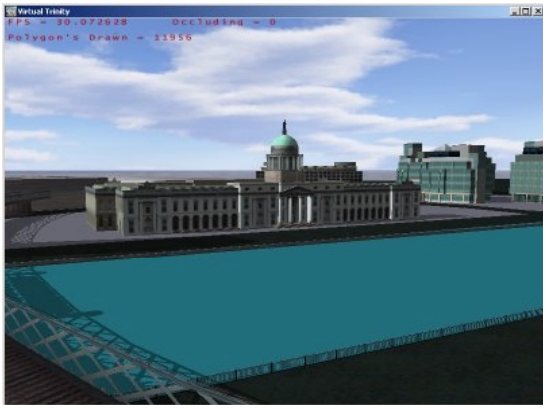

Figure 3. View of Custom House, Dublin

terrain is not flat, the shadows do not appear correctly in the scene. Either they float above the terrain or they are hidden beneath the surface. Another problem with the current Shadow Algorithm is that it blindly creates a shadow for every building in the world. This is done so that off-camera buildings can still correctly project their shadows into view. This results in many unnecessary draws to the Stencil Buffer. These can be eliminated if bounding boxes were to be stored for the shadow planes and tested against the view frustum during rendering. This will be added to the system once the shadows begin to pose a significant performance problem. Fig.3. shows the shadow algorithm in operation projecting the bridge onto the river.

Another visual enhancement used in the system is to vary the light level of the world depending on a configurable Time of Day option. This changes the lighting of the world and gradually alters the sky by fading from blue to starry black. This will eventually be refined with the addition of illumination maps to simulate building lights at night.

Additionally, we utilise 3D positional audio to provide the user with a more immersive experience when using the system. City noises were recorded and using OpenAL, we were able to add ambient sounds to the system. Specific sound effects such as train and automobile noise will be recorded and added to the world as needs arise.

The OpenGL and OpenAL [Ope00] libraries were used to provide the primary graphical and aural interface to the system. OpenGL was chosen for its ease of expansion, portability and consistent interface. By virtue of sharing its naming conventions and many concepts with OpenGL, the OpenAL library was chosen for addition to the system, providing positional 3D audio support.

At present, the Paletted Textures and S3 Texture Compression OpenGL extensions are used to facilitate the large amount of texture data that must be loaded into the world.

6. SYSTEM ISSUES AND LIMITS

The choice to trade model detail for texture detail is a problematic one. On the one hand, it greatly reduces the time required to add a new building to the system and since each building contains relatively few polygons, they do not strain the geometry transform capabilities of the graphics hardware. On the other hand, a large texture memory overhead is unavoidable with this approach. This can lead to memory thrashing on the graphics hardware unless sufficient precautions are taken.

For example, the current data set for Virtual Dublin consists of 80 models with 6.1 Megabytes of geometry data and 1,151 images totalling 210 Megabytes of texture data. In order to load all this data at runtime requires substantial resources. To avoid disk thrashing, at least 256Mb of RAM are required and even with the use of texture compression a 3D graphics accelerator with greater than 64Mb of memory is needed. The hardware that the system currently runs on is an Intel P4 2Ghz with 512Mb Ram and a Creative Geforce4 Ti4600 128Mb 3D Accelerator card.

Frame rates for the current system vary between 25 and 60 frames-per-second on this hardware depending on scene complexity and altitude. Due to the greedy nature of the Shadow Algorithm, many unneeded draws occur that adversely affect the frame rate. Furthermore, when viewing the Virtual City from above a certain altitude, the occlusion system provides almost no benefit (since most world objects are visible to the user, see Fig.4.) leading to the lower frame rate of 25fps. When operating at street level, however, approximately eighty percent of buildings are occluded and frame rates reach the maximum value of 60fps.

Figure 4. Aerial View of Trinity College Dublin

7. FUTURE WORK

As work on the Virtual City proceeds, there are a number of areas of expansion available to us. The short term will focus on evaluating various existing occlusion methods and devising new methods tailored to our needs. Impostors and Dynamic Level-of-Detail methods will also be investigated to improve the capability of the system for handling large city areas.

The spatial expansion of the modelled portions of Dublin City will continue with the eventual aim of producing a detailed model of the inner city area. Impostor methods such as bill boarding will be investigated to evaluate their usefulness in urban modelling. Weather features are a possibility for the system. The addition of rain, snow, fog, and other environmental effects may add to the level of immersion of the current system.

Vegetation and Foliage are currently quite sparse in the system with only some basic tree models dispersed among a few locations (cf. Figure 5.). It is desirable to add more realistic plant and tree models to the system, but is not critical at present.

Others are working on extending the system we have developed. The current city engine is facilitating related projects including large-scale human crowd simulation and traffic congestion modelling. Another project involves the creation of a parallel "Virtual Viking Dublin" simulating the city as it was in the 10th and 11th Centuries AD.

The system is also being used as an experimental test bed for Network Distributed Graphics using the Chromium [Hum02] system. This will be combined with a CAVE [Cru92] facility to allow a completely immersive environment with which to explore the Virtual City.

Finally, work is being done on distributed navigation systems using Handheld Devices. This would allow a visitor to an area to call up a 3D Map of their current location for direction to their destination.

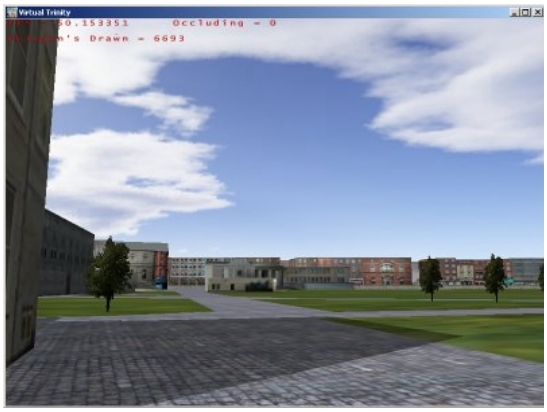

Figure 5. College Park, Trinity College Dublin

8. ACKNOWLEDGEMENTS

We gratefully acknowledge the work done by the following people in the acquisition of image data and modelling for the Virtual City: Duncan Aitken, Andrew Brosnan, Gareth Carter, David Coen, Charles Smith, Ronan Watson.

9. REFERENCES

[Bit98] Bittner J., Havran V. and Slavík P. Hierarchical Visibility Culling with Occlusion Trees. In Proceedings of Computer Graphics International '98 (CGI'98), pp.207-219, 1998.

[Coo97] Coorg S., and Teller S. Real-Time Occlusion Culling for Models with Large Occluders. In ACM Symposium on Interactive 3D Graphics conf.proc., pp.83-90,189, 1997.

[Cru92] Cruz-Neira C., Sandin D.J., DeFanti T.A., Kenyon, R.V., and Hart J.C. The CAVE: Audio Visual Experience Automatic Virtual Environment. In Communications of the ACM 35, pp.67-72, 1992.

[Fuc80] Fuchs H., Kedem Z., and Naylor B. On Visible Surface Generation by a priori Tree Structures. In Computer Graphics (proc. Siggraph 1980), 14(3), pp.124-133.

[Hum02] Humphreys G., Houston M., Ng R., Frank R., Ahern S., Kirchner P. D., and Klosowski J. T. Chromium: A Stream-Processing Framework for Interactive Rendering on Clusters. ACM Transactions on Graphics (proc. Siggraph 2002), 21(3), pp.693-702, 2002.

[Möl01] Downs L., Möller T., and Sequin C. H. Occlusion Horizons for Driving through Urban Scenery. In ACM Symposium on Interactive 3D Graphics conf.proc., pp.121-124, 2001.

[Möl97] Möller T., and Trumbore B. Fast, Minimum Storage Ray-Triangle Intersection. Journal of Graphics Tools, 2(1), pp.21-28, 1997.

[Nvi02] nVidia NV_OCCLUSION_QUERY Specification, February 2002. http://www.nvidia.com/dev_content/nvopenglspecs/GL_NV_occlusion_query.txt

[Ope00] Loki OpenAL Draft Specification, June 2000. http://www.openal.org

[Sao99] Saona-Vazquez C., Navazo I., and Brunet P. The Visibility Octree: A Data Structure for 3D Navigation. In Computer & Graphics, 23(5), pp.635-644, 1999.

[Tel91] Teller S., and Séquin C. H. Visibility Preprocessing for Interactive Walkthroughs. In Computer Graphics (proc. Siggraph 1991), pp.61-69.

[Tel] Teller S., et al. MIT City Scanning Project: Fully Automated Model Acquisition in Urban Areas. http://city.lcs.mit.edu//city.html