OPTIMISING REAL-TIME VIRTUAL REALITY ARCHITECTURE PRESENTATION

TAN BENG KIANG AND DANIEL HII JUN CHUNG
Department of Architecture, School of Design and Environment, National University of Singapore Singapore.
akitanbk@nus.edu.sg; g0500819@nus.edu.sg

Abstract. In real-time virtual reality walkthrough, there is a minimum frame replacement rate needed to maintain the "smoothness" of walking through a scene. This paper identifies the variables that affect efficient real-time walkthrough of architectural models on an affordable PC platform. Experiments were conducted to determine how each variable affects the frame rate and memory consumption. The results will help guide architects and students to better manage their CAD model in order to achieve smooth real-time walkthroughs in their presentation.

1. Introduction

Architecture design idea presentations have been evolving through history in conjunction with technology - hand drawn sketches, hand drafting, blueprint prints, colour prints, computer perspective rendering, animation and now virtual reality presentations. Real-time rendering in virtual reality has come a long way in history (Burdea and Coiffet 2003). Real-time walkthroughs offer viewers a 3D and immersive experience within, outside and around a building and contribute to a better understanding and appreciation of the spaces. It is a good medium for students to learn about architecture and for architects to present projects to clients. Virtual reality (VR) software now runs on PC platforms, however, a good real-time rendering still requires high computer processing power. Powerful computers and graphic cards can improve real-time performance. Even though hardware prices are dropping every year, for architecture schools and architectural firms interested in using VR presentations, they cannot afford to change their computers as fast as computer hardware advances. They also have limited resources and programming knowledge to write new algorithms that optimise and create best results in real-time rendering engines. Thus, the authors are interested in a best practice approach that guides architects and architecture students on how to plan and manage the creation of their CAD model for an efficient real-time virtual reality walkthrough presentation on affordable hardware platforms.

There is research done to identify the factors that affect real-time rendering (Af Klercker 2000, Wimmer and Wonka 2003). However, they do not identify which are the biggest contributor to degradation in frame rates and computer memory consumption, both of which will slow down the performance of a real-time walkthrough.

In this study, we conduct experiments to determine how each variable affects the frame rate and memory consumption and thus the performance of the walkthrough. Our workflow is creating a CAD model and doing texture mapping using a standard commercial software (Autodesk® 3ds Max® 8) and then exporting it to a virtual reality software EON Studio™ (with EON Professional™) for real-time visualization (Figure 1).

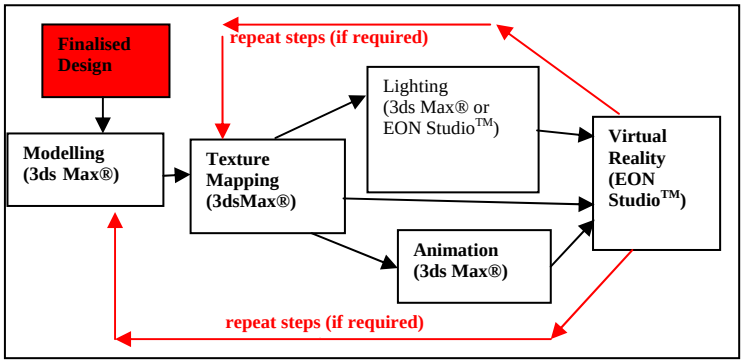

Figure 1. Workflow

There have been architectural and urban visualizations done using game engines such as Quake, Half Life® and Oblivion™ (So-Yeon and Tutar 2004, Digital Urban, UCL 2006). However, we have excluded using them in our experiments as they are meant for game developers to develop 3D games and the architecture community has limited access to them. To use them also requires programming skills which most architecture students are lacking.

Our experiment is basically conducted using the same model on the same hardware platform and increasing the different variables of the model to see how much it affects the overall performance of the presentation. It is hope that the final result will serve as a guide for future users to plan their process of modeling well so that they would not engage in the unnecessary time consuming process of rectifying and modifying the model over and over again to make it run smoothly on the virtual reality platform.

2. The Variables

Real-time rendering means maintaining continuous frame replacements at around 6 frames per second (Möller 2002). This refresh rate is high enough for our eyes to detect the arrival of each frame to replace the one before. Thus, any rate above 6 will be definitely preferred. There are a few variables that affect frame rates and computer memory usage and they are geometry count (i.e. objects), triangle count and number of textures used (Wimmer and Wonka 2003).

A basic surface comprises a few vertices that make up a polygon. The number of vertices and surfaces in the scene will always determine the complexity of the model. Anything created in the scene will require light to show it. Computations of light at vertices will be done and this will interpolate over the surfaces (Möller 2002). Therefore, the more vertices and surfaces there are, the more calculations are required to display the entire 3D model.

The number of objects of a model in the scene will also increase calculations. The more parts there are the more computation is required for shaded and none shaded areas depending on how they all come together.

Texture mapping helps to show the qualities of the materials and in totality, the architectural design. In an architectural model, most surfaces require texture mapping to express the design. The correct methods and formats need to be used to make sure that the most efficient texture is selected.

All these elements in the model collectively affect performance of real-time rendering. Of course, optimization needs to be done at all stages to ensure we do not create something extremely unnecessary which requires more unnecessary calculations by the computer with nothing gained. Good model management (Burdea and Coiffet 2003), texture management and light management all come hand in hand for an overall efficient model for virtual reality visualization.

Real-time rendering, uses the graphic processing unit and the computer processing unit and simulation consumes video memory and the system RAM. When an extremely large and complex model is loaded, virtual memory has to be borrowed from the hard disk. If all the variables such as geometry count triangle count and number of textures are not well managed, low frame rate and memory over-usage will cause the performance of the virtual reality walkthrough to be no longer real-time. Navigation will be slow especially when dealing with huge scenes (Steed 1997).

3. The Experiment

For the experiment, we did not use a scene with simple primitive objects, as it would not be representative of a situation in an architectural office or in an architectural school. Instead we created the model of an unbuilt architecture project, Cloud Forest Biosphere, designed by Aga Khan award winner, Geoffrey Bawa. It is a relatively complex model (geometry count, triangles count and textures) with its large span truss roof structure, an undulating terrain, water, trees, plants and different materials such as glass and stones (Figure 2).

Figure 2. Cloud Forest Biosphere (screen shot from VR software)

The model is used to compare the increase in geometry amount, triangle amount and texture amount and its effect on frame rate. The model as well as

the calculation of the number of triangles, geometry and texture is done in 3ds Max®. The number of triangles, geometry and texture can be counter checked in EON Studio™ and finally when the real-time simulation is run, we check the frame rate directly in EON Studio™.

3.1 CONTROL

The same computer is used for all the experiments with a specification of Intel® Pentium IV Dual Xeon™ 3.2 GHz; 2GB system RAM and Leadtek AGP8X 256MB nVidia® Quadro FX 3000G graphic accelerator card running on Microsoft® Windows® XP Professional Version 5.1.2600 Service Pack 2 Build 2600 operating system.

The experiments are conducted by the same person to maintain consistency. The room temperature is the same throughout the whole experiment so the hardware performance is stable and not interfering with actual results. Each variable namely geometry count, triangle count and number of textures is increased independently in the model and the corresponding frame rate and vertex memory used are recorded.

3.2 THE VARIABLES

3.2.1 Texture Type

We did a texture format comparison to determine the most efficient type to be used in the model for real-time simulation. The common texture formats used in texture mapping are JPEG, PNG and TARGA. A less common format DDS (DirectDraw Surface) texture created in power-of-2 resolution (example: 512X512, 1024X1024, 2048X2048) and mipmap is found to be the most efficient texture format for real-time rendering. It is an uncompressed format so therefore the application does not need to decompress it every time the simulation starts. The other formats are compressed and this means they are generally smaller in size in comparison with DDS.

The Microsoft® DirectDraw Surface (.dds) is a texture file format introduced with DirectX 7.0 and later in OpenGL. It can store with mipmap or without mipmap levels. Mipmap is a texture mapping technique which is using multiple texture maps. Every mipmap is half the size of the previous one, providing several texture maps of various levels of depth. MIP stands for “multum in parvo”, which is Latin for “many in a small place”.

A 2048x2048 and a 1024x1024 texture are saved in both JPEG and DDS formats. When the model simulation is run in EON Studio™, the JPEG version consumed 53MB and the DDS version consumed only 8MB, which is almost 7 times the difference. We found that DDS format clearly uses less memory in comparison with common texture formats like JPEG, PNG and TARGA.

To find out if texture format type affects frame rate, we tested two existing big projects that had common texture format i.e. JPEG and replaced them with textures in DDS format. In a model of Ming De of Chang’An (consisting of 1,148,005 triangles), a building complex in ancient Tang

dynasty, the frame rate improves from an average 2.5 frames per second to 7 frames per second with DDS format textures. In another project, Warren Housing (consisting of 2,623,190 triangles), a large site with many blocks of housing, the use of DDS format texture improves the frame rate from 4 frames per second to 7 frames per second. Thus models with DDS format textures are more efficient in terms of memory usage and easier to navigate in real-time.

For the rest of the experiments below, the DDS texture type was used.

3.2.2 Geometry Count

Figure 3 below shows the graph plotted between frame rate and triangles count for the geometry variations of 50, 100, 200, 400, 800, 1600 and 3200.

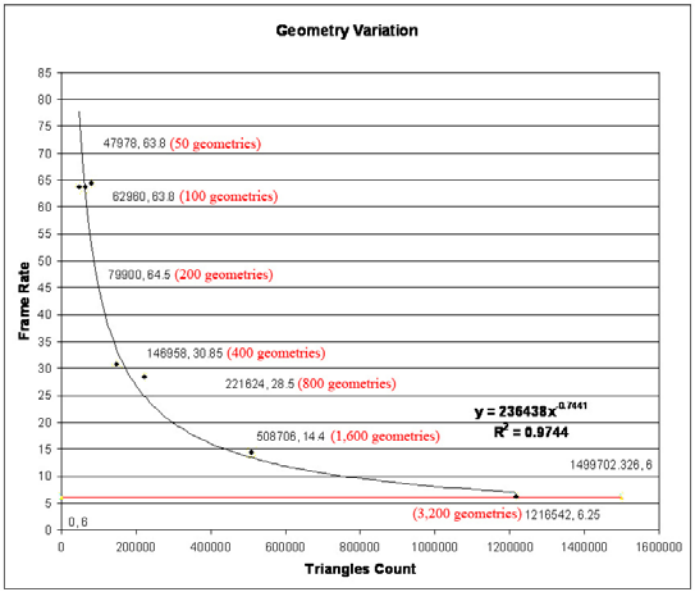

Figure 3. Geometry Count (50, 100, 200, 400, 800, 1,600, 3,200 geometries)

Table 1 below shows the amount of vertex memory used from geometry count of 50 to 3200.

TABLE 1. Geometry Count with Vertex Memory

Models	Geo 50	Geo 100	Geo 200	Geo 400	Geo 800	Geo 1600	Geo 3200
Vertex Memory (KB)	807	1061	1345	2292	3342	4174	4174

This experiment is done to determine whether the increase of geometry will affect the frame rate as well as the vertex memory. It showed that an increase in the number of geometry will increase the vertex memory used. The increase from 50 to 3200 geometries in the scene increases the vertex memory from 807 Kb to 4174 Kb, which is an increase of 80.67%. The frame rate for number of geometries from 50 to 3,200 decreased from 64.5 to 6.25, a drastic drop of 90.31% (Figure 4).

3.2.3 Triangle Count

Figure 4 below shows a graph plotted between frame rate and triangles count.

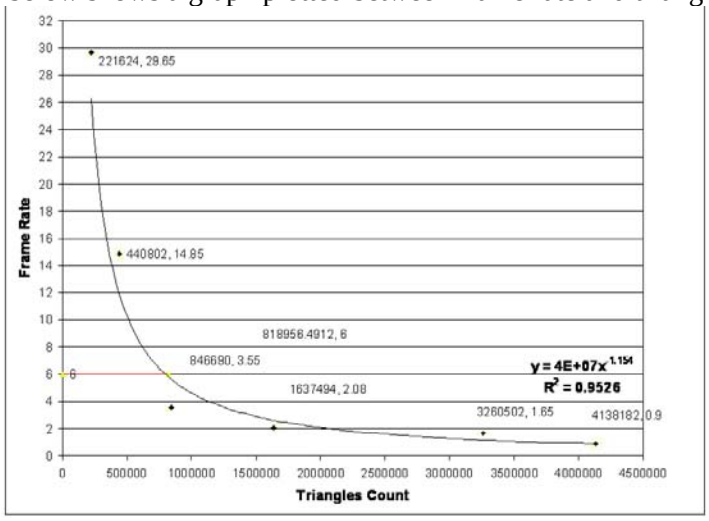

Figure 4. Triangle Count (200,000, 400,000, 800,000, 1,600,000, 3,200,000, 4,000,000 triangles)

Table 2 below shows the amount of vertex memory required from 200,000 triangles until 4,000,000 triangles.

TABLE 2. Triangle count with Vertex Memory

Models	Tri 200k	Tri 400k	Tri 800k	Tri 1600k	Tri 3200k	Tri 4000k
Vertex Memory (KB)	3342	4169	4174	4171	4174	4174

This experiment showed that the increase of triangles does not translate to increase use of vertex memory. The increase of triangles count does not increase the use vertex memory much, from 3342 Kb to 4171 Kb only, which translates to only 19.93% increase. The frame rate however, drops significantly to below 0.9 for the largest number of triangles. That is a huge drop of 96.96%. In comparison to geometry count, we can clearly see that the more geometry in the scene will require more memory than the increase of triangle counts.

3.2.4 Texture Count

Experiments were conducted to determine whether the number of textures used in the scene affect the frame rate. It is shown in Table 3, from 10 to 100 textures, the difference of frame rate drop is 0.1 to 2.8, which is not significant. In other words, an increase in the number of textures used in the scene will not affect frame rate much. It will however, result in an increase in texture memory and vertex memory used.

TABLE 3. Frame rate of models of different sizes with 10 - 100 textures

model size (triangles)	vertex memory (Kb)	texture memory (Kb)	frame rate	difference
200,000	4650 - 4925	32 - 13754	**21.2 - 20.3**	**0.9**
1 million	6622 - 10449	32 - 13853	**4.4 – 2.5**	**1.9**
2 million	7552 - 24315	32 - 15133	**3.5 – 0.7**	**2.8**
4 million	6275 -crash	32 - crash	**1 - crash**	**0.1**

4. Results

Using the graphs plotted from the above variables experiments, we can determine the limit of the number of triangles in a model that will cause it to be displayed at an unacceptable frame rate and thus will not be a real-time walkthrough and affects the VR presentation. If a model exceeds such a limit, for efficient navigation, the user should divide the project into different model parts and the software can load the different parts as the user navigates close to it.

We can also determine how the number of geometry count and the number of textures will affect the usage of memory looking at the trends of the graph and tables. With that, users can plan the amount of memory to be used in a scene given what they have in their video acceleration card and RAM. Users can efficiently plan the usage of memory especially in crucial conditions when the frame rate is at the borderline of acceptable rate. Good planning can help improve frame rate by 3 to 4 frames per second as illustrated in the Chang' An's Ming De and Warren models.

5. Conclusion

Of all the variables, triangle count is proven to affect the frame rate most. An increase in the number of triangles will decrease the frame rates significantly. Any model of more than 800,000 triangles will have a frame rate that falls below the threshold frame rate for real-time navigation. So when creating CAD models of large projects, users must bear in mind to decrease the number of triangles in order to optimise the model for real-time visualisation. For instance, cloning of parent objects should be used as far as possible if any geometry in the same scene differs only in scale, rotation in x, y or z axis.

An unexpected finding is that an increase in the number of textures (where an efficient texture type DDS is used) in a model will not decrease frame rates significantly i.e. will not cause the navigation of a model in real-time to slow down. In a model of 200,000 triangles, a ten-fold increase in the number of textures (from 10 to 100 textures) results in only a frame rate drop of 0.9.

In conclusion, a best practice to achieve optimum performance in real-time rendering is primarily to keep the number of triangles in a model as low as possible through using cloning for repetitive elements, followed by

reducing the number of geometry (objects) as much as possible by attaching object parts together especially if the material is the same and finally using an efficient texture format such as DDS to improve frame rates. If these strategies cannot improve the performance of real-time rendering, then other techniques present in the VR software such as culling and level of detail may have to be applied.

In this paper, we only study how each variable independently affect the frame rate. Future research can be done to have all the variables such as number of triangles, geometry count, and textures considered at the same time through a multiple regression analysis to determine all of them as a whole in affecting the performance of the virtual reality visualization.

Acknowledgements

The authors would like to thank the National University of Singapore for their research fund R-295-000-052-112, students Chew Chun Boon, Chris Hee Jee Kwang, and Siow Ming Khang who assisted in the experiments, Prof. Heng Chye Kiang and Dr. Stephen K. Wittkopf for permission to use Chang'An model and Warren Housing models respectively.

References

Af Klercker, J. :2000, Modelling for Virtual Reality in Architecture, *Pro. of the 18th eCAADe Conference*, pp.209-213.

Association for Computing Machinery, (2006 Nov). *Real-Time Rendering Resources.* Mass: ACM. Retrieved 17 August, 2006 , http://www.realtimerendering.com.

Burdea G. and Coiffet P.: 2003, *Virtual reality technology*, J.Wiley-Interscience, Hoboken, NJ.

Digital Urban, (2006), *Game Engines*, UK: UCL. Retrieved 11 November, 2006, http://digitalurban.blogspot.com/search/label/Game%20Engines

Fernando, R. (ed): 2004, *GPU gems : programming techniques, tips, and tricks for real-time graphics*, Addison-Wesley, Boston, MA.

Fukuda T., Kazuhiro S., Yeo W. and Kaga A.: 2006, Development and Evaluation of a Close-range View Representation Method of Natural Elements in a Real-time Simulation for Environmental Design - Shadow, Grass, and Water Surface, *Pro. of the 24th eCAADe Conference*, pp.58-65.

Möller, T.: 2002, *Real-time rendering*, AK Peters, Natick, Mass.

Slater, M.: 2002, *Computer graphics and virtual environments : from realism to real-time*, Addison Wesley, Harlow, England; Singapore.

So-Yeon, Y., Tutar, M.: 2004, A Comparison Study Between A Computer Game & A Non-Game Web3D Environment, S. Gero,.J., Chase, S., Rosenman, M. (Eds.), *Proceedings of the Ninth Conference on CAADRIA 2004*, pp.601-612.

Steed A.: 1997, Efficient navigation around complex virtual environments, *Pro. of the ACM symposium on Virtual reality software and technology*, pp.173-180.

Willmott J., Wright L.I., Arnold D.B., Day A.M.: 2001, Rendering of large and complex urban environments for real time heritage reconstructions, *Pro. of the 2001 conference on Virtual reality, archaeology, and cultural heritage*, pp.111-120.

Wimmer M. and Wonka P.: 2003, Rendering time estimation for real-time rendering, *Proc. of the 14th Eurographics workshop on Rendering*, pp.118-129.