ADAPTIVE COHERENT HIERARCHICAL CULLING ALGORITHM FOR UTM CAR DRIVING SIMULATOR

MOHD KHALID MOKHTAR

UNIVERSITI TEKNOLOGI MALAYSIA

Replace this page with form PSZ 19:16 (Pind. 1/07), which can be obtained from SPS or your faculty.

Replace this page with the Cooperation Declaration form, which can be obtained from SPS or your faculty.

ADAPTIVE COHERENT HIERARCHICAL CULLING ALGORITHM FOR UTM CAR DRIVING SIMULATOR

MOHD KHALID MOKHTAR

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

JANUARY, 2013

Dedicate to the people that involve in this research. Thanks to Allah gives me the change to finish this research for my master study.

ACKNOWLEDGEMENT

Praise be to Allah, we seek His help and His forgiveness. We seek refuge with Allah from the evil of our own souls and from our bad deeds. Whomsoever Allah guides will never be led astray, and whomsoever Allah leaves astray, no one can guide. I bear witness that there is no god but Allah, the One, having no partner. And I bear witness that Muhammad is His slave and Messenger. I wish to express my sincere appreciation to my supervisor, Associate Professor Dr. Mohd Shahrizal Sunar who has supervised me during four years of my study. He has always supported on my research ideas and activities. With his experiences, encouragements, guidances, critics and friendship brought much impact to my study and also personal life.

I am also would like to thank all members in UTM ViCubeLab research group at Faculty of Computer Science and Information System for their supports, interests and cooperations. Ab Al Hakam, Siti Aida, Hassan, Ku Faisal, Ismahafizi, Zakiah, Nuramalina, Iklima, Dr. Hoirul, Norsharina, Azhar and those friends who close to me during my year of studies. Hope you all will success in the near future. I am also indebted to Malaysia Government Programme for funding my studies with National Science Fund Scholarship. I able to complete my studies and concentrate on my research without facing any financial problems. Unfortunately, this is not possible without support from my family. I am grateful to all my family members especially, my mother Halimah Binti Harun and all my siblings: Khirriri, Khairul, Khamal, Khairuddin, and Norhaslinda.

ABSTRACT

Driving simulator is a virtual reality tool that emulates actual driving environment. In a virtual world, feedback between the user and the application is a very critical aspect to be considered. Fundamentally, the drop of frame rates that influences feedback can cause control delay and disruption of user interactivity. To maintain human and computer interactivity, the best possible quality of rendering graphics would have to be within 60 frames per second. To achieve this quality, good selection of Visibility Culling (VC) algorithms are needed one of which is the Online Occlusion Culling (OOC) algorithm. The OOC requires no pre-processing and is suitable for dynamic scenes. The characteristic of the 3D dynamic scene is one of the main factors to efficiently integrate OOC into an existing driving simulator system. Hence, this research proposed an adaptive algorithm for Coherent Hierarchical Culling (CHC) algorithm derived from OOC to manage complex and dynamic objects in a dynamic driving simulation scene. The CHC requires a two-level process: pre-processing and online processing which are implemented to manage the occlusion queries. The proposed adaptive algorithm aims at improving CHC efficiency by manipulating two parameters which are the car movement speed and the visibility threshold. An experiment was conducted using car driving simulator engine tested on winding and hilly road, and an open straight road. The experiment showed that the speed of the car influenced the number of objects to be culled based on distance whereas the varying visibility thresholds reduced the popping artifacts problem of higher visibility threshold. Concurrently, both of them were able to reduce number of insignificant objects to be rendered. The results of the research showed that this adaptive algorithm improved the CHC rendering performance and maintained visual quality. This research has proven that the adaptive algorithm really improved upon the CHC for real-time simulator applications.

ABSTRAK

Simulator pemanduan merupakan satu alat realiti maya yang menggambarkan suasana pemanduan yang sebenar. Dalam dunia alam maya, tindakbalas antara pemandu dan aplikasi adalah aspek yang paling kritikal untuk diambil kira. Pada asasnya, penurunan bilangan kerangka per saat mempengaruhi maklumbalas yang menyebabkan kelewatan kawalan dan mengganggu interaktiviti pengguna. Kualiti paparan grafik yang terbaik memerlukan sekitar 60 kerangka per saat untuk mengekalkan interaktiviti antara manusia dan komputer. Untuk mencapai kualiti ini, pilihan yang baik pada algoritma Pemilihan Kebolehanlihatan (VC) diperlukan, dimana salah satunya ialah algoritma Pemilihan Terlindung Segera (OOC). OOC ini tidak memerlukan pra-pemprosesan dan ia sesuai untuk persekitaran yang dinamik. Kriteria yang ada dalam persekitaran 3D menjadi faktor penting dalam mengintegrasikan OOC secara efisien dengan sistem simulator pemanduan yang sedia ada. Justeru, penyelidikan ini mencadangkan satu algoritma penyesuaian untuk algoritma Pemilihan Koheren Berhierarki (CHC) dibawah OOC bagi mengurus objek-objek yang kompleks dan dinamik dalam persekitaran simulator pemanduan. CHC memerlukan dua pemprosesan untuk menguruskan pertanyaan terlindung iaitu pra-pemprosesan dan pemprosesan segera. Algoritma penyesuaian yang dicadangkan bertujuan untuk meningkatkan keberkesanan CHC dengan memanipulasikan dua parameter iaitu kelajuan kereta dan nilai ambang kebolehlihatan. Proses pengujian dilaksanakan menggunakan enjin simulator pemanduan pada jalan selekoh berbukit dan jalan besar yang lurus. Hasil ujian menunjukkan bahawa kelajuan kereta mempengaruhi jumlah objek yang akan dibuang berdasarkan jarak manakala nilai ambang kebolehlihatan yang berbeza-beza dapat mengurangkan masalah artifak pemunculan oleh nilai ambang yang tinggi. Pada masa yang sama kedua-duanya dapat mengurangkan jumlah penjanaan objek yang tidak penting. Hasil penyelidikan menunjukkan algoritma penyesuaian ini dapat meningkatkan prestasi CHC dan mengekalkan kualiti penjanaan. Kajian ini membuktikan algoritma penyesuaian telah menambah baik CHC untuk aplikasi masa nyata bagi simulator.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

VR	–	virtual reality
fps	–	frame per second
PVS	–	potential visibility set
TC	–	Temporal Coherence
3D	–	three-dimensional
CHC	–	Coherent Hierarchical Culling
PVS	–	potential visibility set
HZB	–	Hierarchical Z-buffer
HOM	–	Hierarchical Occlusion Maps
NOHC	–	Near Optimal Hierarchical Culling
HITL	–	Human-in-the-Loop
DOF	–	Degree of Freedom
NOHC	–	Near Optimal Hierarchical Culling
CHC++	–	Coherent Hierarchical Culling Revisited

LIST OF SYMBOLS

x	–	X axis
y	–	Y axis
z	–	Z axis

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Rendering physically correct images with rich visual effect is one of driving forces to the advancement of computer graphics (Scherzer *et al.*, 2010). The evolution of real time graphics with better graphics hardware, more detailed models and better techniques gives challenge to researcher and developer to more and more sophisticated algorithm (Mattausch, 2010). Computer graphics refer to a field of computer science where computation techniques are applied for digitally synthesizing and manipulating visual content. Contribution from this area has been made impact to many different type of media such as animation, movies and video game industry. Many researchers have been done in applying computer graphics in simulation and animation for scientific visualization and entertainment. One major problem faced by practitioners is battle with the trade-off between complexity and performance (Luebke *et al.*, 2002).

Both scene complexity and rendering performance are played importance roles in interactive computer graphics (Luebke *et al.*, 2002). Scene complexity refers to the number of primitives (pixels, edges or vertices) which are in view depending on the viewer position and orientation (Gilbert, 1994). Complexity of a virtual scene can influence the rendering performance if the scene is too complex because it is not interactively rendered and frame rate drop (Luebke *et al.*, 2002). The performance of rendering refer to the number of frames rendered in one second also known as frame rates. Low frame rates can cause the system more real-time rendering rather than

interactive. Interactivity in real-time rendering refer to suitable frame rate that require for an application. 60 frame per second (fps) is considered enough frame rate for human observer (Mattausch, 2010; Moller *et al.*, 2008) because currently most LCD monitors and TVs can refresh at least at 60Hz. In one frame, calculation performed involve not only rendering algorithms but also other calculations in application such as input processing, sound rendering and artificial intelligences. All these shared calculations must perform within this time budget while maintain between interactive application and realism of the virtual scene especially in single frame (Scherzer *et al.*, 2011).

Driving simulator is one example of real-time rendering application which is useful in providing a synthetic experience and interaction for it′s user in real-time (Whyte, 2002). Driving simulator consists of computer hardware, software, the input and output devices, the data and the users that support the usage of an interactive, spatial and real-time medium (Burns and Wellings, 1997). The main issue in driving simulator system is to develop a system that able to maintain interactive frame rate and fidelity like a real world (Whyte, 2002). Managing the scene complexity while maintaining interactive frame rates of driving simulator system needs the implementation of visibility algorithm (Cohen-Or *et al.*, 2003).

Visibility algorithm is one of the most important component in real-time rendering system architecture (Bittner *et al.*, 1998; Bittner and Wonka, 2003; Cohen-Or *et al.*, 2003; Coorg and Teller, 1996; Lengyel, 2003; Mattausch, 2010). Since the beginning of computer graphics, visibility algorithm is a fundamental and crucial problem that attract researcher and developer to provide solution (Bittner and Wonka, 2003; Cohen-Or *et al.*, 2003; Mattausch, 2010). There are various kind of visibility algorithms for several visibility problems domain one of them is visibility culling (Bittner and Wonka, 2003). The goal of visibility culling is to reduce computational cost of rendering by limit the portion of complex scene to visible part. Visibility culling play roles by allowing the real-time rendering application to achieve output-sensitive where the render time depends only on the complexity of the actual output, not the scene complexity. It means that any objects that do not make contribution to final image will be removed from calculation.

Occlusion culling is one of the visibility culling technique that still have a room for improvement for research (Staffans, 2006). Occlusion culling can be divided into visibility preprocessing and online occlusion culling (Mattausch, 2010). The introduction of occlusion queries in graphic hardware encouraged research direction to use online occlusion culling. This online occlusion culling called hardware occlusion queries that requires no pre-processing, easy and simple to implement and can handle dynamic scene (Mattausch, 2010). However, only when the introduction of Temporal Coherence (TC) concept in managing hardware occlusion queries has made this technique practicable used in complex virtual system. Introduction of coherent hierarchical culling (CHC) algorithm by Bittner *et al.* (2004) exploits the temporal coherence (TC) concept to avoid CPU stall and GPU starvation in performing hardware occlusion queries. CHC algorithm is suitable to be implemented in arbitrary and dynamic scenes (Bittner *et al.*, 2004; Mattausch, 2010; Scherzer *et al.*, 2011). Thus the driving simulator is suitable as a platform to implement this algorithm which can provide dynamic scenes. A research need to be done to make sure after CHC technique is implemented in the driving simulator system maintains as a real-time rendering application with interactive frame rates and realism of it′s environment .

1.2 Problem Background

Visibility algorithms in computer graphics started with the introduction of visibility line and surfaces method in a synthesized images of a 3D scene (Bittner and Wonka, 2003; Cohen-Or *et al.*, 2003). Roberts (1963) claimed himself as the first person introduced solution to determine hidden line segments. Many different types of visibility algorithms emerged together with the development of computer graphics. Based on survey done by Bittner and Wonka (2003), visibility algorithms classified within several problem domains. This research focuses on domain called visibility culling. Visibility culling started with introduction of two algorithms: back-face culling and view frustum culling (Foley *et al.*, 1990).

Most research in visibility culling later focus on occlusion culling algorithm because the complexity of the algorithm involved interrelationship between polygons (Cohen-Or *et al.*, 2003). A survey on occlusion culling trends and recent

developments is done by Cohen-Or *et al.* (2003) provided researcher a taxonomy on visibility culling. Bittner and Wonka (2003) later provided a better survey on visibility culling where two problems domains are involved in most visibility culling. There are visibility from point and visibility from a region (Bittner and Wonka, 2003; Mattausch, 2010). The visibility from a region worked corresponds to preprocessed visibility is applied offline in preprocessing stage and potential visibility set (PVS) is computed (Aila and Miettinen, 2004; Cohen-Or *et al.*, 1998; Durand *et al.*, 2000; Koltun *et al.*, 2000; Lloyd and Egbert, 2002; Schaufler *et al.*, 2000; Teller, 1992; Wonka *et al.*, 2000). The visibility from point corresponds to online culling is applied online that requires no preprocessing for each particular viewpoint and work also for arbitrary environment (Bittner *et al.*, 1998; Greene *et al.*, 1993; Hudson *et al.*, 1997; Klosowski and Silva, 2001; Wonka and Schmalstieg, 1999; Zhang *et al.*, 1997). Most online occlusion culling algorithms are image based where any objects draw to screen depend on its contribution to the current frame.

The evolution of occlusion culling has seen on those who preform large amount of preprocessing and those who perform occlusion culling online with less dependable to preprocessing (Staffans, 2006). Visibility preprocessing provides simple and powerful solution for certain type of scenarios such as indoor and city scenes which contain static objects. In environment such as outdoor or dynamic scenes and complex scenarios like massive foliage scenes, the visibility preprocessing is suffered from visibility errors and no specific solution available that provide satisfactory result in term of robustness and speed (Mattausch, 2010). Theoretically, online occlusion culling attracted by developer because it can handle dynamic scenes. Mostly online occlusion culling are notorious hardware unfriendly and rarely used in existing game designs (Staffans, 2006).

Hierarchical Z-buffer (HZB) and Hierarchical Occlusion Maps (HOM) is early online occlusion culling conceptually introduced by researchers that big influence on occlusion culling research (Moller *et al.*, 2008). The HZB is worked by Greene *et al.* (1993) using two hierarchical data structure and temporal coherence, octree for object space and z-pyramid for image space. HZB is suitable to be implemented in dynamic environment but current hardware does not support enough for reading of Z-buffer and updating of the Z-pyramid in real-time. Zhang *et al.* (1997) introduced HOM, in this algorithm the scene is subdivided using bounding volume hierarchy in preprocessing stage and require preselection of occluders which not suitable to

be applied in dynamic environment. The differences between HZB and HOM are division between occluders from occludees and add occlusion tests into depth test and overlap test. Zhang *et al.* (1997) utilized hardware texturing to propose HOM. Main problem with HOM is not updated although an object is rendered and the algorithm depend on selecting good occluder. Staffans (2006). dPVS is incremental occlusion map (IOM) the provide solution to improve HOM by extracting the silhouettes of the occluders already in objects space. The disadvantage of dPVS is added additional computation to current algorithm. HOM with software renderer improved the original and incremental HOM but occluder selection is still needed and in some cases the virtual occlusion environment usually do by hand. Wonka and Schmalstieg (1999) use occluder shadows for urban environments to perform occlusion culling.

OpenGL extension for occlusion queries firstly proposed by Bartz *et al.* (1999) before it was realized in graphic hardware. The first OpenGL extension for occlusion queries developed by Hewlett-Packard is HP_occlusion_test and initially implemented in VISUALIZE fx graphics hardware (Scott *et al.*, 1998). Two major limitation in HP_occlusion_, the queried return only a binary visibility classification value to indicate the visibility of geometry and only on queries is allowed at a time. Later, NVIDIA company introduced their extension called NV_occlusion _query and then is available as OpenGL standard extension known as ARB extension to be used in multi platform. The NV query solved the main problem of HP test by allowing multiple queries to be issued before asking their result and return number of visible pixel to know how much of the queried geometry′s visibility.

Introduction of hardware occlusion queries received attention among researchers and developers. Many algorithms started to use hardware occlusion because of its simplicity. Klosowski and Silva (2001) used hardware occlusion queries integrated with conservative visibility culling technique based on the Prioritized-Layered Projection using HP_occlusion_test extension with two-pass approach. Hillesland *et al.* (2002) used NV_occlusion_query in their framework can be called hierarchical stop and wait method. Problem with hierarchical stop and wait method lead to decrease the overall application performance cause by CPU stall and GPU starvation and overhead of occlusion queries (Wimmer and Bittner, 2005). Hardware occlusion queries then start to exploit TC concept to reduce number of queries in OpenSG framework in serial fashion (Staneker *et al.*, 2004). Implementation of TC in hardware occlusion queries later was improved by (Bittner *et al.*, 2004;

Wimmer and Bittner, 2005). The algorithm is called CHC. CHC is an algorithm set as benchmark among researcher in hardware occlusion queries problem because the algorithm is started feasible to be used in real-time rendering. Guthe *et al.* (2006) recognized problems in CHC and successfully introduced a technique called Near Optimal Hierarchical Culling (NOHC). NOHC help to reduce the number of queries based on an intelligent numerical model of occlusion and a hardware calibration (Guthe *et al.*, 2006). Other researcher introduced asynchronous occlusion queries with introduction of new concept called occupancy proportion to alleviate latency (Li *et al.*, 2008). Object occupancy proportion refers to the tightness of bounding box. Newest research that improved on CHC algorithm is Coherent Hierachical Culling Revisited (CHC++) by Bittner *et al.* (2009); Mattausch *et al.* (2008). CHC++ improves prior technique efficiently by using of temporal coherent and spatial coherence of visibility. Also, adaptive visibility prediction and query batching were introduced in CHC++ algorithm to improve previous algorithms. Another disadvantage of online occlusion culling is that it is rarely used in real engine or games. One of latest games use hardware occlusion culling in their engine is Alan Wake (Silvennoinen *et al.*, 2011). In this game, online occlusion culling involves combination between CHC and NOHC.

The potential used of online occlusion culling can be widespread in many type of virtual reality application such as driving simulator. Driving simulator environment provide a dynamic scenes where in this environment will contains moving and static objects. Based on findings, several real-time rendering techniques were implemented in car simulation. Occlusion horizon introduced for driving in urban scenery, in the algorithm built in two and half dimension combined with level of detail (Downs *et al.*, 2001). In this algorithm, only applied to static object in the environment such as building. Grundhfer *et al.* (2005) implemented level of detail based on occlusion culling for dynamic scene of driving environment. The research only focus on efficient level of detail technique for interactive scenes with arbitrary cars. Hardware occlusion queries were implemented in this algorithm to perform occlusion culling without considering the drawbacks of the queries. Based on previous research, hardware occlusion queries rarely used to handle dynamic scene such as driving environment in effectively. Implementation of hardware occlusion queries in real-time rendering must consider characteristic of the scene, the driving simulator is suitable application to implement the hardware occlusion queries.

1.3 Problem Statement

In real-time rendering, at least four goals to be achieved (Moller *et al.*, 2008). The goals are more frame rates, higher resolution and sampling rates, more realistic materials and lighting and increased complexity. 60-85 fps is a speed considered as enough frame rate (Moller *et al.*, 2008). Fast frame rate able to reduce the latency when interact with a scene. High resolution means that capability of display for example LCD TV with 1360×768 pixels can have the same achievement by graphic program in full HD TV with resolution of 1920×1080 pixels. Next, more realistic material and lighting can be achieved which means that it takes more computing power to produce the interplay of light and surface. Lastly, the complexity of virtual environment does not have limit as long as the capability of hardware is able to manage that complexity. There is no limit to decide the end of complexity. In hope to produce more realistic real-world representation, three dimensional (3D) objects are transformed into models having the illusion of depth for display onto a two-dimensional computer screen. This is accomplished by using a number of polygons to represent a three-dimensional object. The more detail and large for a 3D object, the more computation cost is needed to be rendered by one application.

Online and offline visibility culling have clear advantages and disadvantages (Bittner *et al.*, 2004; Bittner and Wonka, 2003). Based work done by Bittner *et al.* (2004), offline occlusion culling suffers from four major problems. The online solution will to help solve the problem but with additional computations for each frame. He added that to make this extra computation efficient depending on the number of assumption about the scene structure and it′s characteristics. Emergence of hardware occlusion queries faces two main problems related to naive usage of it. Overhead caused by number of issued queries and delay due to latency of the query result caused rarely used in commercial games engines(Malhotra, 2002). According to Gomez *et al.* (2011), online occlusion culling faces three problems; too slow due to latency of query result, too conservative meaning not enough objects culled and approximate visibility tend lead to produce popping effects.

Large and complex virtual environment needs better scene management because bigger scene increase the number of polygon to be rendered. Increased of polygon number will increase computational cost and will decrease the number of

fps. Varying complexity of virtual system causes the algorithm to react to achieve constant frame rates or interactive frame rates. Constant frame rates in highly varying complexity are achieved by upper bound of the rendering time (Wimmer and Wonka, 2003). In driving simulator, interactive frame rate is very critical to maintain the interactivity between user and application (Gruening *et al.*, 1998). Complex of a scene in driving simulator must be managed to maintain constant frame rate to make response time between user and the application not interrupted and at the same time maintain the realism of visual information. Realism images of rendering 3D environment is important to make sure visual information same with the real world and is very importance for driving scene.

Implementing hierarchical hardware based occlusion query in driving simulator must follow the scene structure and it′s characteristics. The problem faced by hierarchical hardware based occlusion query is that there are no automatic parameters adaptation during walkthrough as the scene requires (Mattausch *et al.*, 2008). The suitable automatic parameters adaptation are selected to help in maintaining the computation cost within time budgeted for one frame. An adaptive algorithm will be introduced by using these suitable parameters in maintaining image realism and interactivity frame rates.

1.4 Aims

The aims of this research is to introduce an adaptive algorithm for coherent hierarchical culling algorithm to be implemented in UTM car driving simulator.

1.5 Objectives

In order to arrive to this goal, there are objectives that need to be achieved, which are:

(i) To study and integrate Coherent Hierarchical Culling algorithm with existing driving simulator system.

(ii) To produce an adaptive algorithm for Coherent Hierarchical Culling algorithm based on speed of car movement and varying visibility thresholds value.

(iii) To evaluate the proposed adaptive algorithms for Coherent Hierarchical Culling algorithm for dynamics objects in driving simulator system.

1.6 Research Justification

This research is attempting to improve previous acceleration algorithm to allow it be used in dynamic scene in real-time rendering system. The natural scene of driving simulator itself which consists of large scenes and arbitrary objects made it suitable to implemented with online occlusion culling. Integration of online occlusion culling to driving simulator system requires proper solution to make sure the rendering performance and realism of the system are maintained. This integration is trying prove the improvement on online occlusion culling algorithm must figure out the suitability of the scene and application before efficiently and useful to be used in real-time rendering application. The research is aim to integrate online occlusion culling algorithm to manage dynamic objects in driving simulator system by introducing an adaptive algorithm. This adaptive algorithm is introduced in this research to provide solution for hardware based occlusion culling algorithms to maintain quality of rendering and interactivity of an application by adding new requirements based on flexibility and scalability of the scene. The research will encourage the usage of hardware based occlusion culling to be implemented in modern engine.

1.7 Scope and Research Limitations

The research focuses on managing arbitrary scene in the driving simulator. Several number of scopes and research limitations need to be considered. The system will include a non-immersive driving simulator used in this research consist of 1 pc and 1 display. The system manages dynamic and random vehicles during daylight. It does not include the building, tree, human character and other related objects. The system only uses car as dynamic and random vehicles that consist of two types of car in large outdoor scene. Two types of road used in the system as follow; straight road and hilly and curvy road. 3DS file format is used for the 3D objects. 60 fps is targeted frame rate or can be called interactive frame rate for the driving simulator application.

1.8 Thesis Organization

This thesis is divided into six chapters. Each chapter has contributed to show the flow of the research from start till the end. Chapter 1 starts with introduction of the research, then follows by elaboration of the background of the research and statement of the research problem that formulate to the idea of the research. The aims, objectives, scopes and justification for the research are explained in this chapter. Following is a brief overview of the chapters ahead.

Chapter 2 reviews related works that were previously done by other researcher. From previous research, the idea of research can be analyzed and strengthen to make sure this is researchable and contribute to the knowledge. This chapter starts with introduction to graphic pipeline, hierarchical data structure, visibility culling techniques and previous close and related research.

Chapter 3 presents the methodology in performing this research. Methodology is important to make sure all objectives set by the researcher are achieved to fulfill the aims of the research. Every steps involved in the research is discussed.

Chapter 4 discusses on the implementation of coherent hierarchical culling algorithm with proposed adaptive algorithm solution. In this chapter clearer explanation on how the development of an algorithm is performed. Step by step process of the proposed with more detail explanation compared to what has been discussed in Chapter 3.

Chapter 5 is testing of the proposed algorithm and analyse the result based on comparison with previous implementation and original CHC. In this chapter, evaluations are based on two characteristics; rendering performance and realism. The result then discussed before proceeding to the last chapter.

Chapter 6 is the last chapter which summarize all works done in the research and re-stating the contributions that have been made through the research. This chapter provides conclusion on what supposedly the directions for future research to be referred by other researcher.

REFERENCES

Adams, A. (2005). *Trucking: Tractor-Trailer Driver Handbook/Workbook*. Thomson Delmar Learning.

Aila, T. and Miettinen, V. (2004). dPVS: An Occlusion Culling System for Massive Dynamic Environments. *Computer Graphics and Applications, IEEE*. 24(2), 86–97.

Allen, R. W., Rosenthal, T. J. and Cook, M. L. (2010). A Short History of Driving Simulator. In *Handbook of Driving Simulation for Engineering, Medicine, and Psychology*. Taylor and Francis Group.

Assarsson, U. and Moller, T. A. (1999). *Optimized View Frustum Culling Algorithms*. Technical report. Department of Computer Engineering, Chalmers University of Technology.

Assarsson, U. and Moller, T. A. (2000). Optimized View Frustum Algorithms for Bounding Boxes. *Journal of Graphics Tools*. 5, 9–22.

Bartz, D., Meibner, M. and Httner, T. (1999). OpenGL-Assisted Occlusion Culling for Large Polygonal Models. In *Computers and Graphics*. 667–679.

Bittner, J., Havran, V. and Slavik, P. (1998). Hierarchical Visibility Culling with Occlusion Trees. In *Proceedings of the Computer Graphics International 1998*. IEEE Computer Society, 207.

Bittner, J., Mattausch, O. and Silvennoinen, A. (2011). Shadow Caster Culling for Efficient Shadow Mapping. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games 2011*. ACM.

Bittner, J., Mattausch, O. and Wimmer, M. (2009). Game-Engine-Friendly Occlusion Culling. In Engel, W. (Ed.) *SHADERX7: Advanced Rendering Techniques*. (pp. 637–653). vol. 7. Charles River Media.

Bittner, J., Wimmer, M., Piringer, H. and Purgathofer, W. (2004). Coherent Hierarchical Culling: Hardware Occlusion Queries Made Useful. *Computer Graphics Forum*. 23(3), 615–624.

Bittner, J. and Wonka, P. (2003). Visibility in Computer Graphics. *Environment and Planning B: Planning and Design*. 30(5), 729–755.

Blana, E. (1996). *A Survey of Driving Research Simulators Around the World*. Technical report.

Bormann, K. (2000). An Adaptive Occlusion Culling Algorithm for Use in Large VEs. In *Proceedings of the IEEE Virtual Reality 2000 Conference*. IEEE Computer Society, 290.

Burns, A. and Wellings, A. (1997). *Real-Time Systems and Programming Languages*. Addison Wesley.

Cheng, H. (2011). *Autonomous Intelligent Vehicles: Theory, Algorithms, and Implementation*. Springer.

Clark, J. (1976). Hierarchical Geometric Models for Visible Surface Algorithms. *Communications of the ACM*. 19(10), 547–554.

Cohen-Or, D., Chrysanthou, Y. L., Silva, C. T. and Durand, F. (2003). A Survey of Visibility for Walkthrough Applications. *Visualization and Computer Graphics, IEEE Transactions on*. 9(3), 412–431.

Cohen-Or, D., Fibich, G., Halperin, D. and Zadicario, E. (1998). Conservative Visibility and Strong Occlusion for Viewspace Partitioning of Densely Occluded Scenes. *Computer Graphics Forum*. 17(3), 243–253.

Coorg, S. and Teller, S. (1996). Temporally Coherent Conservative Visibility. In *Proceeding of the Twelfth Annual ACM Symposium on Computational Geometry*. 78–87.

Cremer, J., Kearney, J. and Papelis, Y. (1996). Driving Simulation: Challenges for VR Technology. *Computer Graphics and Applications, IEEE*. 16(5), 16–20.

Downs, L., Mller, T. and Sequin, C. H. (2001). Occlusion Horizons for Driving through Urban Scenery. In *Proceedings of the 2001 Symposium on Interactive 3D graphics*. ACM, 121–124.

Durand, F., Drettakis, G., Thollot, J. and Puech, C. (2000). Conservative Visibility Preprocessing using Extended Projections. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 239–248.

Ferwerda, J. A. (2003). Three varieties of realism in computer graphics, 290–297. 10.1117/12.473899.

Fisher, D. L., Caird, J. K., Rizzo, M. and Lee, J. D. (2010). Handbook of Driving Simulation : An Interview. In *Handbook of Driving Simulation for Engineering,*

Medicine, and Psychology. Taylor and Francis Group.

Foley, J., Dam, A. V., Feiner, S. and Hughes, J. (1990). *Computer Graphics: Principles and Practice, Second Edition*. Addison-Wesley Professional.

Fuchs, H., Kedem, Z. M. and Naylor, B. F. (1980). On Visible Surface Generation by a Priori Tree Structures. *SIGGRAPH Computer Graphics*. 14(3), 124–133.

Gettman, D. and Head, L. (2003). Surrogate Safety Measures from Traffic Simulation Models. *Transportation Research Record: Journal of the Transportation Research Board*. 1840(-1), 104–115.

Gilbert, D. A. (1994). *Rendering Systems for Virtual Reality Applications*. Ph.D. Thesis. University of Mancester.

Gobbetti, E., Kasik, D. and Yoon, S.-e. (2008). Technical Strategies for Massive Model Visualization. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*. ACM, 405–415.

Gomez, D., Poulin, P. and Paulin, M. (2011). Occlusion Tiling. In *Proceedings of Graphics Interface 2011*. Canadian Human-Computer Communications Society, 71–78.

Govindaraju, N., Sud, A., Yoon, S.-E. and Manocha, D. (2003). Interactive Visibility Culling in Complex Environments using Occlusion-Switches. In *Proceedings of the 2003 symposium on Interactive 3D Graphics*. ACM, 103–112.

Greene, N., Kass, M. and Miller, G. (1993). Hierarchical Z-buffer Visibility. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*. ACM, 231–238.

Gruening, J., Bernard, J., Clover, C. and Hoffmeister, K. (1998). Driving Simulation. In *SAE Technical Paper 980223*.

Grundhfer, A., Brombach, B., Scheibe, R. and Frhlich, B. (2005). Level of Detail based Occlusion Culling for Dynamic Scenes. In *Proceedings of the 3rd International Conference on Computer graphics and Interactive Techniques in Australasia and South East Asia*. ACM, 37–45.

Guthe, M., Balazs, A. and Klein, R. (2006). Near Optimal Hierarchical Culling: Performance Driven Use of Hardware Occlusion Queries. In Heidrich, T. A.-M. and Wolfgang (Eds.) *Proceedings of Eurographics Symposium on Rendering 2006*. The Eurographics Association, 207–214.

Havran, V. (2000). *Heuristic Ray Shooting Algorithms*. Ph.D. Thesis. Czech Technical University.

Hillesland, K., Salamon, B., Lastra, A. and Manocha, D. (2002). *Fast and Simple*

Occlusion Culling using Hardware-Based Depth Queries. Technical report. University of North Carolina.

Hudson, T. C., Manocha, D., Cohen, J. D., Lin, M. C., Hoff III, K. E. and Zhang, H. (1997). Accelerated Occlusion Culling using Shadow Frusta. In *Proceedings of the thirteenth annual symposium on Computational geometry*. ACM, 1–10.

James, G., James, B., Chris, C. and Kurt, H. (1998). Driving Simulation. In *SAE Special Publications, v 1361, Feb, 1998, 980223, Vehicle Dynamics and Simulation*. (pp. 49–59). CiteSeerX - Scientific Literature Digital Library and Search Engine [http://citeseerx.ist.psu.edu/oai2] (United States) ER.

Jia, L., Han, S., Wang, L. and Liu, H. (2007). Development and Realization of a Street Driving Simulator for Virtual Tour. In *Proceedings of the 40th Annual Simulation Symposium*. IEEE Computer Society, 133–136.

Kang, H. S., Abdul Jalil, M. K. and Mailah, M. (2004). A PC-based Driving Simulator using Virtual Reality Technology. In *Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry*. ACM, 273–277.

Klosowski, J. T. and Silva, C. T. (2001). Efficient Conservative Visibility Culling using the Prioritized-Layered Projection Algorithm. *IEEE Transactions on Visualization and Computer Graphics*. 7(4), 365–379.

Koltun, V., Chrysanthou, Y. and Or, D. C. (2000). Virtual Occluders: An Efficient Intermediate PVS Representation. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*. Springer, 59–70.

Kovalcik, V. and Sochor, J. (2005). Occlusion Culling with Statistically Optimized Occlusion Queries. In *Proceedings of Winter School of Computer Graphics (Short Papers)*. 109–112.

Kumar, S., Manocha, D., Garrett, W. and Lin, M. (1996). Hierarchical Back-Face Computation. In *Proceedings of the Eurographics Workshop on Rendering Techniques '96*. Springer-Verlag, 235–ff.

Lebram, M., Engstrom, H. and Gustavsso, H. (2006). A Driving Simulator based on Video Game Technology. In *In Proceedings of SIGRAD 2006*.

Lengyel, E. (2003). *Mathematics for 3D Game Programming and Computer Graphics, Second Edition*. Charles River Media, Inc.

Li, B., Wang, C. and Li, L. (2008). Efficient Occlusion Culling with Occupancy Proportion. In *International Conference on Computer Science and Software Engineering, 2008*, vol. 2. 1058–1061.

Liao, D. (2006). A Real-time High-fidelity Driving Simulator System Based on PC Clusters. In *Proceedings of the 11th IEEE International Conference on Engineering of Complex Computer Systems*. IEEE Computer Society, 209–216.

Lloyd, B. and Egbert, P. (2002). Horizon Occlusion Culling for Real-Time Rendering of Hierarchical Terrains. In *Proceedings of the conference on Visualization '02*. VIS '02. 403–409.

Luebke, D., Watson, B., Cohen, J. D., Reddy, M. and Varshney, A. (2002). *Level of Detail for 3D Graphics*. Elsevier Science Inc.

Malhotra, P. (2002). *Issues involved in Real-Time Rendering of Virtual Environments*. Master's thesis. College of Architecture and Urban Studies.

Mattausch, O. (2010). *Visibility Computations for Real-Time Rendering in General 3D Environments*. Dissertation. Vienna University of Technology.

Mattausch, O., Bittner, J. and Wimmer, M. (2008). CHC++: Coherent Hierarchical Culling Revisited. *Computer Graphics Forum (Proceedings Eurographics 2008)*. 27(2), 221–230.

Moller, T. A., Haines, E. and Hoffman, N. (2008). *Real-Time Rendering 3rd Edition*. A. K. Peters, Ltd.

Roberts, L. G. (1963). *Machine Perception of Three-Dimensional Solids*. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York.

Saona-Vazquez, C., Navazo, I. and Brunet, P. (1999). The Visibility Octree. A Data Structure for 3D Navigation. *Computers and Graphics*. 23(5), 635–643.

Schaufler, G., Dorsey, J., Decoret, X. and Sillion, F. X. (2000). Conservative Volumetric Visibility with Occluder Fusion. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 229–238.

Scherzer, D., Yang, L. and Mattausch, O. (2010). Exploiting Temporal Coherence in Real-Time Rendering. In *ACM SIGGRAPH ASIA 2010 Courses*. ACM, 1–26.

Scherzer, D., Yang, L., Mattausch, O., Nehab, D., V. Sander, P., Wimmer, M. and Eisemann, E. (2011). A Survey on Temporal Coherence Methods in Real-Time Rendering. In *State of the Art Reports Eurographics*. Eurographics State of the Art Report.

Scott, N. D., Olsen, D. M. and Gannet, E. W. (1998). An Overview of the Visualize fx Graphics Accelerator Hardware. *Hewlett Packard Journal*, 28–34.

Sekulic, D. (2004). Efficient Occlusion Culling. In Pharr, M. and Fernando, R. (Eds.) *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*.

Pearson Higher Education.

Silvennoinen, A., Soininen, T., Mki, M. and Tervo, O. (2011). Occlusion Culling in Alan Wake. In *ACM SIGGRAPH 2011 Talks*. ACM, 1–1.

Slob, J. J. (2008). *State-of-the-Art Driving Simulators, a Literature Survey*. Technical report. Eindhoven University of Technology.

Smith, R. (2006). *Open Dynamics Engine V0.5 User Guide [Online]*. Retrievable at `http://www.ode.org/ode-latest-userguide.html`.

Staffans, J. (2006). *Online Occlusion Culling*. Ph.D. Thesis. Abo Akademi.

Staneker, D., Bartz, D. and Straer, W. (2004). Occlusion Culling in OpenSG PLUS. *Computers and Graphics*. 28(1), 87–92.

Sun, C., Xie, F., Feng, X., Zhang, M. and Pan, Z. (2007). A Training Oriented Driving Simulator. In Ma, L., Rauterberg, M. and Nakatsu, R. (Eds.) *Entertainment Computing ICEC 2007*. (pp. 1–9). *Lecture Notes in Computer Science*, vol. 4740. Springer Berlin / Heidelberg.

Sunar, M. S., Abdullah, M. Z. and Sembok, T. M. T. (2006a). Effective Range Detection Approach for Ancient Malacca Virtual Walkthrough. *The International Journal of Virtual Reality*. 5(4), 31–38.

Sunar, M. S., Azhar, M. A. M., Mokhtar, M. K. and Daman, D. (2009). Crowd Rendering Optimization for Virtual Heritage System. *The International Journal of Virtual Reality*. 8(3), 57–62.

Sunar, M. S., Sembok, T. M. T. and Zin, A. M. (2006b). Accelerating Virtual Walkthrough with Visual Culling Techniques. In *International Conference on Computing and Informatics, 2006*. 1–5.

Sunar, M. S., Zin, A. M. and Sembok, T. M. T. (2006c). Range Detection Approach in Interactive Virtual Heritage Walkthrough. In *Proceedings of the 16th International Conference on Artificial Reality and Telexistence-Workshops*. IEEE Computer Society, 599–602.

Sung, L. W., Ha, K. J. and Hee, C. J. (1998). A Driving Simulator as a Virtual Reality Tool. In *Proceedings in IEEE International Conference on Robotics and Automation*, vol. 1. 71–76.

Teller, S. J. (1992). *Visibility Computations in Densely Occluded Polyhedral Environments*. Technical report. University of California.

Vehicles, F. D. o. H. S. and Motor (2006). *Florida CDL Handbook 2006*. Florida Department of Highway Safety and Motor Vehicles.

Whyte, J. (2002). *Virtual Reality and the Built Environment*. Architectural Press.

Wimmer, M. (2001). *Representing and Rendering Distant Objects for Real-Time Visualization*. Ph.D. Thesis. Vienna University of Technology.

Wimmer, M. and Bittner, J. (2005). Hardware Occlusion Queries Made Useful. In Pharr, M. and Fernando, R. (Eds.) *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*. (pp. 91–108). Addison-Wesley.

Wimmer, M. and Wonka, P. (2003). Rendering Time Estimation for Real-time Rendering. In *Proceedings of the 14th Eurographics Workshop on Rendering*. Eurographics Association, 118–129.

Wonka, P. and Schmalstieg, D. (1999). Occluder Shadows for Fast Walkthroughs of Urban Environments. *Computer Graphics Forum*. 18(3), 51–60.

Wonka, P., Wimmer, M. and Schmalstieg, D. (2000). Visibility Preprocessing with Occluder Fusion for Urban Walkthroughs. In *Rendering Techniques 2000 (Proceedings Eurographics Workshop on Rendering)*. Springer-Verlag Wien New York, 71–82.

Yoshimoto, K. and Suetomi, T. (2008). The History of Research and Development of Driving Simulators in Japan. *Journal of Mechanical Systems for Transportation and Logistics*. 1(2), 159–169.

Young, V. (2004). *Programming a Multiplayer FPS in DirectX*. Charles River Media.

Zamri, M. N. (2006). *An Efficient Real-Time Terrain Data Organization and Visualization Algorithm based on Enhanced Triangle-Based Level of Detail Technique*. Master's thesis. Universiti Teknologi Malaysia.

Zerbst, S. (2004). *3D Game Engine Programming (Game Development Series)*. Premier Press.

Zhang, H., Manocha, D., Hudson, T. and Hoff III, K. E. (1997). Visibility Culling using Hierarchical Occlusion Maps. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 77–88.