

การเพิ่มสมรรถนะของบิลบอร์ดเพื่อการแทนวัตถุประสงค์สามมิติรายละเอียดสูง



นายพงษ์วาริน วิจิตรเวชไพศาล

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2548

ISBN 974-17-3429-8

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

AN ENHANCEMENT OF BILLBOARD FOR HIGHLY DETAILED 3D OBJECT
REPRESENTATION

Mr. Phongvarin Vichitvejpaisal



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Computer Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2005

ISBN 974-17-3429-8

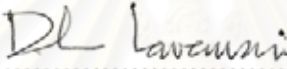
Thesis Title An Enhancement of Billboard for Highly Detailed 3D Object Representation

By Mr. Phongvarin Vichitvejpaisal

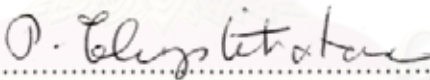
Field of Study Computer Engineering

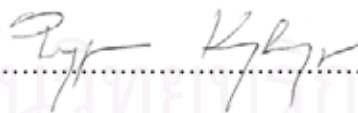
Thesis Advisor Pizzanu Kanongchaiyos, Ph.D.


Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

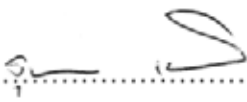

..... Dean of the Faculty of Engineering
(Professor Direk Lavansiri, Ph.D.)

THESIS COMMITTEE


..... Chairman
(Associate Professor Prabhas Chongstitvatana, Ph.D.)


..... Thesis Advisor
(Pizzanu Kanongchaiyos, Ph.D.)


..... Member
(Vishnu Kotrajaras, Ph.D.)


..... Member
(Supatana Auethavekiat, Ph.D.)

พงษ์วาริน วิจิตรเวชไพศาล : การเพิ่มสมรรถนะของบิลบอร์ดเพื่อการแทนวัตถุสามมิติ
รายละเอียดสูง (AN ENHANCEMENT OF BILLBOARD FOR HIGHLY DETAILED
3D OBJECT REPRESENTATION) อาจารย์ที่ปรึกษา: ดร.พิชญ์ คนองชัยยศ, 70 หน้า.
ISBN 974-17-3429-8

กลุ่มรูปภาพบิลบอร์ดสามารถนำมาใช้แสดงวัตถุสามมิติสำหรับการลดทอนรายละเอียด
ในการแสดงผลภาพแบบทันที การใช้วิธีแบบรูปภาพบิลบอร์ดจะแตกต่างจากวิธีใช้โพลีกอนตรง
ที่ว่า วิธีแบบรูปภาพบิลบอร์ดใช้เวลาในการแสดงผลภาพเป็นอัตราส่วนผันแปรกับความละเอียด
ของภาพที่รูปภาพบิลบอร์ดแสดง ไม่เหมือนกับวิธีโพลีกอนซึ่งโพลีกอนนั้นมีเวลาที่ใช้ในการแสดง
ภาพคงที่ ไม่ขึ้นกับความละเอียดของรูปภาพ ทำให้วิธีแบบรูปภาพบิลบอร์ดสามารถใช้กับงานที่มี
หลายระดับความละเอียด อย่างไรก็ตามวิธีรูปภาพบิลบอร์ดแบบก่อนๆ มีมุมมองในการแสดงผลที่
จำกัด และไม่สามารถ แสดงวัตถุสามมิติได้โดยละเอียด งานวิจัยชิ้นนี้ ได้นำเสนอ การปรับปรุง
เพิ่มสมรรถนะของบิลบอร์ด เพื่อใช้แสดงวัตถุสามมิติที่มีรายละเอียดสูง ชั้นแรก รูปภาพบิลบอร์ด
แต่ละอัน ที่ใช้แทนบางส่วนของวัตถุสามมิตินั้น จะประกอบไปด้วยภาพ สองมิติ สีภาพ อันได้แก่
ภาพแสดงความลึก ภาพนอร์มอล ภาพสี ภาพความโปร่งใส โดยวัตถุสามมิติจะถูก โปรเจคไปยัง
ระนาบมุมมองในหลายๆ ทิศทาง และเก็บเป็นแบบรูปภาพบิลบอร์ดที่เพิ่มสมรรถนะ ต่อมารูปภาพ
บิลบอร์ดแบบเพิ่มสมรรถนะจะถูกแสดงผลด้วยวิธีการตรวจสอบการชนแบบ เรย์ไทร์ฟิลด์อินเตอร์
เซกชัน โดยประมวลผลในแผ่นวงจรรกราฟิค การจัดเก็บวัตถุสามมิติด้วยวิธีนี้ สามารถเก็บรูปทรง
และเส้นขอบของวัตถุโดยสามารถแสดงผลได้ทุกมุมมอง และยังแสดงผลได้ในเวลาทันที

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา.... วิศวกรรมคอมพิวเตอร์.....ลายมือชื่อนิสิต.....
สาขาวิชา....วิศวกรรมคอมพิวเตอร์.....ลายมือชื่ออาจารย์ที่ปรึกษา.....
ปีการศึกษา2548.....ลายมือชื่ออาจารย์ที่ปรึกษาร่วม.....

4770366021 : MAJOR COMPUTER ENGINEERING

KEY WORD: IMAGE-BASED RENDERING / DISPLACEMENT / BILLBOARDS / REAL-TIME RENDERING / MODEL SIMPLIFICATION

PHONGVARIN VICHITVEJPAISAL: AN ENHANCEMENT OF BILLBOARD FOR HIGHLY DETAILED 3D OBJECT REPRESENTATION. THESIS ADVISOR: PIZZANU KANONGCHAIYOS, Ph.D., 70 pp. ISBN 974-17-3429-8

A set of billboards can represent 3D models for simplification in real-time rendering. Unlike conventional polygon methods, a billboard based technique has the rendering time of the model proportional to its contribution of data to the image. Thus, it has a Level of detail property. However, previous techniques still have limited viewing angle and do not accurately represent the model. This research presents an adaptive rendering method using enhanced billboards. First, each enhanced billboard, representing a portion of the model, is defined to have four maps: depth map, normal map, color map and transparency map. The model is then projected onto a number of viewing planes in different viewing directions. These enhanced billboards are rendered based on ray-height-field intersection algorithm implemented on graphics processing unit (GPU). This representation can maintain the geometry and the silhouette of the model with no limit in viewing direction. Moreover, real-time rendering is supported.

Department.... Computer Engineering.... Student's signature.....*Phongvarin Vichitvejpaisal*
 Field of study.... Computer Engineering...Advisor's signature.....*Pizanu Kanongchaiyos*
 Academic year ...2005.....Co-advisor's signature.....

ACKNOWLEDGMENTS

It is a great pleasure to acknowledge my thesis advisor, Pizzanu Kanongchaiyos, Ph.D., for his intellectual advices and invariable assistances throughout this research. I would also like to express my grateful thanks to my thesis committee, Associate Professor Prabhas Chongstitvatana, Ph.D, Vishnu Kotrajaras, Ph.D., Supatana Auethavekiat, Ph.D. for their beneficial guidance and suggestions.

I want to extend my thanks to all research members especially my associates in computer graphic lab (CG Lab) for their generous helps, encouragements and relationships which make my life through the course filled with amusements and happiness.

Finally, I deeply wish to thank my parents for their love, understanding and invaluable supports throughout my graduate study.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

TABLE OF CONTENTS

	Page
ABSTRACT (THAI)	iv
ABSTRACT (ENGLISH)	v
ACKNOWLEDGMENTS	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	ix
LIST OF TABLES	xi
CHAPTER 1 INTRODUCTION	1
1.1 Background and Statement of Problems	1
1.2 Objectives	3
1.3 Scope of Study	3
1.4 Research Procedure	3
1.5 Expected Benefits	4
1.6 Thesis Structure	4
1.7 Publication	4
CHAPTER 2 THEORETICAL BACKGROUND AND RELATED WORKS	6
2.1 Theoretical Background	6
2.1.1 Basic Local Illumination Shading	6
2.1.2 Ray-Casting Algorithm	7
2.1.3 Ray-height-field Algorithm	7
2.1.4 Programmable Graphics Pipeline	8
2.1.5 Image Comparison	9

	Page
2.2 Related Works	10
2.2.1 Mesh Simplification.....	10
2.2.2 Bump Map.....	10
2.2.3 Displacement Map.....	11
2.2.4 General Displacement Map.....	12
2.2.5 Billboard Cloud.....	12
CHAPTER 3 PROPOSED ENHANCED BILLBOARDS	14
3.1 Enhanced Billboards	14
3.2 Building Enhanced Billboards	15
3.3 Rendering Enhanced Billboards	17
3.4 Solving Holes Artifact	19
CHAPTER 4 EXPERIMENTAL RESULTS	22
4.1 Testing Environment.....	22
4.2 Experiment and Results	23
4.3 Discussion	26
CHAPTER 5 CONCLUSION AND FUTURE WORK	28
5.1 Conclusion.....	28
5.2 Future Work	29
APPENDIX.....	30
APPENDIX A.....	31
APPENDIX B	37
REFERENCES	68
BIOGRAPHY	70

LIST OF FIGURES

Figure	Page
1-1 Billboard of a tree.....	2
1-2 Horse model. (1) An enhanced billboard. (2) Front part of horse model. (3) Complete horse model.....	3
2-1 The diagram of shading	6
2-2 The ray-casting algorithm	7
2-3 A Depth map of a horse model	8
2-4 The hardware graphics pipeline.....	8
2-5 Progressive mesh.....	10
2-6 Bump map. (1) Normal map. (2) Bump map model	11
2-7 Relief mapping.....	12
2-8 General displacement map.....	12
2-9 Forest represented with a group of billboards	13
2-10 Billboards Cloud.....	13
3-1 The first five enhanced billboards of the horse model	15
3-2 The projection cameras around the horse model	17
3-3 The ray-casting diagram	18
3-4 Pseudo-code of the algorithm for rendering enhanced billboards	18
3-5 Horse billboard. (1) Horse model rendered with one size. (2) Horse model rendered with all three size.....	19
3-6 The holes artifact	20
3-7 Hit range threshold. (1) Hit range threshold $e = 0.01$. (2) Hit range threshold $e = 0.02$	21
3-8 Enhanced billboards with coarse mesh.....	21
4-1 The enhanced billboards of the horse model	23
4-2 The horse model rendered using enhanced billboards	24
4-3 The storage size of original polygon models compare to enhance billboards...24	
4-4 The greedy projection of the dragon model.....	25
4-5 The axis projection of the dragon model.....	25
4-6 The render speed of all the models (with and without coarse mesh) in 800x600 resolution.....	26

Figure	Page
A-1.1 Cylinder model.....	38
A-1.2 Cylinder model with different resolution	38
A-2.1 Dolphin model.....	39
A-2.2 Dolphin model with different resolution	39
A-3.1 Horse model.....	40
A-3.2 Horse model with different resolution	40
A-4.1 Dragon model.....	41
A-4.2 Dragon model with different resolution	41
A-5.1 Moai model.....	42
A-5.2 Moai model with different resolution	42
A-6.1 Sphere model.....	43
A-6.2 Sphere model with different resolution	43
A-7.1 Teapot model.....	44
A-7.2 Teapot model with different resolution	44
A-8.1 Torus model.....	45
A-8.2 Torus model with different resolution	45
A-9.1 Woman model.....	46
A-9.2 Woman model with different resolution	46
A-10.1 Cone model.....	47
A-10.2 Cone model with different resolution	47
A-11.1 Face model.....	48
A-11.2 Face model with different resolution	48
A-12.1 Hand model.....	49
A-12.2 Hand model with different resolution	49
A-13.1 Head model.....	50
A-13.2 Head model with different resolution	50
A-14.1 Face2 model.....	51
A-14.2 Face2 model with different resolution	51
A-15.1 Man model.....	52
A-15.2 Man model with different resolution	52
A-16.1 Hand2 model.....	53
A-16.2 Hand2 model with different resolution	53

Figure	Page
A-17.1 Shape1 model.....	54
A-17.2 Shape1 model with different resolution	54
A-18.1 Shape2 model.....	55
A-18.2 Shape2 model with different resolution	55
A-19.1 Shape3 model.....	56
A-19.2 Shape3 model with different resolution	56
A-20.1 Shape4 model.....	57
A-20.2 Shape4 model with different resolution	57
A-21.1 Shape5 model.....	58
A-21.2 Shape5 model with different resolution	58
A-22.1 She model.....	59
A-22.2 She model with different resolution	59
A-23.1 Tutan model.....	60
A-23.2 Tutan model with different resolution	60
A-24.1 Face3 model.....	61
A-24.2 Face3 model with different resolution	61
A-25.1 Leg model.....	62
A-25.2 Leg model with different resolution	62
A-26.1 Tail model.....	63
A-26.2 Tail model with different resolution	63
A-27.1 Monster model.....	64
A-27.2 Monster model with different resolution	64
A-28.1 Body model.....	65
A-28.2 Body model with different resolution	65
A-29.1 Boat model.....	66
A-29.2 Boat model with different resolution	66
A-30.1 Fan model.....	67
A-30.2 Fan model with different resolution	67

LIST OF TABLES

Table	Page
A-1 The rendering time of the enhanced billboards with three different resolutions.....	32
A-2 The average intensity difference of the enhanced billboards and the original polygon mesh	35



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER 1

INTRODUCTION

1.1 Background and Statement of Problems

One of the crucial problems in computer graphics is how to handle an overwhelming complexity of the scene. A highly detailed scene can give more attractive and realistic look but lower performance.

Normally, polygon representation is used in rendering a 3D model. A typical complex scene might contain millions of polygons. In addition, graphic applications such as games usually require high rendering performance as well as real-time frame rate. As a result, the need of reality in the computer-generated image field is increasing rapidly.

The computational time for rendering an image represented by polygons is proportional to the number of polygons used. However, a creator still puts more polygons in a scene owing to the advancement of graphics hardware. The more polygons, the more computational time is required to capture the details of 3D image. However, pixels may play an important role in this part of innovation.

The number of pixels used in rendering an image changes very little for the last ten years. Generally, to render a computer-generated image with good quality occupies the resolution of 1024x768 pixels. In addition, this resolution cannot be increased, since a human has a limitation of visual perception. In fact, if the computational time depends on the resolution, the rendering speed will be promising.

Many techniques have been invented to manage the tradeoff between speed and quality of rendered images. Most techniques propose either new model representations or new rendering algorithms.

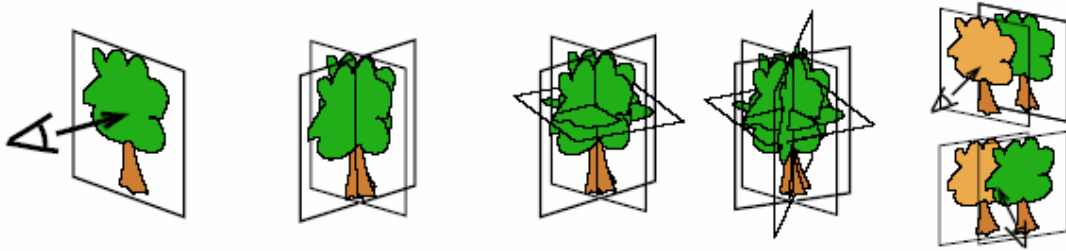


Figure 1-1: Billboard of a tree [1].

Billboard is one of the various image-based model simplification methods, which have advantages of simplicity and fast rendering speed. Typically, one billboard represents one 3D model as shown in Figure 1-1. The image of billboard is placed over the position of a model and is rendered as a texture mapped square-shaped polygon. Usually, a billboard is placed far away from viewers. The image of billboard is precomputed. Some billboards can generate their images by re-projecting original 3D models at any point in time. The image-based property of the billboard enables linear rendering time with its screen contribution. However, this technique has some limitations. Billboard cannot represent all geometric structure of a 3D model. For example, a user has the limitation of viewing direction since only 2D image is presented.

Polygon technique, on the other hand, uses triangles to approximate 3D geometric models. Triangular shape is also an easy, simple form for computation. Thus, polygon representation method is versatile. As a result, most graphic hardware in the market supports this method. Although, hardware helps to accelerate polygon rendering, the time spent for such process depends the number of polygons. Therefore, the polygon representation method consumes a constant rendering time independent of the image resolution.

The goal of this research is to find a 3D model representation and rendering method, which computation time is linearly proportional to the screen resolution. This research proposes how billboard can be enhanced to produce high quality images. The new representation for the model, called *enhanced billboards*, is a series of images. Each image consists of a color map, a transparency map, a normal map and a depth map, or displacement map [2]. The images of enhanced billboards are projected from different directions of the model. An example of enhanced billboards is shown in Figure 1-2. Previous simplification techniques for generating billboard

typically stored only some parts of the source model, and therefore, much of the information was lost. A new ray-height-field intersection algorithm to render the enhanced billboards is developed to achieve this goal.

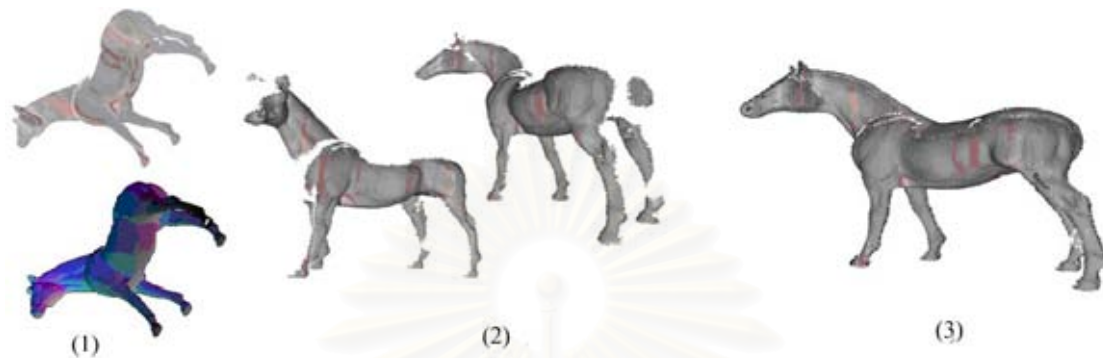


Figure 1-2: Horse model. (1) An enhanced billboard. (2) Front part of horse model. (3) Complete horse model.

1.2 Objectives

The objective of this study is to present an enhancement of billboard for highly detailed 3D object that can be rendered in real-time. The method can render silhouette and self-occlusion of the models correctly with correct lighting and shadowing. Moreover, this method has Level of detail characteristic. Thus, there is no need to use different versions of model resolution in rendering.

1.3 Scope of Study

1. The system is 2.8 GHz PC with 512 MB of memory, using a GeForce FX6800 with 128 MB of memory or higher.
2. This method can be used with arbitrary polygonal models (> 10,000 polygons).
3. This method can render in real-time (> 24fps).
4. This method use GPU that support Shader model 3.0 or later.

1.4 Research Procedure

1. Review literature

Related-theories

- The Fundamental Graphics theories
- The Surface mapping theories

- The Rendering theories

Previous research studies, articles, books, online information technology.

2. Design an algorithm.
3. Program development
4. Monitoring and evaluation
5. Implementation
6. Analysis
7. Deliverables and conclusion

1.5 Expected Benefits

1. This method can render complex 3D models in real-time.
2. This method can render models with correct silhouette and surface self-occlusion in any viewing directions.
3. This method can be used for real-time application such as games and virtual reality.

1.6 Thesis Structure

This thesis has six chapters - introduction, theory, related works, enhanced billboards algorithms and implementation, experimentation and results as well as conclusion.

The first chapter provides rationale background, objectives, scope, research procedure, benefits, research structure and publications. Chapter 2 gives a brief description of the rendering theories, the ray-height field intersection algorithm and the graphics hardware pipeline. Chapter 3 discusses on previous works regarding 3D model representation and rendering. In chapter 4, the proposed enhanced billboards algorithm is presented with implementation detail. Chapter 5 shows the experimental results. The last chapter is the conclusion.

1.7 Publications

Some parts of this research had been published in the international conference proceedings, the 14-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2006 (WSCG 2006) which was held on January 29 – February 1, 2006, Plzen, Czech Republic. The paper title is Enhanced

Billboard for Model Simplification. The authors are Phongvarin Vichitvejpaisal and Pizzanu Kanongchiayos.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER 2

THEORETICAL BACKGROUND AND RELATED WORKS

This chapter reviews some fundamental theories in computer graphics rendering. The basic local illumination shading algorithm, ray casting algorithm and ray-height-field algorithm, as well as programmable graphics pipeline of the hardware are mentioned. The related works are in the field of mesh simplification. The mapping techniques used to increase the model details and billboard improvement methods are also presented.

2.1 Theoretical Background

2.1.1 Basic Local Illumination Shading

In order to render an image, at least three information of the object are needed, which are object position, object surface property and normal vector of the object surface. For each point on the diffuse surface, the light intensity is computed using the formula (2.1). The diagram of shading is shown in Figure 2-1 [3].

$$L(\omega_r) = k_d (n \cdot l) \quad (2.1)$$

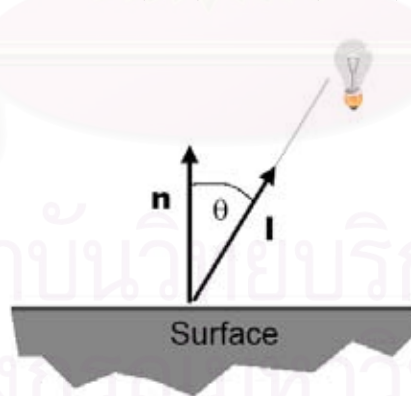


Figure 2-1: The diagram of shading.

$L(\omega_r)$ is the light intensity that is seen by the user in direction ω_r .

k_d is the diffuse reflectance property.

n is the normal vector of the surface.

l is the light vector.

Light intensity for the diffuse surface depends on the angle between the normal vector and the light vector.

2.1.2 Ray-Casting Algorithm

Ray-casting algorithm [3] is used in rendering images. In rasterization, polygons are projected from a scene to screen pixels, while ray-casting algorithm cast ray from each pixel, to find intersection point with the polygons in the scenes. That intersection point is rendered on screen. The algorithm is as follows:

1. For each pixel
 - a. Cast ray into the scenes
 - b. Find the intersection of ray and polygons in the scenes
 - c. Compute color using shading algorithm

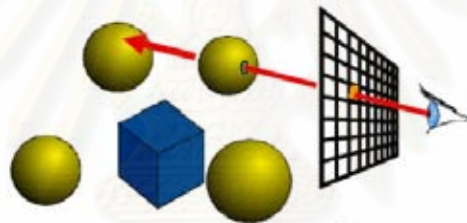


Figure 2-2: The ray-casting algorithm [3].

2.1.3 Ray-height-field Algorithm

Ray-height-field algorithm [3] uses ray-casting algorithm to render the models that are represented as height maps or depth maps. Height map stores the height information as gray scale image. This method casts ray to find intersection with the height map. The algorithm is as follow

1. For each pixel
 - a. Cast ray to find the intersection with the height map
 - b. While ray does not hit the height map
 - i. Linearly stepping the ray in the viewing direction
 - c. Compute color using shading algorithm

The algorithm performs the ray intersection test with the height map using linear step search. If the step is too large, it is possible that the algorithm may miss

the details in the height map. On the other hand, if the step is too small, rendering the height map may take very long.

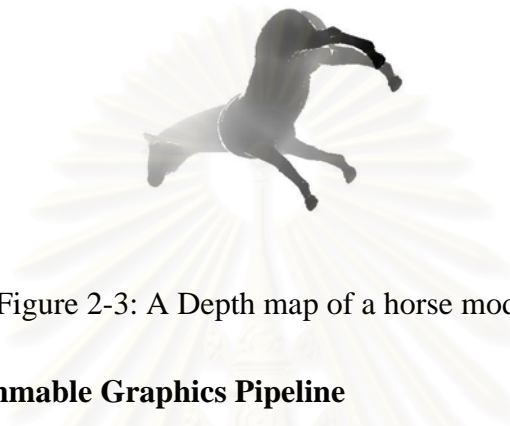


Figure 2-3: A Depth map of a horse model.

2.1.4 Programmable Graphics Pipeline

Recently, Graphics processing unit (GPU) [4] allows users to program its graphics pipeline. Figure 2-4 shows an overview of the graphics pipeline. First, the vertices are the input of the pipeline. These vertices are then assembled and interpolated. GPU allows its user to program 2 parts of the pipeline by Vertex shader and Pixel shader. Vertex shader handles incoming vertices. It can manipulate all the properties of each vertex, position, color, normal, texture coordinate and transformation. The modified vertices are then assembled and interpolated, resulting in fragments, which are passed to the Pixel shader. The Pixel shader manipulates the interpolated fragments. The fragments are tested with a depth test and a stencil test before being rendered on screen as pixels.

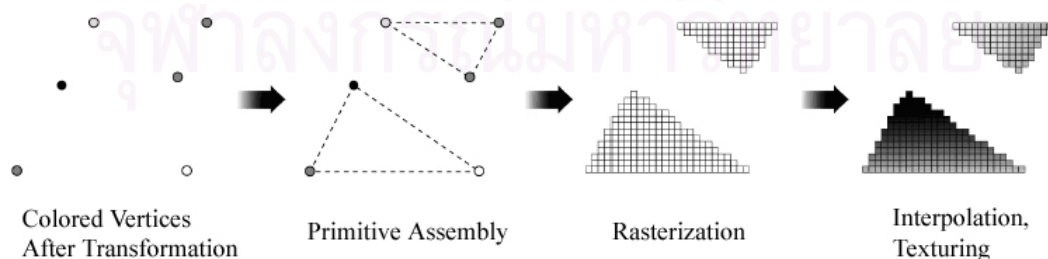


Figure 2-4: The hardware graphics pipeline [4].

2.1.5 Image Comparison

Image comparison [5] is an error metric used to measure how two images are different from each other. There are many image comparison techniques. Some methods are based on comparing color of pixels one by one. The others also use distance between pixels in the computation. Usually, all the pixels are put into the computation. The output is the average intensity difference of the two images.

This research uses pixel-by-pixel comparison basis. It is straightforward to tell the difference between two images by looking at the number of different pixels on the images and the amount of intensity difference between corresponding pixels. Each pixel in the two images is tested to find the color difference between each other. The comparison is performed in CIE Luv color model because this model can compute the intensity difference. If the color is stored in RGB color format, it has to be converted to CIE Luv format. The formula for finding the average intensity difference between two images is as follow.

$$\text{Average intensity difference} = \frac{\sum_i^{\text{pixels}} \sqrt{(L_{1i} - L_{2i})^2 + (u_{1i} - u_{2i})^2 + (v_{1i} - v_{2i})^2}}{\text{pixels}} \quad (2.2)$$

pixels is the number of the object pixels in the image.

L_{1i} is the L component of the pixel i in the first image.

L_{2i} is the L component of the pixel i in the second image.

u_{1i} is the u component of the pixel i in the first image.

u_{2i} is the u component of the pixel i in the second image.

v_{1i} is the v component of the pixel i in the first image.

v_{2i} is the v component of the pixel i in the second image.

The average intensity difference has the range between 0 to 580. The value 580 is the difference between the red color and the blue color. These two colors are the most distinct colors.

2.2 Related Works

2.2.1 Mesh Simplification

Some example methods for rendering highly detailed objects with fewer polygons are mesh simplification [6-7]. These techniques reduce the number of polygons rendered on screen. Numbers of polygons are reduced by edge collapsing or vertex removing algorithm. Thus, the quality of the model is lower than the original. The results may be rough models with discontinuity. Some of the key visual details such as the nose of a face can be lost. These techniques depend on the property of the input model, such as topology or connectivity. Thus, different algorithm has to be used for different style of the model. Another drawback is that these methods have to store many resolution versions of the model. These will increase the cost of storage and introduce pop-up artifact when changing the resolution of the model. Progressive mesh [8, 9], shown in Figure 2-5, is invented to solve the pop-up problem. It has an auxiliary data structure to smoothly add and collapse vertex of an object. Thus, it can continuously increase or decrease the quality of the model unnoticeable.



Figure 2-5: Progressive mesh [8].

2.2.2 Bump Map

Bump map [10-12] is one of the first methods for adding the bumpy look to surfaces. Bump map uses the same amount of polygon, however, it alters the normal property of every points on the surface. Bump map uses normal map to change the original surface normal. The changed normal is then used in lighting calculation. Figure 2-6 (1) shows a 3D encoded normal map. The X, Y and Z components map to the R, G and B channel of the image respectively. Figure 2-6 (2) shows rendered image using bump map technique, which uses only two polygons. The image has correct lighting. However, its geometry still looks flat. Therefore, the same plane of

surface with different normal is seen as different surface shape. Unfortunately, it cannot produce self-occlusion of the surface nor correct silhouette.

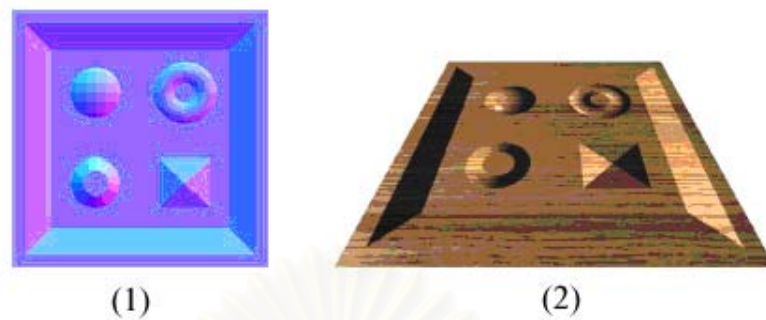


Figure 2-6: Bump map. (1) Normal map. (2) Bump map model [13].

2.2.3 Displacement Mapping

Displacement mapping [14-17] actually perturbs the surface position. Nevertheless, it uses high amount of micro-polygon, thus not suitable for real time application. These approaches are usually used by ray tracing and global illumination algorithm, of which rendering time is not the main limitation. The solution to storage problem of the displacement map is continuously developed.

The Relief mapping [13], shown in Figure 2-7, and per-pixel displacement map [18] cast ray to intersect with the displaced surface. These two methods are quite similar. Their results are indistinguishable. The per-pixel displacement map method uses sphere tracing to step the ray. This technique is guaranteed to intersect the surface at the right position, but it must store a distant map, for ray to jump, in 3D texture, resulting in high storage space. In addition, the lower bound of the number of steps used is still high. On the other hand, Relief map uses linear search combined with binary search to find the intersection point. This method may miss some high frequency in the displacement map, but it is quite fast and viewers can hardly notice the artifact. It can render surface self-occlusion correctly, however it cannot display silhouette.



Figure 2-7: Relief mapping [13].

2.2.4 General Displacement Map

General displacement map (GDM) [19] and view-dependent displacement map (VDM) [20] in Figure 2-8 use five dimensions storage. These methods can quickly find the coordinate of the displaced surface. Instead of casting ray to the surface, these methods store the pre-compute distance information of the displaced point to the original point. These data must be stored at all 3D positions in the displacement region and at all viewing directions, thus they use 5D data structures. Thus, the GDM and VDM require large data storage and must be compressed. They can render correct silhouette.

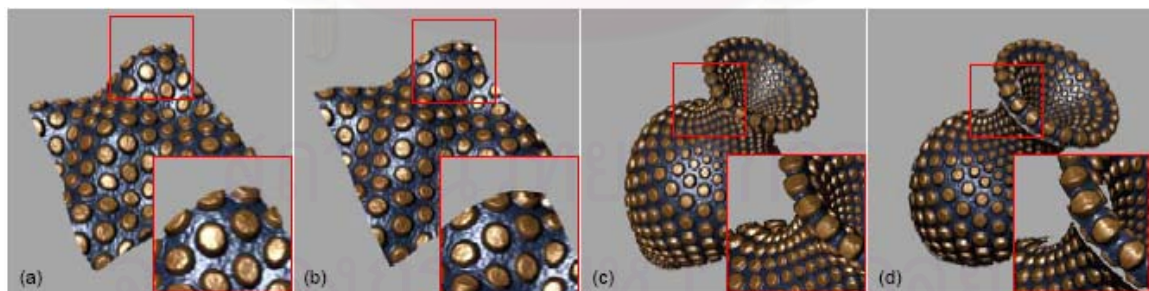


Figure 2-8: General displacement map [19].

2.2.5 Billboard Cloud

Billboards [21, 22] were proposed to render a forest scene. These methods use a group of billboards, each being the sliced image of some group of trees. These methods have the problem with limited viewing region, which makes them more suitable for a bird eye's view of the forest. The Billboard Cloud Technique [23] also

uses a number of billboards, each representing some portion of the model. The billboard cloud is rendered as a conventional texture mapped polygon. Because it uses a plane to approximate nearby geometry, the billboard cloud results in cracks in the image. The cracks are caused from the approximation in each plane of the billboard as seen in Figure 2-9.

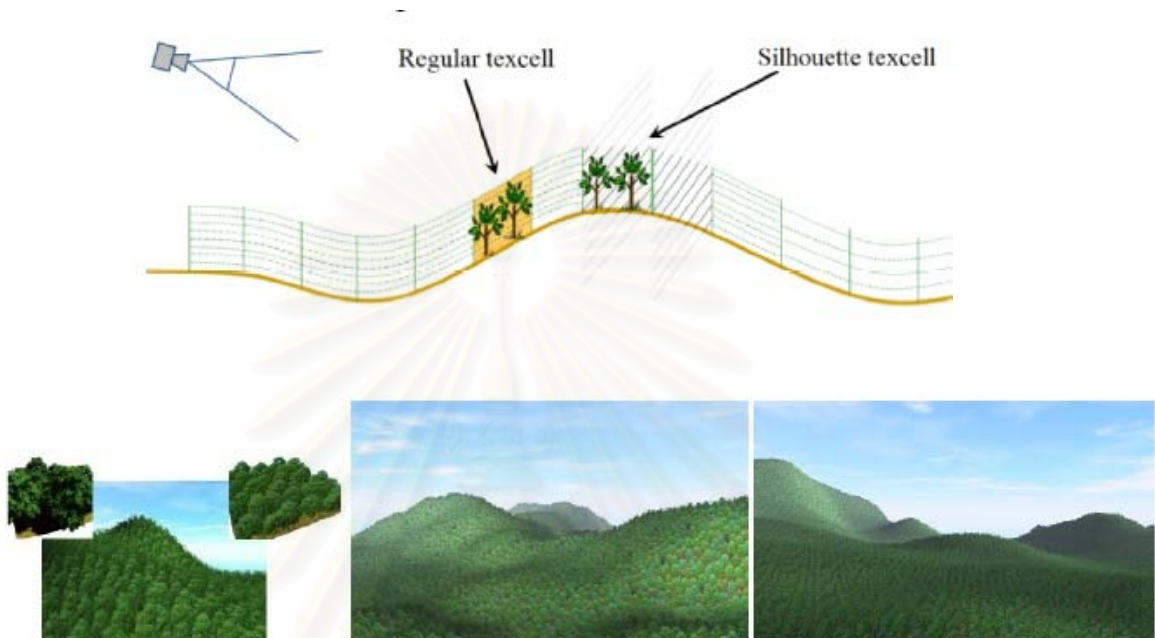


Figure 2-9: Forest represented with a group of billboards [1].

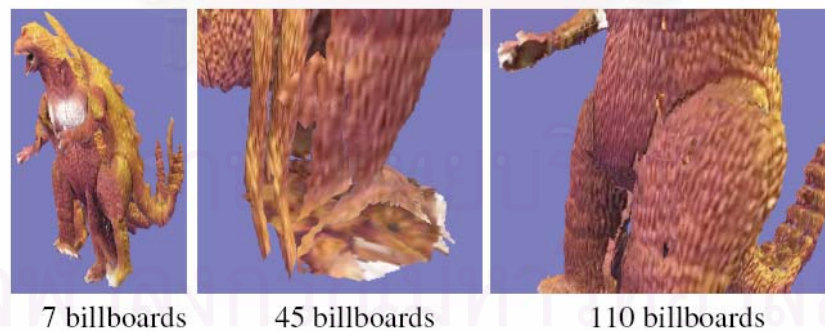


Figure 2-10: Billboards Cloud [23].

CHAPTER 3

PROPOSED ENHANCED BILLBOARDS

In this section, a modeling technique using enhanced billboards for representing arbitrary polygon models is described. The data structure of the enhanced billboards is shown in section 3.1. The algorithm to construct enhanced billboards from polygon mesh is described in section 3.2. The rendering algorithm for enhanced billboards is shown in section 3.3. Finally, methods to increase the quality of the rendered image using enhanced billboards are shown in section 3.4.

3.1 Enhanced Billboards

Enhanced billboard is a representation of a static 3D model. This representation is extending from the Billboard clouds mentioned in section 2.2.5. Enhanced billboards has an additional height map used for storing the location of the 3D model.

An enhanced billboard b with its plane p consists of a color map, a transparency map, a normal map and a depth map. These maps are stored in two images. One image stores the color map in the RGB channel and the transparency map in the alpha channel. The other image stores the normal map in the RGB channel and the depth map in the alpha channel. Color and normal of projected polygon $poly$ is stored in color map and normal map respectively. Depth map stores the distance between $poly$ and p . Transparency map informs which pixels of b contain projected $poly$.

There are some initial conditions for projecting a polygon to an enhanced billboard.

1. $poly$ can be stored in b , if and only if the dot product of the normal vector of $poly$ and the normal vector of p is less than zero.

2. b cannot store overlapping polygons in the same billboard. Thus, the polygons that are occluded by some other polygons when projected to p have to be stored in another enhanced billboard.

3. If the distance between $poly$ and p is more than the limited depth value e_{depth} , b cannot store that $poly$. This prevents the varying depth of each billboard that will result in inconsistency in rendering time.

The bounding box of B is computed from the height map of b and store with b . This bounding box will be used during rendering time.



Figure 3-1: The first five enhanced billboards of the horse model.

3.2 Building Enhanced Billboards

This section shows how a polygon model is represented as a set of enhanced billboards. The planes of enhanced billboards are chosen to minimize the number of enhanced billboard needed to capture all the polygons in the model. The algorithm to choose the planes of enhanced billboards is based on using the normal vectors of the polygons.

$$F = \{poly_1, poly_2, \dots, poly_n\}$$

The algorithm is as follow.

1. Repeat until $F = 0$.
2. Build an array A of view plane. Each element a_i of A is a view plane located around the input model and its normal vector is point to the center of the model. The dots shown in Figure 3-2 represent the view plane.
3. Project the model to all a_i .
4. Assign a' be the a_i which has the most number of polygons projected on it.
5. Assign plane p_i be the plane of a' .
6. Assign S be the set that stores all the polygons which can be projected to p_i .
7. Assign S' be the set of polygons in S , which each $poly$ has the distance between $poly$ and p_i smaller than the limited depth value e_{depth} .
8. Assign S'' be the set of polygons in S' , which each $poly$ does not occluded by others $poly$.
9. Orthogonally project all polygons in S'' to p_i .
10. Assign p_i be the plane of enhanced billboard b_i .
11. Delete all $poly$ in F that are the members of S'' .
12. $i \leftarrow i + 1$

This results in a series of enhanced billboards representing some portions of the model from different viewing directions. By using all enhanced billboards, the complete model can be reconstructed.

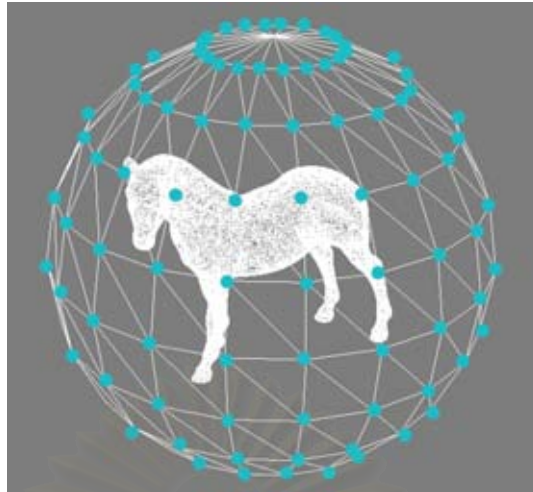


Figure 3-2: The projection cameras around the horse model.

3.3 Rendering Enhanced Billboards

Because an enhanced billboard is treated as a height map, a ray-height field intersection algorithm can be used to render enhanced billboards. The algorithm utilizes the programmability of the graphics hardware to do per-pixel ray-height field intersection. The algorithm is as follow.

1. For each enhanced billboard b_i , the bounding box B of b_i is constructed in screen space. b_i is rendered with all of the sides of its bounding box, which guarantees that all geometry represented in the enhanced billboards will be rendered.

2. For each texel t on the bounding box,

- 2.1 Construct ray Ray that has the origin at eye location and has the direction toward t .

- 2.2 Traverse Ray through the depth map of b_i with the step size d along the Ray direction until one finds a depth value close to the current depth with some threshold value e . Let (u, v) be the coordinate of the depth map at this intersection. If Ray travels pass b_i and does not find any depth value. t is rendered as transparent pixel.

- 2.3 Retrieve the color and the normal information from the color map and normal map of b_i using (u, v) . This information is used to compute shading of t .

In Figure 3-3, the ray hits the depth map at the circle and reports a hit. Figure 3-4 shows the pseudo-code of the algorithm.

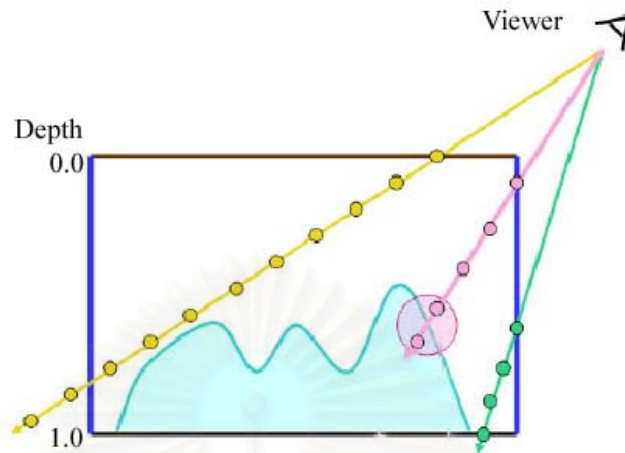


Figure 3-3: The ray-casting diagram.

Render Enhanced Billboards

Input: enhanced billboards images

Output: picture rendered on screen

e = hit range, $maxStep$ = number of step

For each enhanced billboards

Build bounding box of enhanced billboards

For each texel on the plane of box

$start$ = texel position

dir = direction from eye position to $start$

$d = 0$; $hitd = 0$; $hit = false$;

$step = \sqrt{2} / maxStep$

For 1 to $maxStep$

$ray = start + dir * d$

$depth = GetDepthMapValue(ray.xy)$

if ($ray.z > depth$) and ($ray.z - e < depth$)

$hit = true$; $hitd = d$; exit loop

$d = d + step$

if (hit)

$hitpos = start + dir * hitd$

Render_Light($hitpos$)

else

render as transparent pixel

Figure 3-4: Pseudo-code of the algorithm for rendering enhanced billboards.

In Figure 3-5 (1), a horse model is rendered using only one plane. Some parts of the geometry are missing. Figure 3-5 (2) shows the horse model rendered with all bounding box planes. Thus, all geometries in the enhanced billboard can be shown. In the view direction, three sides of the box are visible at most, the top and the two sides. The bottom of the box does not need to be rendered because the bottom plane shows the rear side of the polygons.

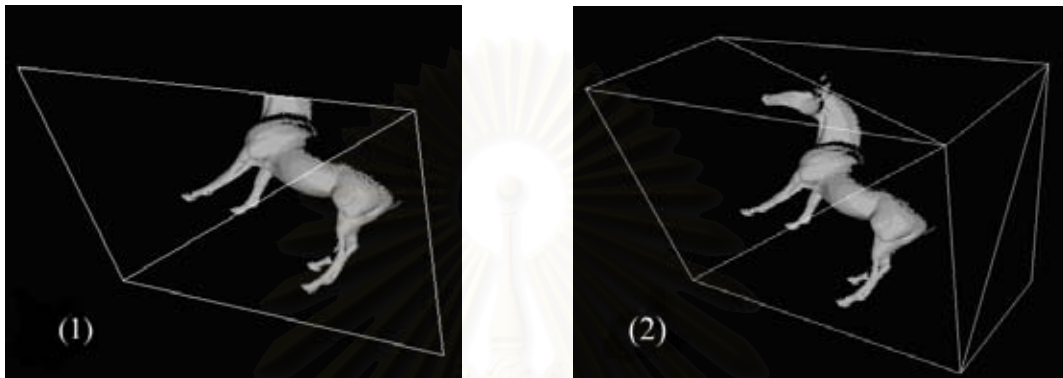


Figure 3-5: Horse billboard. (1) Horse model rendered with one size. (2) Horse model rendered with all three size.

3.4 Solving Holes Artifact

There may be holes containing in the rendered image using enhanced billboards (Figure 3-6). There are two reasons why holes exist in a rendered image. The first one is because when a model is projected to the enhanced billboards, some polygons in the model are so tiny that their projected areas are less than a pixel. The second one is because of the render algorithm. Since the algorithm uses linear step ray intersection test, if the step is too coarse, the algorithm can miss the test and report a non hit. If the step is too fine, the algorithm runs very slow. The line fragments in Figure 3-6 are not rendered because the algorithm reports no hit. This problem depends on the view direction. The view in Figure 3-6 (1) does not show much of the holes, but in Figure 3-6 (2), the holes are clearly visible. It is difficult to identify where holes will be shown.



Figure 3-6: The holes artifact.

In order to solve the holes artifact, the hit range threshold e is increased to cover the gap in the ray step size. By experiment, when e is doubled to 0.02 to fill the holes, a resulting image is actually coarser than the original one, but this is hardly noticeable. However, if e is increased further, the silhouette of the model image apparently does not work properly.

Another method to fill the holes artifact in a rendered image is to use a coarse mesh, which is constructed from the original model. It represents the overall structure of the model and ignores the details. The enhanced billboards are rendered first and then the coarse mesh. The coarse mesh fills the pixels that are not rendered by the enhanced billboards. By experiment, the coarse meshes usually have polygons 10 times smaller than the original ones. Figure 3-8, (1) shows the horse model rendered with enhanced billboards. Figure 3-8, (2) illustrates the coarse polygon model of the horse. Figure 3-8, (3) shows the horse model rendered using both the enhanced billboards and the coarse polygon mesh.

This hybrid method performs pretty well. It can fill holes and the quality of the image is still high. The running time in rendering is not much more different from that of the enhanced billboards alone. The pixels rendered by low quality coarse mesh, are hardly distinguishable from the pixels rendered by enhanced billboards. This is because the holes spread all over the image and do not group together.



Figure 3-7: Hit range threshold. (1) Hit range threshold $e = 0.01$. (2) Hit range threshold $e = 0.02$.



Figure 3-8: Enhanced billboards with coarse mesh.

CHAPTER 4

EXPERIMENTAL RESULTS

This section shows the experimental results of the proposed enhanced billboards. The number of tested model is 30. The running time and the average intensity difference to the original model are also provided. The resulting images of all the models with rendering information are shown in Appendix A and Appendix B.

4.1 Testing Environment

The rendering technique described in this research is implemented using High Level Shading Language (HLSL). HLSL is included with DirectX library. It is used to write the Vertex and Pixel Shader. The programmability of the GPU is used for casting rays through enhanced billboards. The program is implemented with Visual C++ 2003 and DirectX 9.0 with Shader model 3.0. The testing system is a 2.8 GHz PC with 512 MB of memory, using a GeForce FX6800 with 128 MB of memory. The method is tested with a variety of polygon models, containing around 5,000 to 50,000 polygons. The number of the tested model is 30.

Two programs are built. One is enhanced billboards projector. It accepts a polygon model as input and its output are images of enhanced billboards. The other is enhanced billboards renderer. This one is used to render the enhanced billboards. It is a real-time application of which user can interact by moving the model and the camera freely. The resolution of the output screen can be modified.

In the enhanced billboards projector program, the array that used as projection camera has the size of 36x18 elements. The projection cameras are distributed evenly in all directions around the object. The enhanced billboards have the resolution of 512x512 pixels. The resolution of the billboard can be adjusted to suit the screen resolution.

In enhanced billboards renderer, a depth map has one unit width and one unit height. The ray length to perform the intersection test is $\sqrt{2}$. This is the diagonal of the 1x1 unit depth map. The algorithm uses linear step search to find the intersection of the ray and the height map, the number of steps is 100 and the hit range of each

step is 0.01. Thus, 100×0.01 equal to 1 but the ray length is $\sqrt{2}$. This can cause problem since the number of steps is too coarse. Some parts of the ray can miss the test. This problem can be solved by increasing the hit range threshold to 0.02 since the test can cover all the ray length.

4.2 Experiment and Results

The test models are chosen to represent models that are used in real world applications. The test models in this experiment have a variety of surfaces such as flat surfaces, curve surfaces and bumpy surfaces. There are models that have many surface occlusions or simple surface occlusions like sphere shape. There are models that were built from curve editing tools, polygon tools and the 3D scan.

Figure 4-1 shows an example of the enhanced billboards of the horse model. Figure 4-2 shows the horse model rendered using enhanced billboards. In Figure 4-2, (1) the horse model is rendered with enhanced billboards. There is holes artifact in the image. In Figure 4-2, (2-3) the holes artifact is solved by increasing the threshold value by 0.01 and adding coarse mesh. Figure 4-2, (4) shows the original polygon model. The original horse model has about 40,000 polygons. Its coarse mesh has 4,300 polygons. The storage size of all enhanced billboards is 1 MB while the original model uses nearly 3.5 MB. The coarse mesh uses 365 KB. A significant number of storage are saved. Figure 4-3, shows the storage size of ten high-resolution polygon models and compares with their enhance billboards representation.



Figure 4-1: The enhanced billboards of the horse model.

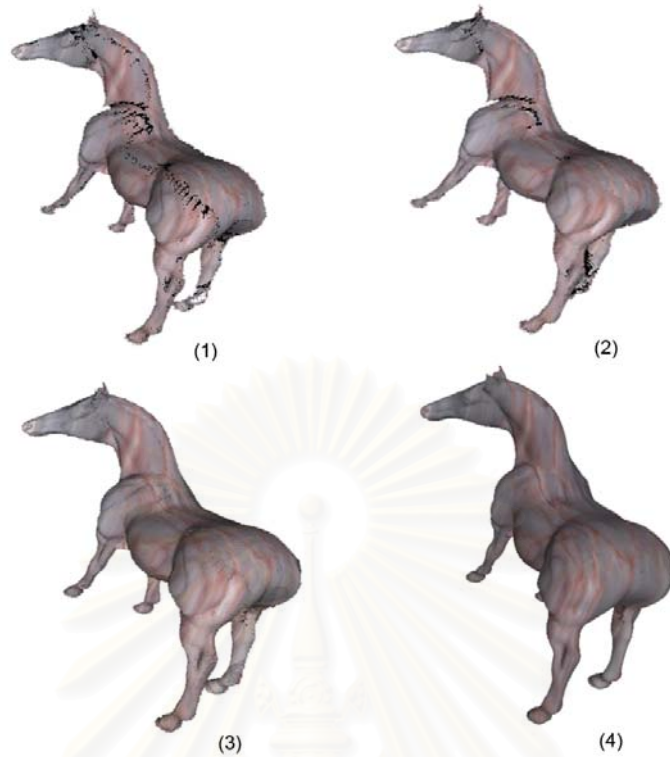


Figure 4-2: The horse model rendered using enhanced billboards.

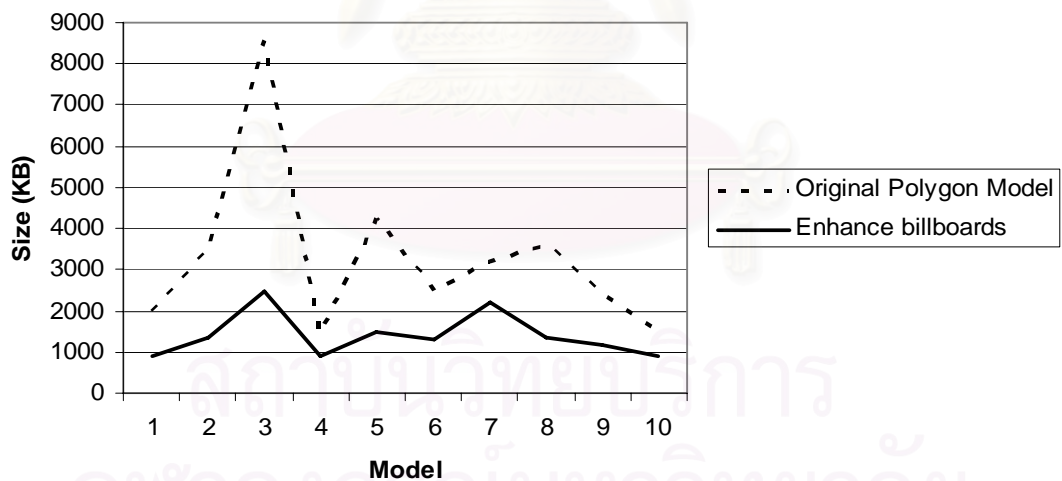


Figure 4-3: The storage size of original polygon models compare to enhance billboards.

This research also does an experiment on the projection of the enhanced billboards. Figure 4-4 shows the enhanced billboards of a dragon model that are projected by the proposed greedy algorithm. Figure 4-5 shows the enhanced billboards of the dragon model that are projected from the direction of the X axis, -X

axis, Y axis, -Y axis, Z axis and -Z axis of the world coordinate. This method uses greater number of billboards than the greedy method to represent the same number of polygons. The greedy method is better both in term of storage size and rendering time, because these factors depend on the number of billboards.

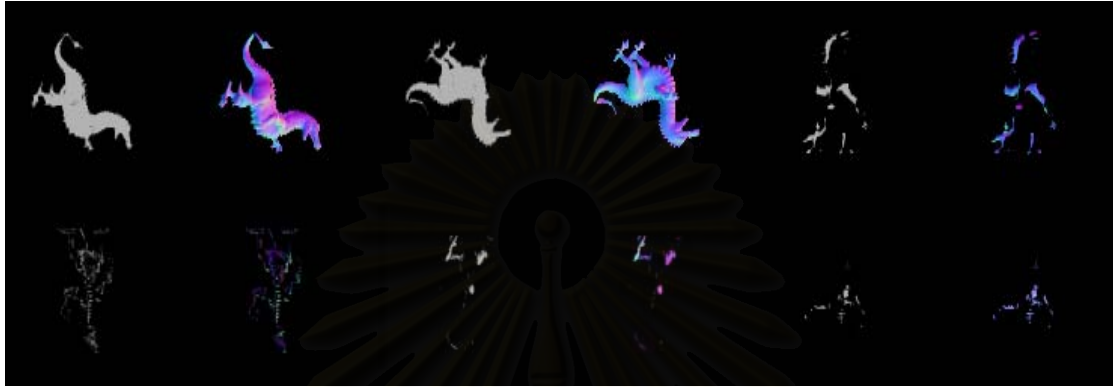


Figure 4-4: The greedy projection of the dragon model

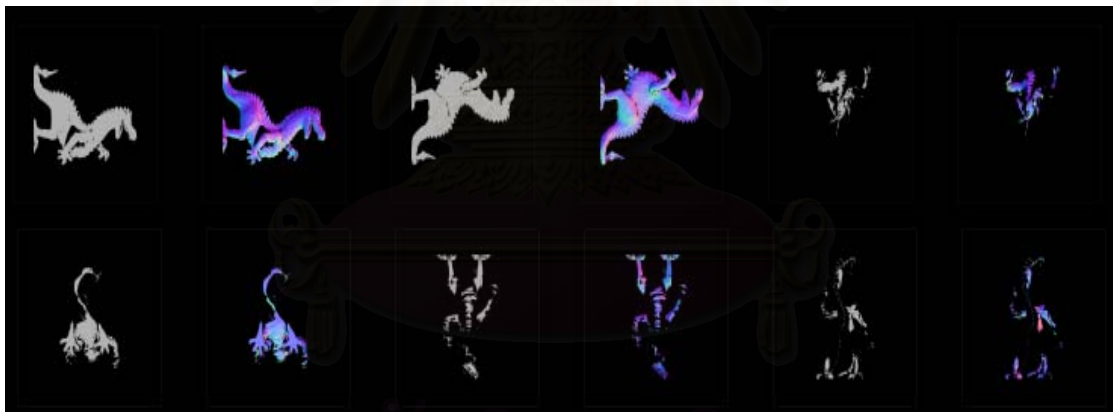


Figure 4-5: The axis projection of the dragon model.

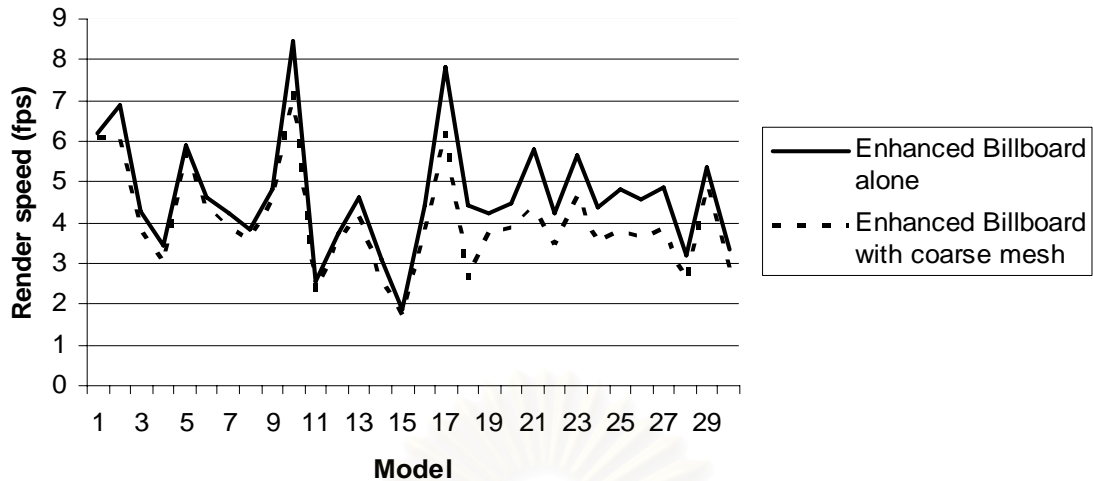


Figure 4-6: The render speed of all the models (with and without coarse mesh) in 800x600 resolution.

From Figure 4-6, the running time of each model is nearly the same, if they use the same number of enhanced billboards. The rendering time of the enhanced billboard with coarse mesh and without coarse mesh are not much different from each other.

The average intensity difference between enhanced billboards with hit range = 0.01 and enhanced billboards with hit range = 0.02 is 0.66 ± 4.56 .

The average intensity difference between enhanced billboards with hit range = 0.01 and enhanced billboards with coarse mesh is 9.04 ± 6.61 .

The average intensity difference between enhanced billboards with resolution 800x600 and enhanced billboards with resolution 400x300 is 2.61 ± 2.56 .

The average intensity difference between enhanced billboards with resolution 800x600 and enhanced billboards with resolution 200x150 is 4.78 ± 4.35 .

4.3 Discussion

From the experimental results, enhanced billboards are usually limited by the fill rates of the pixel shader, which uses many instructions. However, it scales well with the resolution and renders efficiently at medium to high distance in any direction with complete effects, such as parallax or silhouette. In contrast, the polygon representation method does not scale according to the resolution. In addition, many

distant polygons are rasterized to the same pixel, resulted as an alias in the rendered image.

There might be a problem if the input model has two polygons that are intersecting one another. The occlusion query will report that these two polygons are occluded and are not chosen. The solution to this problem is to force these intersecting polygons to bypass the occlusion test. Another problem is that some tiny polygons, with projected area less than a pixel are also discarded.

Enhanced billboards containing little polygons seen as minute fragments in the image are also ignored, since they have little contribution to the final rendered image. This helped to increase the rendering performance.

The polygons are not projected if they are almost perpendicular to the chosen direction even though they were facing forward. This was because the enhanced billboard could not capture the information of almost perpendicular polygons. These polygons covered a small area of the enhanced billboard and thus were not sufficient for reconstructing the model.

The holes artifact can be solved either by increasing the threshold value or by rendering enhanced billboards with a coarse mesh. However, using a coarse mesh seems to be a more reliable method. An extra time for rendering the coarse mesh is about 2-3 % of the original rendering time using the enhanced billboards alone. The enhanced billboards show consistency in rendering time, since they are solely dependent on the number of pixels. Moreover, the computation time drops automatically when the user moves away from the object. Thus, enhanced billboards have continuous Level-of-Details.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This section concludes the thesis and provides some discussion on the experimental results. Future works are also suggested.

5.1 Conclusion

This research presents a method for simplifying models by using a new representation and rendering technique. The enhanced billboards, each representing some portions of a model, store all information of the original model. The information including a depth map, a normal map, a color map and a transparency map is projected onto a plane by the greedy algorithm. The enhanced billboards are rendered using the new ray-height field intersection algorithm. They can produce effects such as parallax and silhouette in any viewing directions. The quality of the enhanced billboards is comparable to that of the source model while the frame rate does not depend on the number of polygons. The experimental data shows that the speed increases while the resolution of the image decreases. The frame rate of the test models appears to increase 4 times when the resolution is halved.

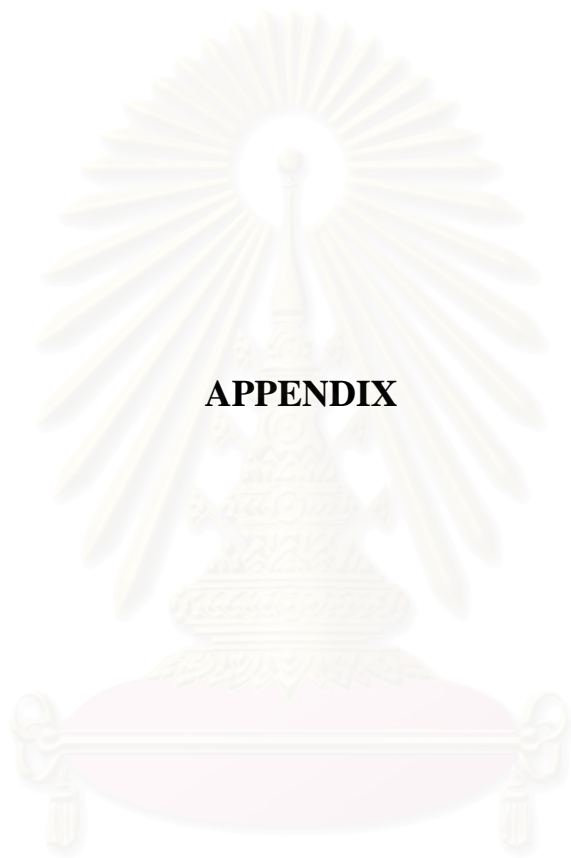
In addition, the hit range threshold value of 0.02 improves the average intensity difference by 0.66 ± 4.56 from the value of 0.01. This seems to be a little improvement at first. However, the holes artifacts were solved effectively in some models such as Shape2 and She. If the image rendered with the hit range value of 0.01 has a low average intensity difference, the one rendered with the value of 0.02 may not improve the difference. The best way is to adjust the hit range value to 0.02 when average intensity difference of the rendered image with the value of 0.01 is high. The enhanced billboards with the coarse mesh, however, improve the average intensity difference of the image by 9.04 ± 6.61 . This method shows the best improvement in the correctness of the rendered image. The errors of the images could be different when viewed in different directions.

The average intensity difference between the enhanced billboards and the original polygon model is high in some models such as Body and Fan because of

difference in shading algorithm. The constant terms used in both rendering styles might have some different values. Thus, the quality of a rendered image could be judged by the rendered images (*see* Appendix B). The average intensity difference when using high-resolution images is higher than when using low resolution. This is because in low resolution, the color differences were averaged to the nearby pixels, hence lower the differences.

5.2 Future Work

The enhanced billboard technique can be easily extended to render shadow using a shadow map. The shadow map algorithm is not different from the polygonal model. First, the enhanced billboards are rendered from the light viewpoint and stored as a shadow map using the same rendering algorithm as viewing enhanced billboards. Then, the map is used during rendering time as in the polygonal model. In addition, enhanced billboards can be used to speed up the ray intersection test for a ray-tracing algorithm. Normally, this algorithm builds an accelerated structure, such as a grid cell where the ray is located, to query all the geometric structures, which occupy in the cell. However, enhanced billboards already represent the positions of all geometric models, which located inside of the bounding box. Thus, there is no need to build an auxiliary data structure.



APPENDIX

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDIX A.

This section shows the result rendering times of enhanced billboards from the experiments. The number of the test models is 30.

Table A-1 shows the rendering time of the enhanced billboards with and without coarse mesh. The rendering times are linearly scaled with the image resolution. They depend on the number of pixels rendered on screen and the number of billboards being used.

Table A-2 shows the average intensity difference between enhanced billboards and the reference original polygon models. The average intensity difference has the range 0 – 580. The differences on the tests are due to the differences in shading algorithm between enhanced billboards and polygon method. The enhanced billboards compute lighting per pixel basis and can give more accurate look while polygon uses some approximation and interpolation in shading. The constant terms used in both rendering styles may have some different values.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Table A-1: The rendering time of the enhanced billboards with 3 different resolutions.

Model	Render Speed (fps) High Resolution (800x600)	Render Speed (fps) Medium Resolution (400x300)	Render Speed (fps) Low Resolution (200x150)
1. Cylinder	6.20	30.10	125.33
2. Cylinder + coarse mesh	6.10	29.12	110.23
3. Dolphin	6.89	26.45	96.70
4. Dolphin + coarse mesh	6.12	24.32	86.44
5. Horse	4.27	21.45	111.34
6. Horse + coarse mesh	3.85	20.34	66.98
7. Dragon	3.42	9.65	50.60
8. Dragon + coarse mesh	3.02	8.30	45.36
9. Moai	5.89	27.10	115.84
10. Moai + coarse mesh	5.72	26.30	111.42
11. Sphere	4.63	27.80	105.30
12. Sphere + coarse mesh	4.37	23.43	85.62
13. Teapot	4.23	22.34	77.32
14. Teapot + coarse mesh	3.84	19.30	65.30
15. Torus	3.85	22.40	111.94
16. Torus + coarse mesh	3.60	18.98	80.33
17. Woman	4.83	16.45	51.39
18. Woman + coarse mesh	4.52	14.35	48.32
19. Cone	8.45	35.56	130.45
20. Cone + coarse mesh	7.24	30.30	110.34
21. Face	2.58	9.38	65.06

Model	Render Speed (fps) High Resolution (800x600)	Render Speed (fps) Medium Resolution (400x300)	Render Speed (fps) Low Resolution (200x150)
22. Face + coarse mesh	2.30	8.90	62.5
23. Hand	3.72	15.11	42.09
24. Hand + coarse mesh	3.58	13.32	38.20
25. Head	4.60	12.20	31.39
26. Head + coarse mesh	4.20	11.09	29.04
27. Face2	3.10	14.51	50.55
28. Face2 + coarse mesh	2.58	13.10	47.89
29. Man	1.86	5.40	24.18
30. Man + coarse mesh	1.70	4.47	20.25
31. Hand2	4.43	16.39	82.04
32. Hand2 + coarse mesh	3.76	14.14	77.02
33. Shape1	7.80	31.84	108.32
34. Shape1 + coarse mesh	6.25	29.5	94.53
35. Shape2	4.43	14.71	56.02
36. Shape2 + coarse mesh	2.59	12.74	54.67
37. Shape3	4.23	13.94	42.35
38. Shape3 + coarse mesh	3.78	11.65	37.43
39. Shape4	4.50	17.08	42.49
40. Shape4 + coarse mesh	3.90	13.90	38.81
41. Shape5	5.78	21.07	82.59
42. Shape5 + coarse mesh	4.33	18.43	80.10
43. She	4.25	18.66	45.6

Model	Render Speed (fps) High Resolution (800x600)	Render Speed (fps) Medium Resolution (400x300)	Render Speed (fps) Low Resolution (200x150)
44. She + coarse mesh	3.45	17.84	43.11
45. Tutan	5.68	27.18	92.99
46. Tutan + coarse mesh	4.66	24.62	91.00
47. Face3	4.36	13.00	59.28
48. Face3 + coarse mesh	3.50	10.59	54.25
49. Leg	4.80	16.02	58.37
50. Leg + coarse mesh	3.84	14.56	46.98
51. Tail	4.55	23.08	49.63
52. Tail + coarse mesh	3.66	18.14	45.56
53. Monster	4.86	15.46	56.25
54. Monster + coarse mesh	3.83	13.63	50.64
55. Body	3.19	15.39	48.92
56. Body + coarse mesh	2.58	12.20	45.66
57. Boat	5.38	62.34	150.09
58. Boat + coarse mesh	4.91	51.34	145.45
59. Fan	3.36	36.33	102.34
60. Fan + coarse mesh	2.83	28.97	95.74

Table A-2: The average intensity difference of the enhanced billboards and the original polygon mesh.

Model	Enhanced billboards with hit range = 0.01			Enhanced billboards with hit range = 0.02 Resolution 800x600	Enhanced billboards with coarse mesh Resolution 800x600
	Resolution 800x600	Resolution 400x300	Resolution 200x150		
1. Cylinder	39.45	37.06	31.85	37.06	31.85
2. Dolphin	28.42	25.99	25.29	33.61	24.83
3. Horse	12.14	9.71	8.00	10.52	6.91
4. Dragon	22.79	14.33	10.22	24.74	4.97
5. Moai	32.49	26.56	22.38	26.55	19.80
6. Sphere	25.81	26.57	25.59	26.04	6.56
7. Teapot	28.20	18.51	11.78	11.67	5.96
8. Torus	58.09	56.71	50.96	57.64	51.80
9. Woman	124.50	121.23	118.73	126.98	124.52
10. Cone	5.56	5.81	5.16	5.48	3.33
11. Face	37.21	35.79	33.61	40.88	16.24
12. Hand	27.38	24.65	21.73	28.23	21.68
13. Head	14.60	13.65	13.07	14.93	12.93
14. Face2	27.68	25.39	23.30	28.30	24.25
15. Man	51.26	46.83	41.98	50.51	37.66
16. Hand2	35.07	33.19	28.74	36.60	24.53
17. Shape1	21.22	17.66	13.82	21.73	9.39

Model	Enhanced billboards with hit range = 0.01			Enhanced billboards with hit range = 0.02	Enhanced billboards with coarse mesh
	Resolution 800x600	Resolution 400x300	Resolution 200x150	Resolution 800x600	Resolution 800x600
18. Shape2	11.57	11.51	11.22	11.44	2.88
19. Shape3	3.82	3.00	2.33	4.38	2.61
20. Shape4	8.16	7.79	7.62	7.99	0.79
21. Shape5	22.04	17.71	17.39	17.03	4.59
22. She	22.69	15.62	10.90	8.51	5.20
23. Tutan	13.78	9.89	7.06	14.50	7.88
24. Face3	12.29	9.74	7.97	13.09	7.16
25. Leg	21.36	21.71	23.00	23.30	19.30
26. Tail	12.21	12.42	13.56	12.99	10.47
27. Monster	81.63	77.70	74.41	84.95	69.46
28. Body	41.64	40.76	40.92	41.43	26.12
29. Boat	12.13	11.11	11.28	13.68	10.89
30. Fan	33.84	32.19	31.79	34.33	23.38

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDIX B.

This section shows the result images of enhanced billboards from the experiments. The number of the test models is 30. The number of polygons, the file size and the number of billboards being used are also provided.

Two figures are shown for each model. Figure1 shows the model rendered with four difference techniques. Figure1.1 shows model rendered using enhanced billboards. Figure1.2 shows enhanced billboards rendered using increase threshold. Figure1.3 shows enhanced billboards rendered with coarse mesh. Figure1.4 shows the original polygon model. Figure2 shows the model rendered with three different resolutions, which are 800x600, 400x300 and 200x150 respectively. Figure2.1 shows model rendered using enhanced billboards. Figure2.4 shows the original polygon model.



1. Cylinder Model

Polygons in original model	4,200 polygons
Polygons in coarse model	78 polygons
Original model file size	234 KB
Coarse model file size	5 KB
Number of billboards used in rendering	4
Summation of images size	800 KB

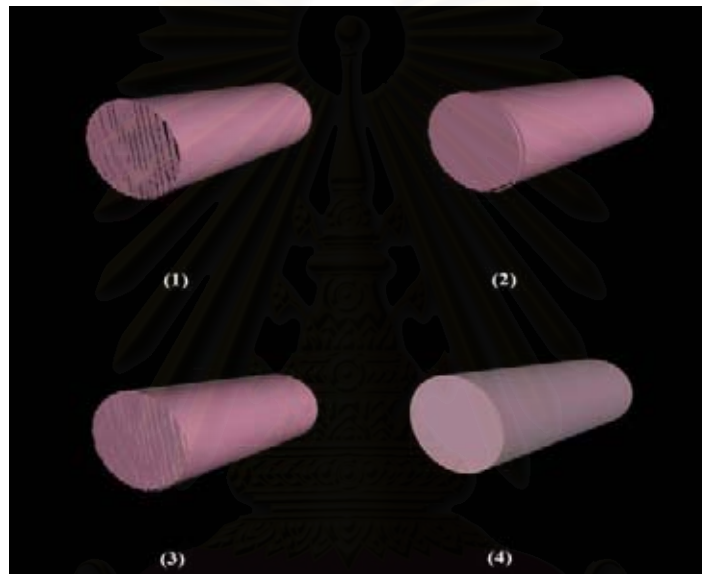


Figure A-1.1: Cylinder model.

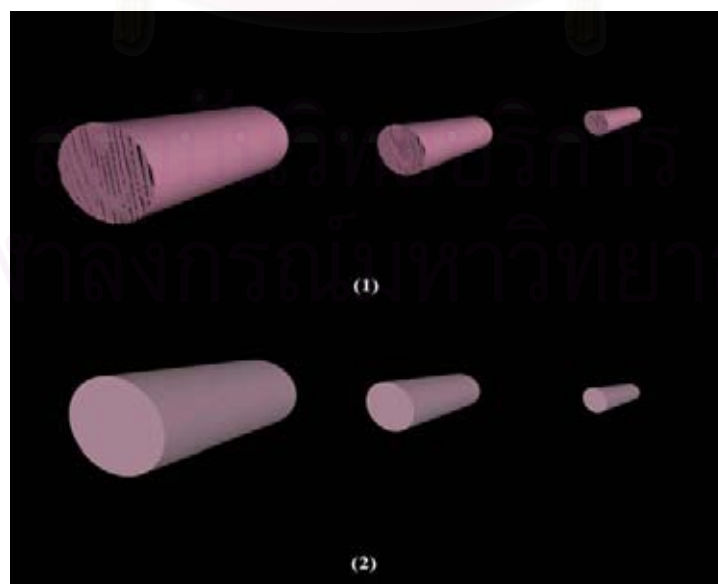


Figure A-1.2: Cylinder model with different resolution.

2. Dolphin Model

Polygons in original model	26,000 polygons
Polygons in coarse model	1,700 polygons
Original model file size	2.01 MB
Coarse model file size	121 KB
Number of billboards used in rendering	4
Summation of images size	800 KB

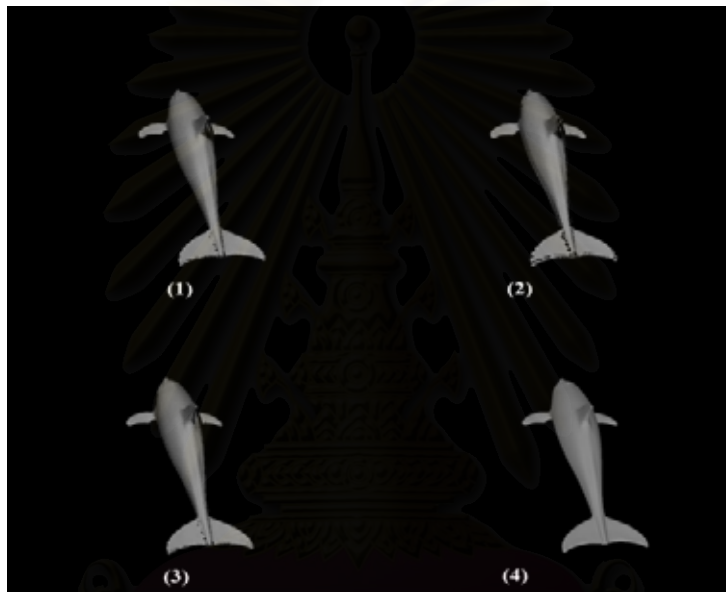


Figure A-2.1: Dolphin model.

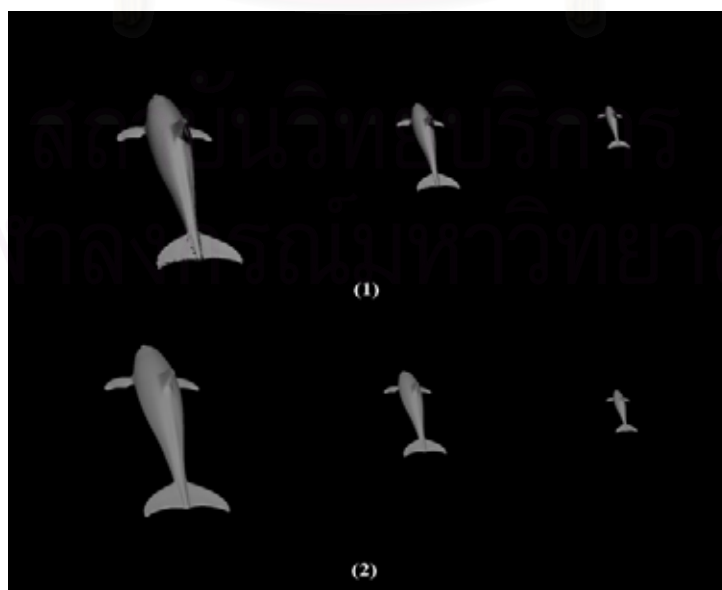


Figure A-2.2: Dolphin model with different resolution.

3. Horse Model

Polygons in original model	39,700 polygons
Polygons in coarse model	4,300 polygons
Original model file size	3.47 MB
Coarse model file size	365 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

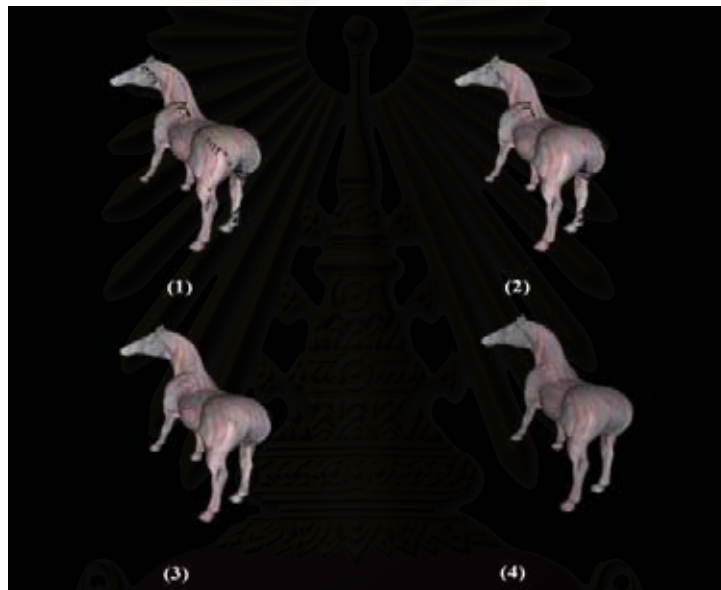


Figure A-3.1: Horse model.



Figure A-3.2: Horse model with different resolution..

4. Dragon Model

Polygons in original model	117,000 polygons
Polygons in coarse model	16,000 polygons
Original model file size	8.53 MB
Coarse model file size	1.28 MB
Number of billboards used in rendering	6
Summation of images size	1.20 MB



Figure A-4.1: Dragon model.



Figure A-4.2: Dragon model with different resolution.

5. Moai Model

Polygons in original model	20,000 polygons
Polygons in coarse model	1,160 polygons
Original model file size	1.46 MB
Coarse model file size	81 KB
Number of billboards used in rendering	4
Summation of images size	800 KB



Figure A-5.1: Moai model.

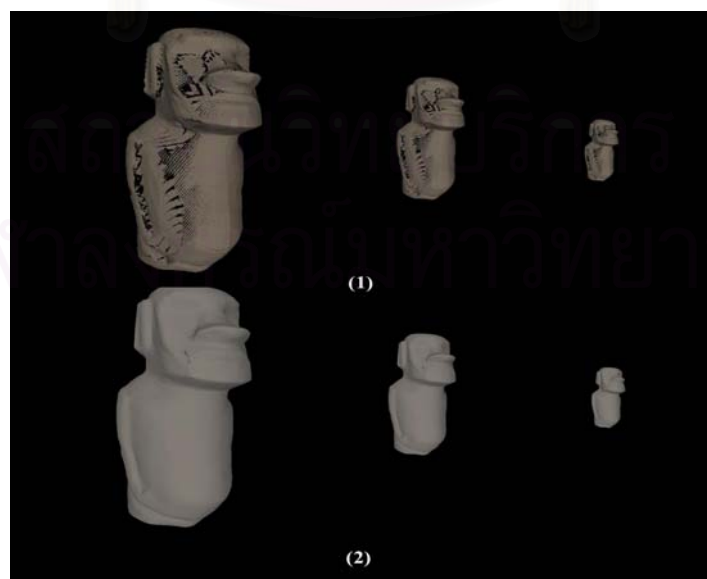


Figure A-5.2: Moai model with different resolution.

6. Sphere Model

Polygons in original model	9,800 polygons
Polygons in coarse model	850 polygons
Original model file size	736 KB
Coarse model file size	62 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

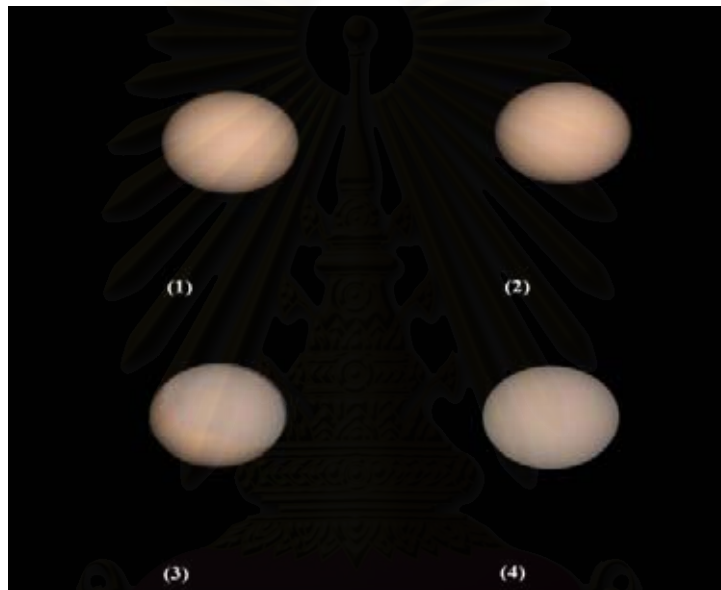


Figure A-6.1: Sphere model.

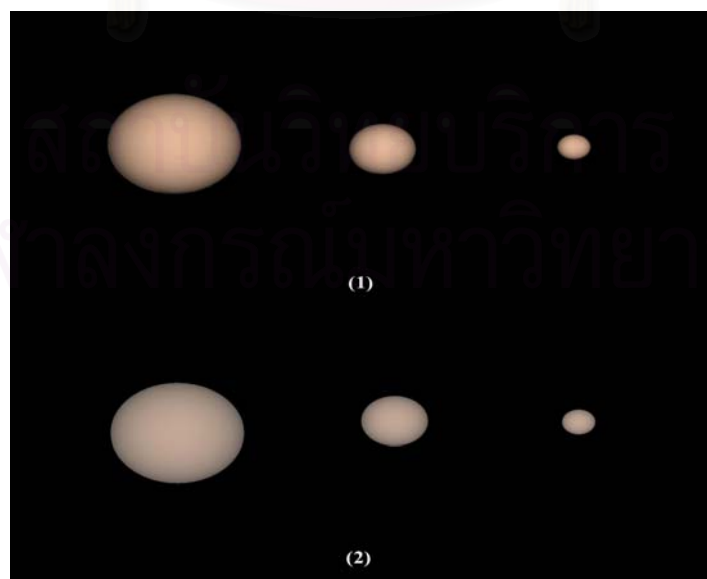


Figure A-6.2: Sphere model with different resolution.

7. Teapot Model

Polygons in original model	14,300 polygons
Polygons in coarse model	950 polygons
Original model file size	1.24 MB
Coarse model file size	89.2 KB
Number of billboards used in rendering	6
Summation of images size	1.20 MB



Figure A-7.1: Teapot model.



Figure A-7.2: Teapot model with different resolution.

8. Torus Model

Polygons in original model	9,600 polygons
Polygons in coarse model	770 polygons
Original model file size	724 KB
Coarse model file size	53 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

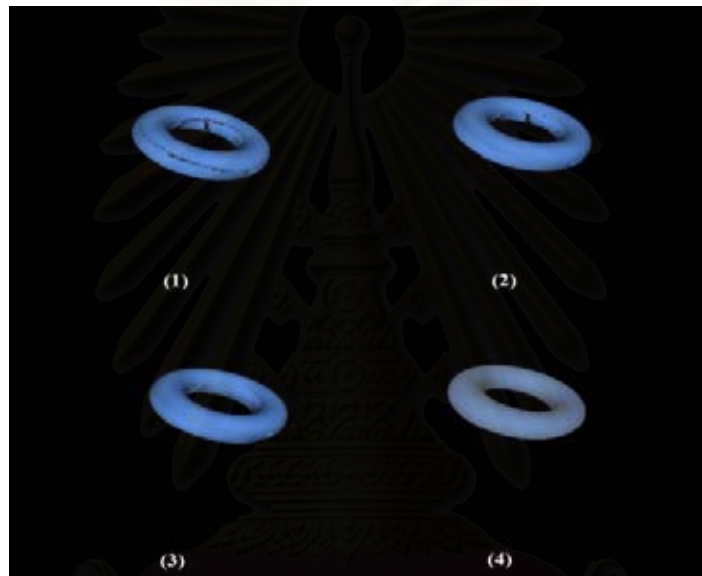


Figure A-8.1: Torus model.

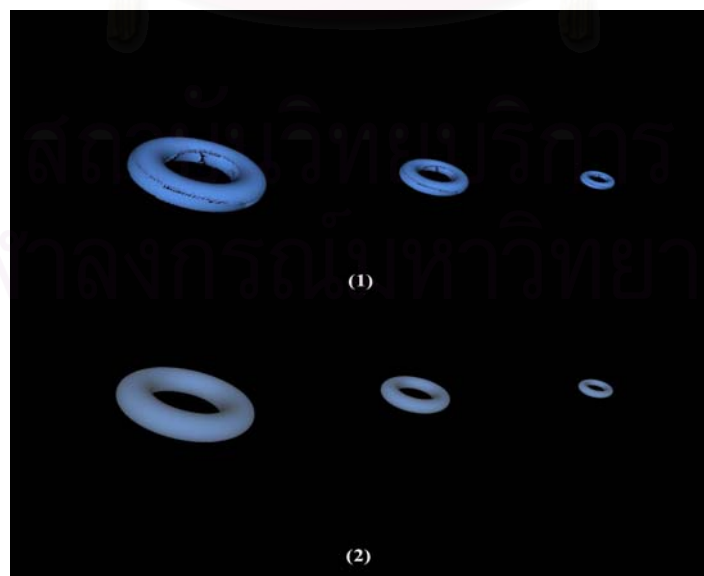


Figure A-8.2: Torus model with different resolution.

9. Woman Model

Polygons in original model	44,200 polygons
Polygons in coarse model	7,300 polygons
Original model file size	4.22 MB
Coarse model file size	690 KB
Number of billboards used in rendering	4
Summation of images size	800 KB



Figure A-9.1: Woman model.



Figure A-9.2: Woman model with different resolution.

10. Cone Model

Polygons in original model	8,200 polygons
Polygons in coarse model	32 polygons
Original model file size	694 KB
Coarse model file size	5.96 KB
Number of billboards used in rendering	4
Summation of images size	800 KB



Figure A-10.1: Cone model.



Figure A-10.2: Cone model with different resolution.

11. Face Model

Polygons in original model	28,000 polygons
Polygons in coarse model	2,500 polygons
Original model file size	2.48 MB
Coarse model file size	294 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

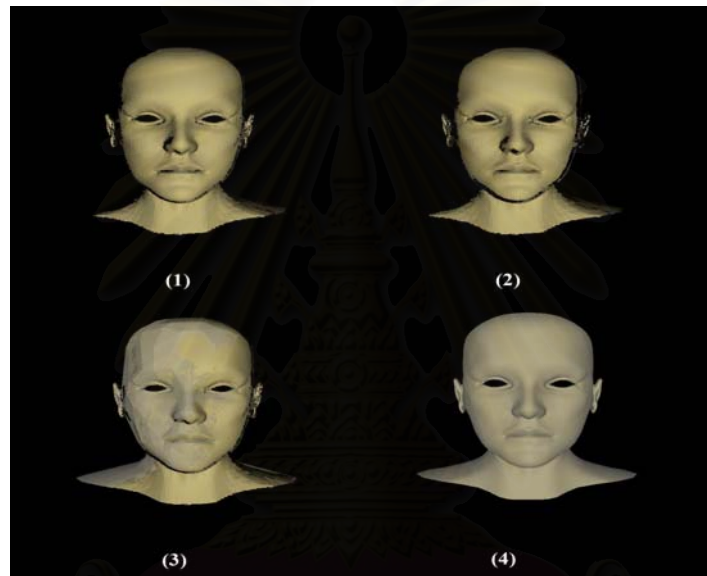


Figure A-11.1: Face model.

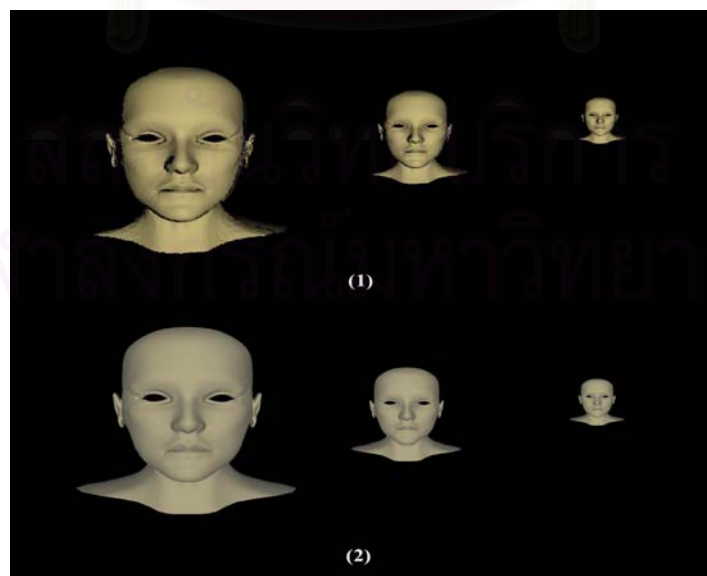


Figure A-11.2: Face model with different resolution.

12. Hand Model

Polygons in original model	12,300 polygons
Polygons in coarse model	770 polygons
Original model file size	935 KB
Coarse model file size	53.8 KB
Number of billboards used in rendering	6
Summation of images size	1.20 MB



Figure A-12.1: Hand model.

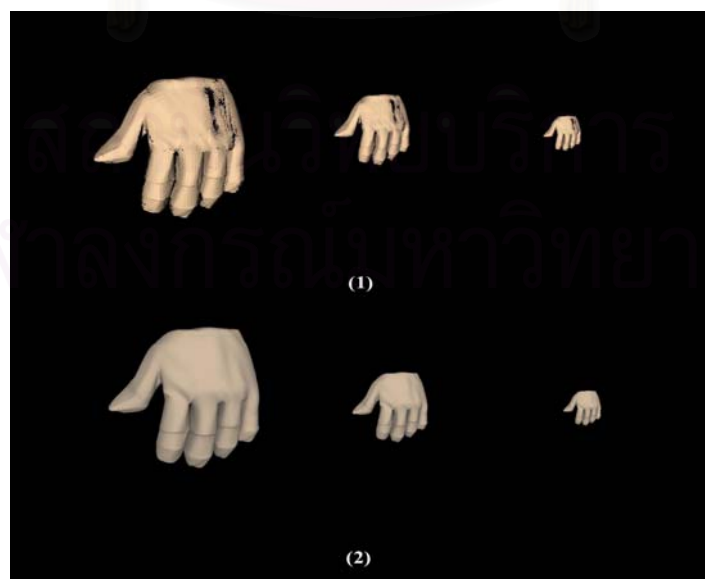


Figure A-12.2: Hand model with different resolution.

13. Head Model

Polygons in original model	29,350 polygons
Polygons in coarse model	1,250 polygons
Original model file size	23.5 MB
Coarse model file size	95 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

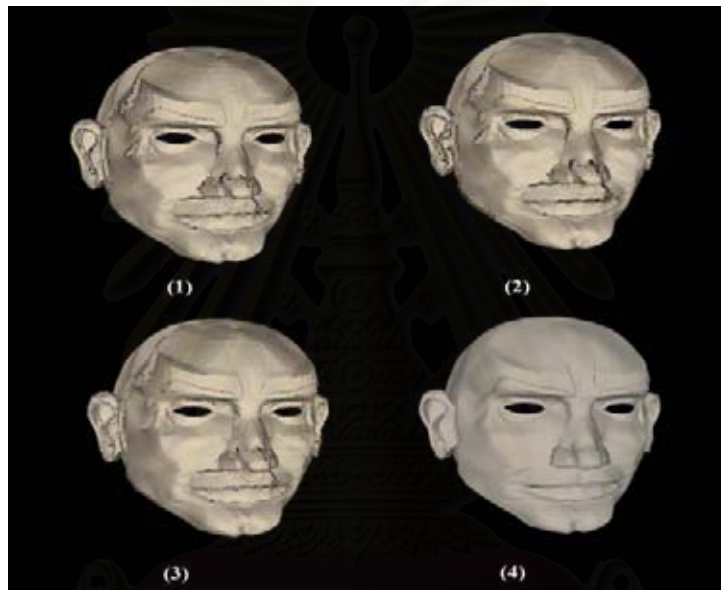


Figure A-13.1: Head model.

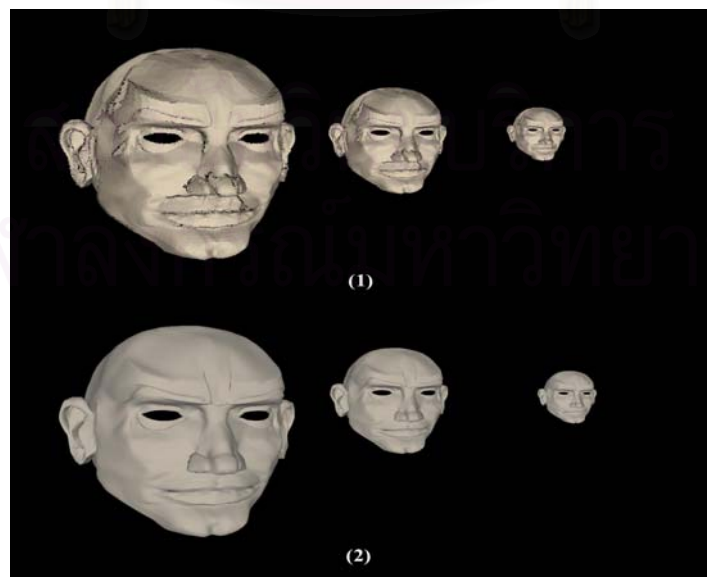


Figure A-13.2: Head model with different resolution.

14. Face2 Model

Polygons in original model	35,000 polygons
Polygons in coarse model	2,900 polygons
Original model file size	3.2 MB
Coarse model file size	400 KB
Number of billboards used in rendering	9
Summation of images size	1.80 MB

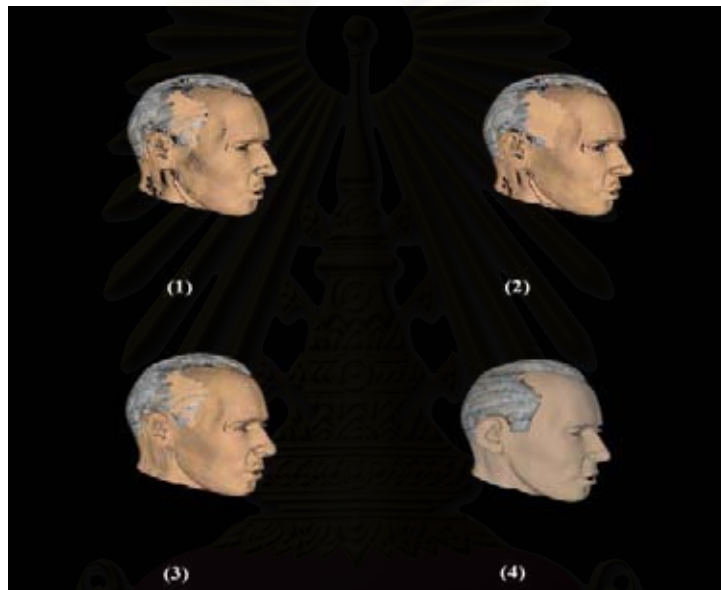


Figure A-14.1: Face2 model.

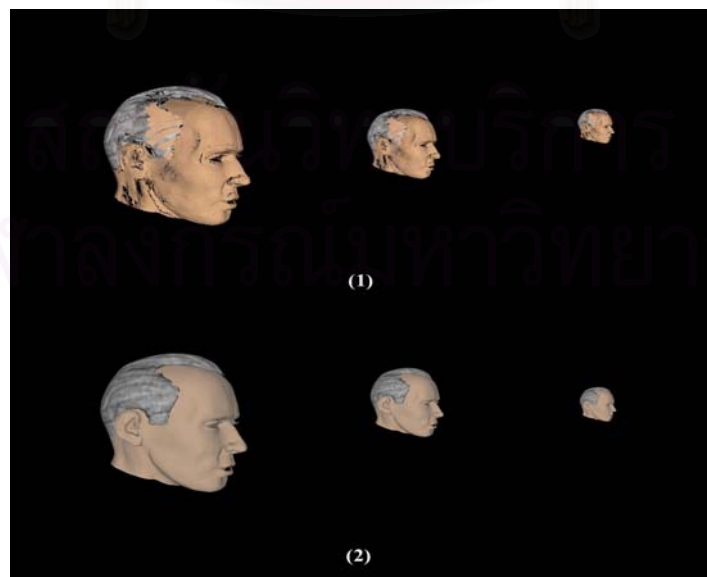


Figure A-14.2: Face2 model with different resolution.

15. Man Model

Polygons in original model	18,000 polygons
Polygons in coarse model	3,100 polygons
Original model file size	3.61 MB
Coarse model file size	157 KB
Number of billboards used in rendering	6
Summation of images size	1.20 MB



Figure A-15.1: Man model.

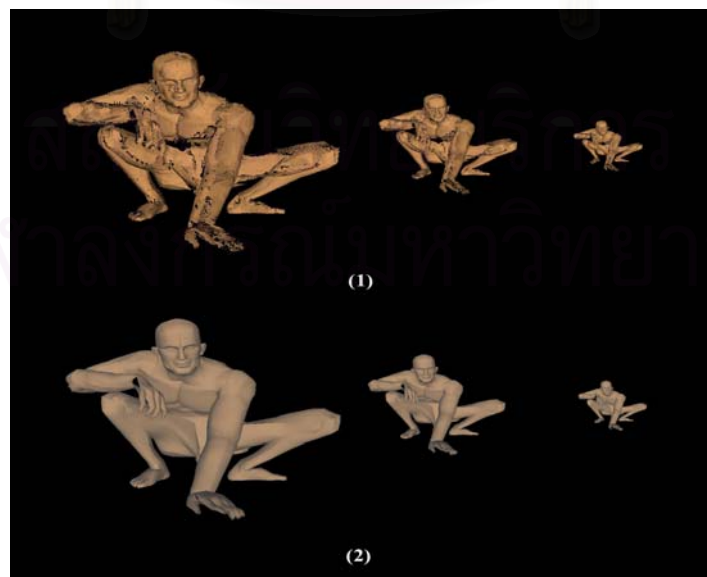


Figure A-15.2: Man model with different resolution.

16. Hand2 Model

Polygons in original model	11,000 polygons
Polygons in coarse model	1,700 polygons
Original model file size	931 KB
Coarse model file size	172 KB
Number of billboards used in rendering	8
Summation of images size	1.60 MB



Figure A-16.1: Hand2 model.



Figure A-16.2: Hand2 model with different resolution.

17. Shape1 Model

Polygons in original model	14,400 polygons
Polygons in coarse model	800 polygons
Original model file size	1.21 MB
Coarse model file size	91.6 KB
Number of billboards used in rendering	4
Summation of images size	800 KB



Figure A-17.1: Shape1 model.



Figure A-17.2: Shape1 model with different resolution.

18. Shape2 Model

Polygons in original model	9,360 polygons
Polygons in coarse model	970 polygons
Original model file size	804 KB
Coarse model file size	120 KB
Number of billboards used in rendering	4
Summation of images size	800KB



Figure A-18.1: Shape2 model.

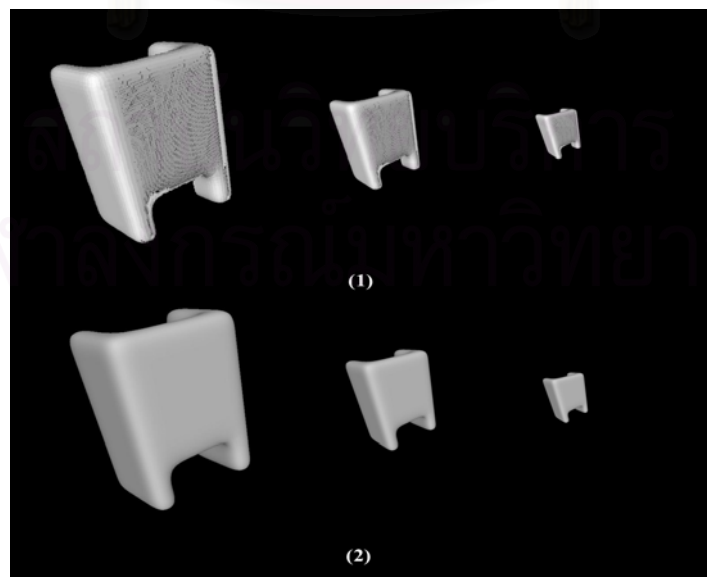


Figure A-18.2: Shape2 model with different resolution.

19. Shape3 Model

Polygons in original model	12,000 polygons
Polygons in coarse model	700 polygons
Original model file size	1.00 MB
Coarse model file size	112 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

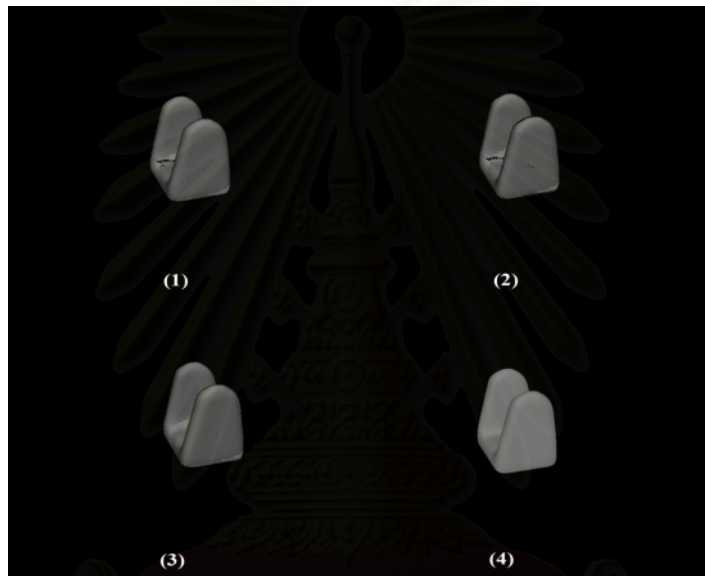


Figure A-19.1: Shape3 model.



Figure A-19.2: Shape3 model with different resolution.

20. Shape4 Model

Polygons in original model	28,000 polygons
Polygons in coarse model	1,700 polygons
Original model file size	2.38 MB
Coarse model file size	162 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

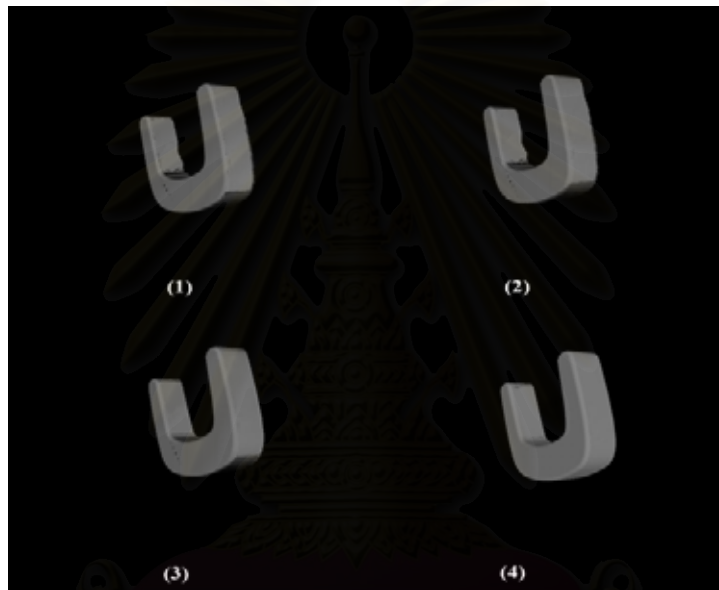


Figure A-20.1: Shape4 model.



Figure A-20.2: Shape4 model with different resolution.

21. Shape5 Model

Polygons in original model	12,400 polygons
Polygons in coarse model	1,380 polygons
Original model file size	1.05 MB
Coarse model file size	194 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

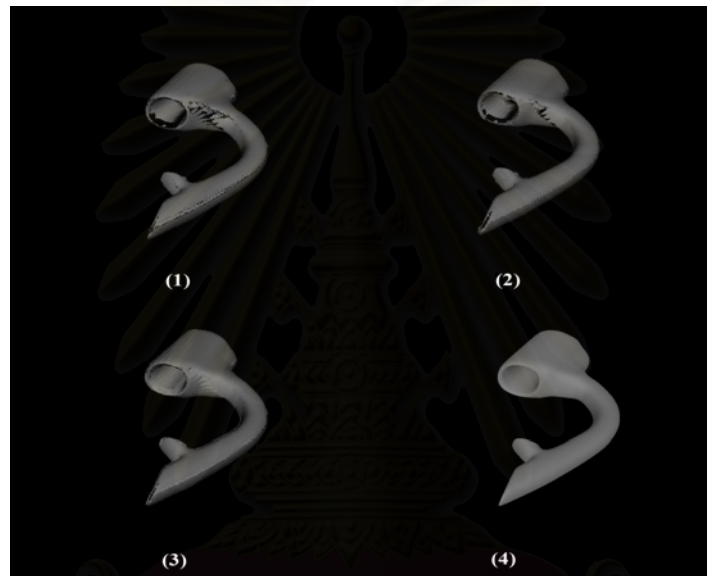


Figure A-21.1: Shape5 model.

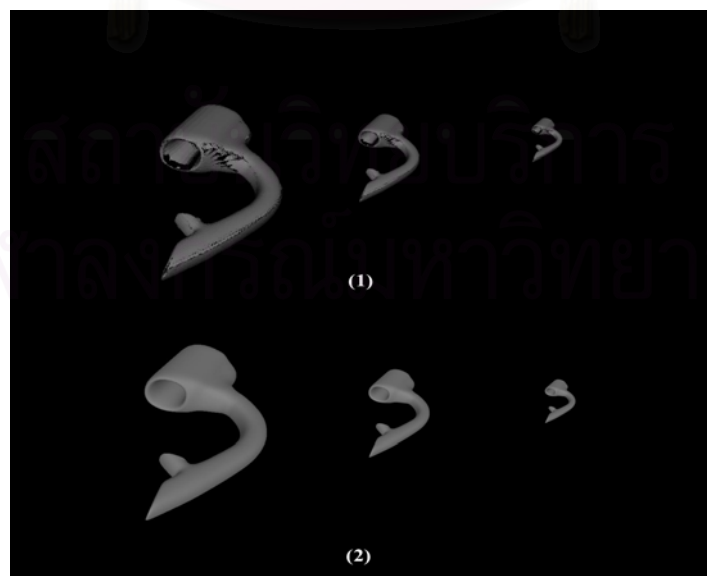


Figure A-21.2: Shape5 model with different resolution.

22. She Model

Polygons in original model	15,000 polygons
Polygons in coarse model	3,000 polygons
Original model file size	1.37 MB
Coarse model file size	270 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

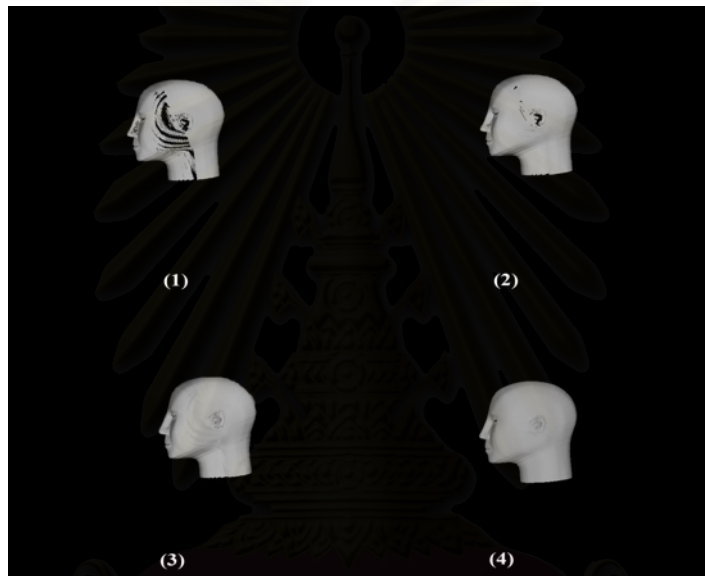


Figure A-22.1: She model.

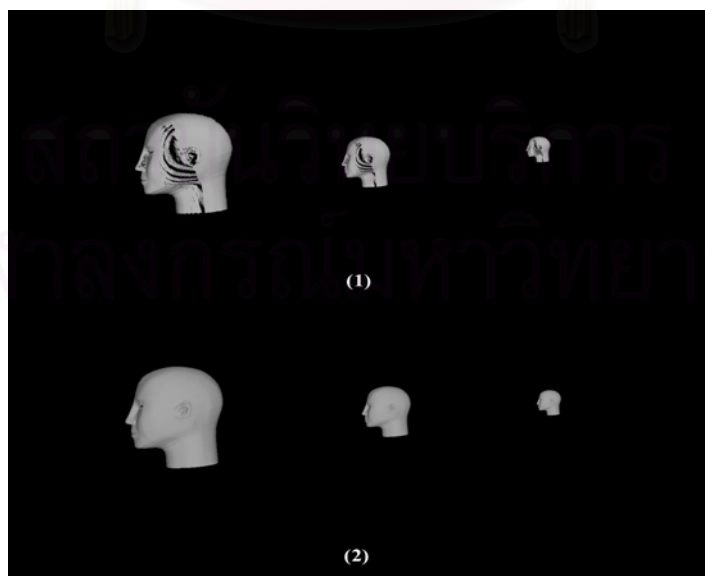


Figure A-22.2: She model with different resolution.

23. Tutan Model

Polygons in original model	18,000 polygons
Polygons in coarse model	4,500 polygons
Original model file size	1.63 MB
Coarse model file size	434 KB
Number of billboards used in rendering	4
Summation of images size	800 KB

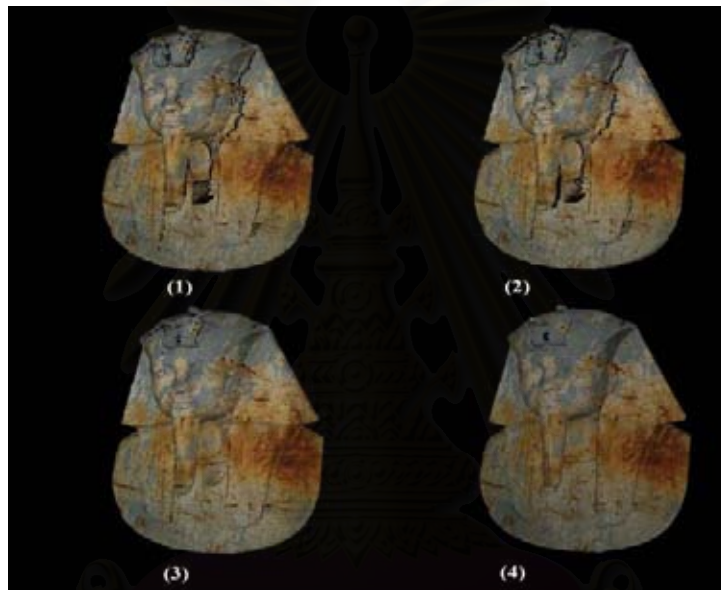


Figure A-23.1: Tutan model.

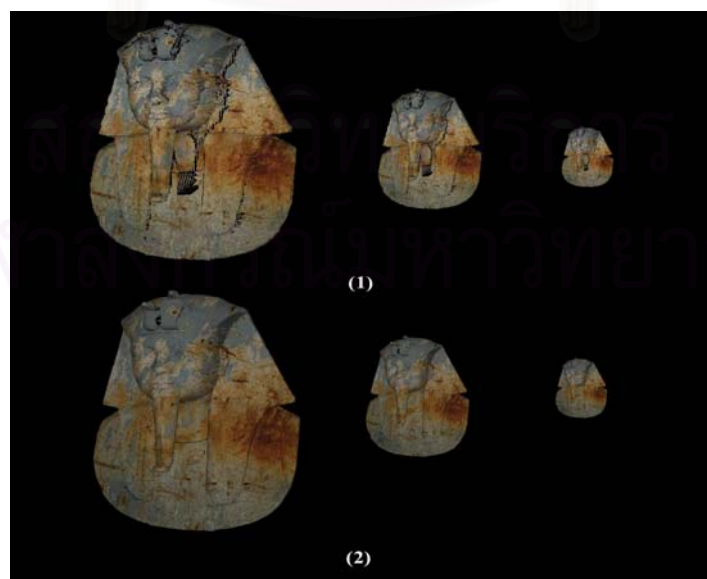


Figure A-23.2: Tutan model with different resolution.

24. Face3 Model

Polygons in original model	13,300 polygons
Polygons in coarse model	3,500 polygons
Original model file size	1.04 MB
Coarse model file size	306 KB
Number of billboards used in rendering	3
Summation of images size	600 KB

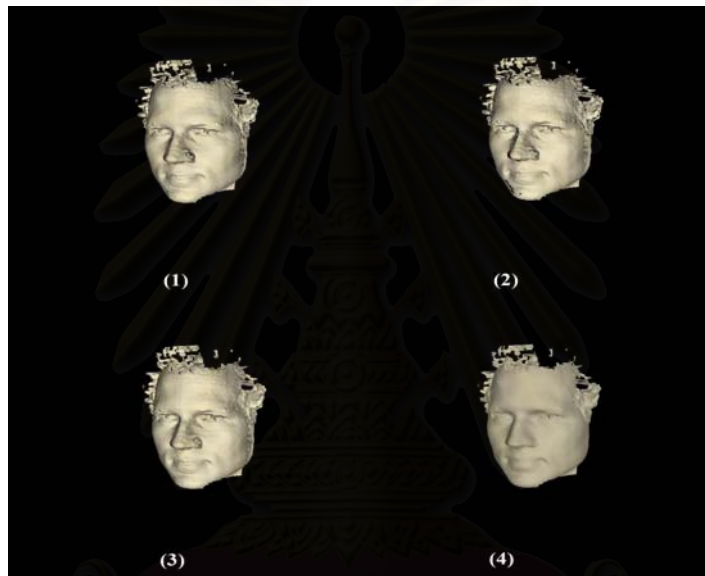


Figure A-24.1: Face3 model.



Figure A-24.2: Face3 model with different resolution.

25. Leg Model

Polygons in original model	19,700 polygons
Polygons in coarse model	1,270 polygons
Original model file size	1.50 MB
Coarse model file size	92.7 KB
Number of billboards used in rendering	4
Summation of images size	800 KB



Figure A-25.1: Leg model.

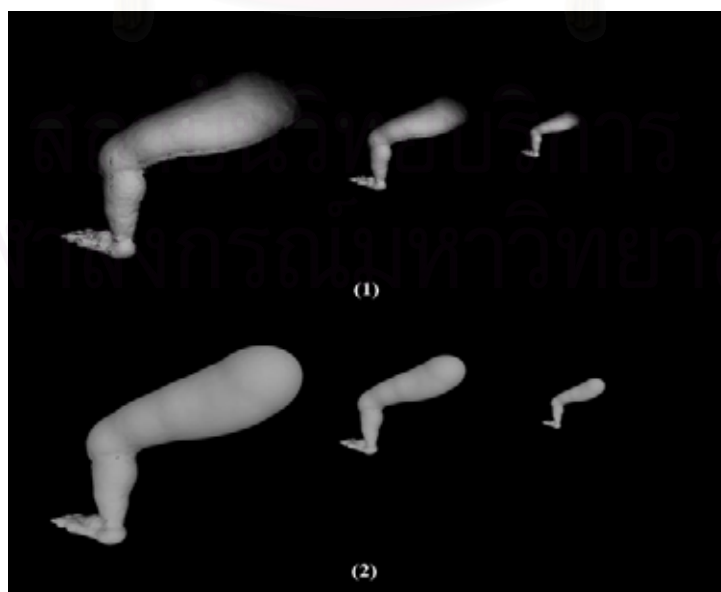


Figure A-25.2: Leg model with different resolution.

26. Tail Model

Polygons in original model	10,000 polygons
Polygons in coarse model	1,200 polygons
Original model file size	815 KB
Coarse model file size	87 KB
Number of billboards used in rendering	3
Summation of images size	600 KB



Figure A-26.1: Tail model.



Figure A-26.2: Tail model with different resolution.

27. Monster Model

Polygons in original model	11,300 polygons
Polygons in coarse model	3,300 polygons
Original model file size	884 KB
Coarse model file size	251 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

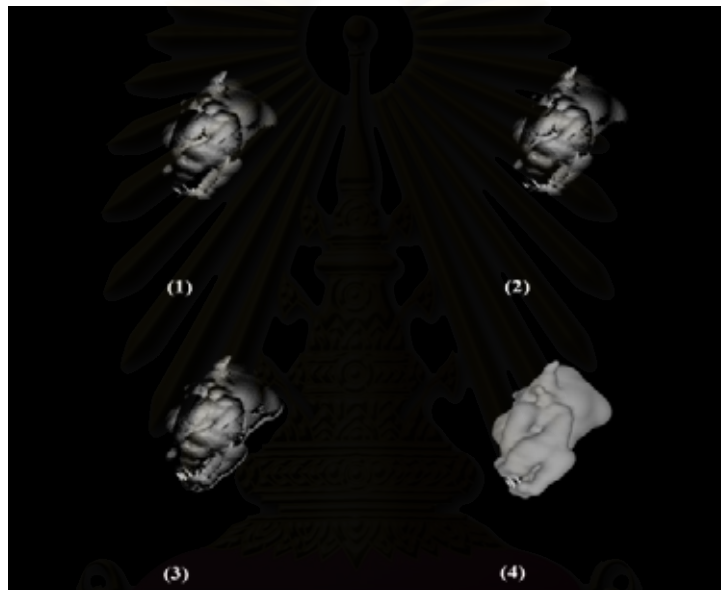


Figure A-27.1: Monster model.

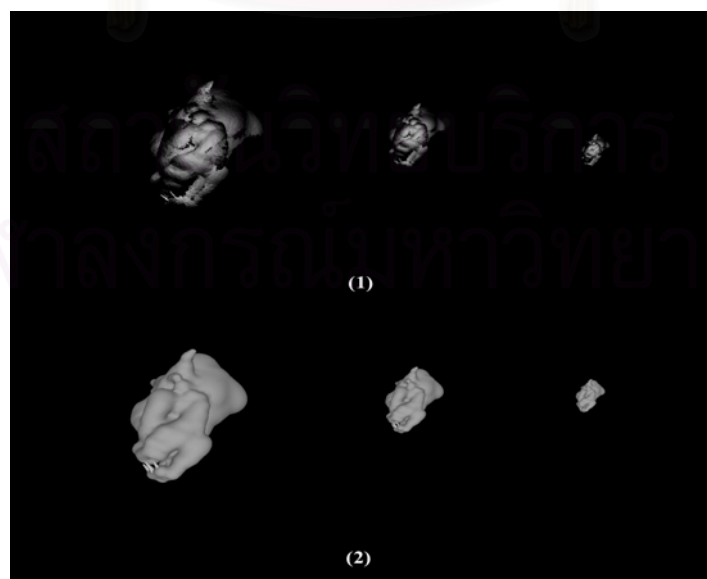


Figure A-27.2: Monster model with different resolution.

28. Body Model

Polygons in original model	28,000 polygons
Polygons in coarse model	3,970 polygons
Original model file size	1.51 MB
Coarse model file size	301 KB
Number of billboards used in rendering	5
Summation of images size	1.00 MB

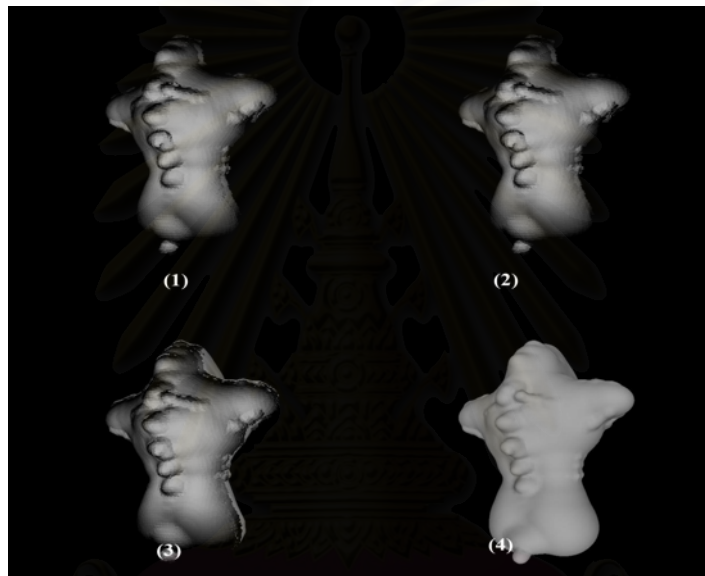


Figure A-28.1: Body model.



Figure A-28.2: Body model with different resolution.

29. Boat Model

Polygons in original model	8,800 polygons
Polygons in coarse model	550 polygons
Original model file size	684 KB
Coarse model file size	42 KB
Number of billboards used in rendering	3
Summation of images size	600 KB

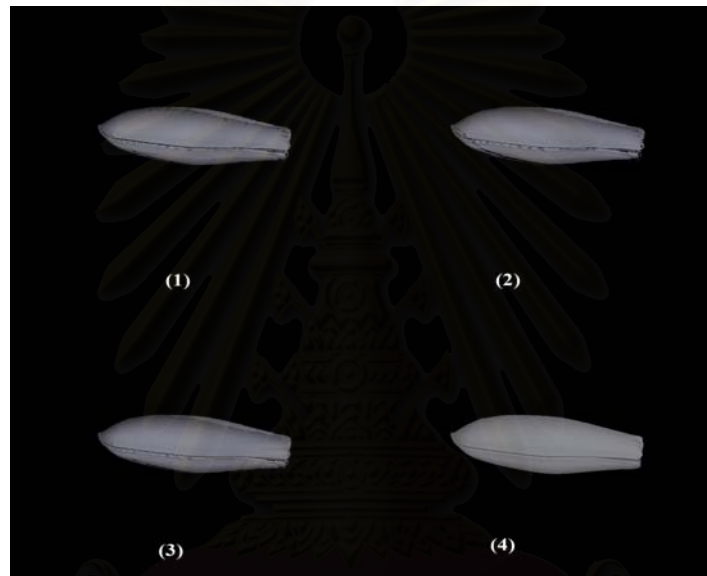


Figure A-29.1: Boat model.



Figure A-29.2: Boat model with different resolution.

30. Fan Model

Polygons in original model	5,120 polygons
Polygons in coarse model	320 polygons
Original model file size	396 KB
Coarse model file size	26 KB
Number of billboards used in rendering	6
Summation of images size	1.20 MB

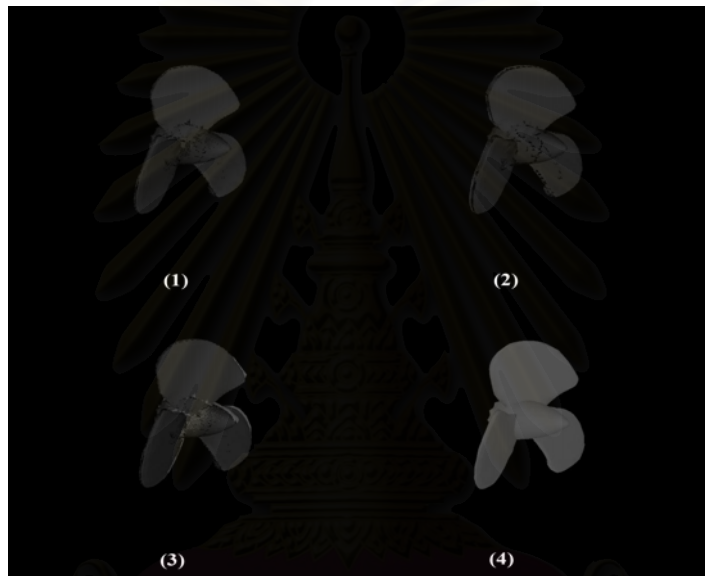


Figure A-30.1: Fan model.

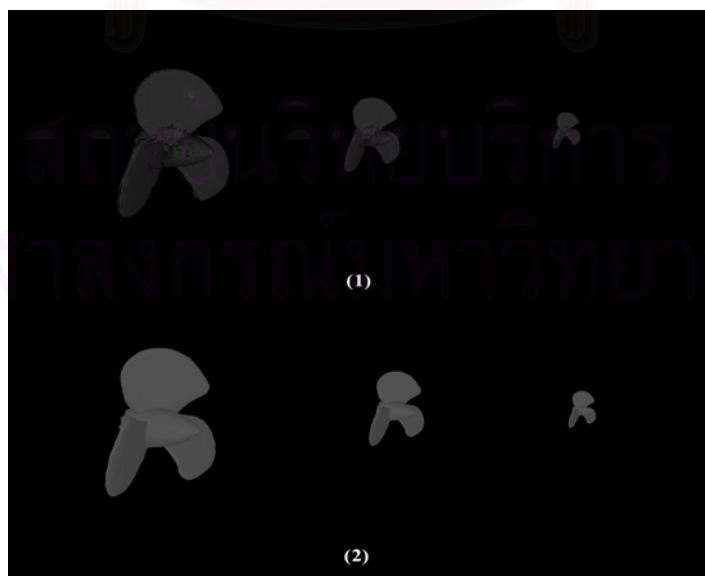


Figure A-30.2: Fan model with different resolution.

REFERENCES

- [1] P. Decaudin and F. Neyret, "Rendering Forest Scenes in Real-Time," *Eurographics Rendering Workshop 2004*, pp. 93-102, 2004.
- [2] P. Vichitvejpaisal and P. Kanongchaiyos, "Enhanced Billboards for Model Simplification," *WSCG 2006*, pp. 125-132, 2006.
- [3] Durand and Cutler, "MIT computer graphics course slide," 2005.
- [4] R. Fernando, "Programming the GPU: High Level Shading Language," 2005.
- [5] A. Watt, "3D Computer Graphics," *Addison-Wesley*, pp. 409-435, 1993
- [6] B. Baxter, A. Sud, N. Govindraj, and D. Manocha, "Gigawalk: Interactive walkthrough of complex 3d environments," *Proc. of Eurographics Workshop on Rendering*, pp. 203-214, 2002.
- [7] M. Garland and P. Heckbert, "Surface simplification using quadric error bounds," *Proc. of ACM SIGGRAPH*, pp. 209-216, 1997.
- [8] H. Hoppe, "Progressive meshes," *ACM SIGGRAPH 1996*, pp. 99-108, 1996.
- [9] H. Hoppe, "View dependent refinement of progressive meshes," *ACM SIGGRAPH Conference Proceedings*, pp. 189-198, 1997.
- [10] J. Blinn, "Simulation of Wrinkled Surfaces," *SIGGRAPH 78*, pp. 286-292., 1978.
- [11] M. Peercy, J. Airey, and B. Cabral, "Efficient bump mapping hardware," *SIGGRAPH '97*, pp. 303-306, 1997.
- [12] N. Max, "Horizon mapping: shadows for bump-mapped surfaces," *The Visual Computer*, Volume 4, pp. 109-117, 1988.
- [13] F. a. Policarpo, M. M. Oliveira, and J. a. L. D. Comba, "Real-Time Relief Mapping on Arbitrary Polygonal Surfaces," *ACM SIGGRAPH 2005 Symposium on Interactive 3D Graphics and Games 2005*, pp. 935-935, 2005.
- [14] M. Pharr and P. Harahan, "Geometry caching for ray-tracing displacement maps," *Eurographics Rendering Workshop 1996*, pp. 31-40, 1996.

- [15] J. Hirche, A. Ehlert, S. Guthe, and M. Doggett, "Hardware accelerated per-pixel displacement mapping," *Graphics Interface*, pp. 153 – 158, 2004.
- [16] W. Heidrich and H. P. Seidel, "Ray-tracing procedural displacement shaders," *Graphics Interface*, pp. 8-16, 1998.
- [17] G. Schaufler and M. Priglinger, "Efficient displacement mapping by image warping," *Eurographics Rendering Workshop 1999*, pp. 175-186, 1999.
- [18] W. Donnelly, "Per-Pixel Displacement Mapping with Distance Functions," *GPU GEMS 2, Addison-Wesley*, pp. 123-135, 2005.
- [19] X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H.-Y. Shum, "Generalized Displacement Maps," *Eurographics Rendering Workshop 2004*, pp. 227-233, 2004.
- [20] L. Wang, X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H. Shum, "View-dependent displacement mapping," *ACM Trans. Graph.* 22, pp. 334-339, 2003.
- [21] A. Meyer and F. Neyret, "Interactive volumetric textures," *Eurographics Rendering Workshop*, pp. 157-168, 1998.
- [22] A. Meyer, F. Neyret, and P. Poulin, "Interactive rendering of trees with shading and shadows," *Eurographics Workshop on Rendering (Jul 2001)*, pp. 183-196, 2001.
- [23] X. Decoret, F. Durand, F. o. X. Sillion, and J. Dorsey, "Billboard Clouds for Extreme Model Simplification," *SIGGRAPH 03*, pp. 689-696, 2003.

BIOGRAPHY

Mr. Phongvarin Vichitvejpaisal was born on December 27, 1982 in Bangkok. He studied at Suankularb Wittayalai School. He graduated from Computer Engineering, Chulalongkorn University in 2004



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย