AN EFFICIENT REAL-TIME TERRAIN DATA ORGANIZATION AND VISUALIZATION ALGORITHM BASED ON ENHANCED TRIANGLE-BASED LEVEL OF DETAIL TECHNIQUE

MUHAMAD NAJIB BIN ZAMRI

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

Faculty of Computer Science and Information System
Universiti Teknologi Malaysia

MAY 2006
To

my beloved father and mother, Zamri Ismail and Rashidah Shariff;
my brothers, Azainizam, Shahirman and Hanafi;
my sisters, Anira, Aziha and Aizan; and
my friends
ACKNOWLEDGEMENT

First of all, thanks to Allah SWT, the Creator, for giving me the strength to complete this thesis. I would like to express my appreciation to the Ministry of Science, Technology and Innovation (MOSTI) for providing me the financial support to further my education to this level under National Science Fellowship (NSF) program. Obviously, without their help, this journey would not have allowed.

I would like to express my sincere gratitude to my main supervisor, Associate Professor Safie Mat Yatim and my co-supervisor, Mr. Noor Azam Md. Sheriff, for their guidance and support through my studies as well as their constructive criticism of my research. This thesis would not have been possible without them. I would like to thank lecturers that involved and gave their brilliant ideas and solutions for this research especially Associate Professor Sarudin Kari, Associate Professor Daut Daman, Mr. Nik Isrozaidi Nik Ismail and Mr. Mohd Shafry Mohd Rahim.

Furthermore, I would also like to acknowledge my fellow postgraduate students, colleagues and housemates who have made my lives so enjoyable. They are Irwandi Hipni, Yusrin, Hamimah, Mun San, Weng, Baladave, Sandilian, Kim Heok, Adlur, Fendi, Amri, Mohd Hafiz, Asyraf and Asrulakmar. Special thanks go to foreign researchers that help me in explaining the related technical processes and developing the prototype system. They are Stefan Röttger, Mark Duchaineau, Peter Lindstrom, Chris Cookson, ZhaoYoubing, Thatcher Ulrich and Ulf Assarsson.

Finally, I am grateful to my parents and my family for their support and patience during my stay at the Universiti Teknologi Malaysia.
The massive volume of data involved in the development of a real-time terrain visualization and navigation system is inevitable. Based upon the current off-the-shelf hardware capability, it is impossible to process the amount of data using the conventional approach. This is due to the fact that the amount of data to be processed exceeds the capacity that can be loaded into main memory. This problem is further compounded by other hardware constraints such as memory bus speed and data transfer bandwidth from the main memory to the graphics card. Consequently, this limitation has affected the total system performance. The triangle-based level of detail (LOD) technique has been developed in order to reduce the drawback but it still suffers from the main memory constraint and slow data loading. The purpose of this research is to design, develop and implement an algorithm for enhancing the rendering efficiency of triangle-based LOD technique. A prototype system has been developed using digital elevation data for testing purposes. The system was evaluated based on the following criteria, namely: data size, processing time for data partitioning, memory usage, loading time, frame rate, triangle count and geometric throughput. According to the results obtained during the pre-processing, the partitioning of digital elevation data into tiles has successfully reduced the data size although it required a longer processing time. During the run-time processing, the integration of dynamic tile loading scheme, view frustum culling and enhanced triangle-based LOD technique has produced encouraging results with significant overall improvement as compared to the techniques that have been developed earlier. The proposed algorithm in this thesis is very practical in developing interactive graphics applications, which consider real-time rendering as one of the important elements.
ABSTRAK

# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE PAGE</td>
<td></td>
<td>i</td>
</tr>
<tr>
<td>DECLARATION OF ORIGINALITY AND EXCLUSIVENESS</td>
<td></td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td></td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td></td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td></td>
<td>v</td>
</tr>
<tr>
<td>ABSTRAK</td>
<td></td>
<td>vi</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td></td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td></td>
<td>xii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td></td>
<td>xiii</td>
</tr>
<tr>
<td>LIST OF ALGORITHMS</td>
<td></td>
<td>xvi</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td></td>
<td>xviii</td>
</tr>
<tr>
<td>LIST OF APPENDICES</td>
<td></td>
<td>xx</td>
</tr>
</tbody>
</table>

## 1 INTRODUCTION

1.1 Introduction

1.2 Problem Background

1.2.1 Massive Terrain Dataset

1.2.2 Geometric Complexity

1.2.3 Limited Capability of Triangle-Based LOD Techniques

1.3 Problem Statements

1.4 Goal
1.5 Objectives 9
1.6 Scopes 9
1.7 Organization of the Thesis 10

2 LITERATURE REVIEW 11
2.1 Introduction 11
2.2 Spatial Data Structures 12
  2.2.1 Kd-Tree 13
  2.2.2 Quadtree 15
  2.2.3 R-Tree 16
  2.2.4 Tiled Block 18
  2.2.5 Comparative Studies 19
2.3 Visibility Culling Methods 20
  2.3.1 Types of Culling Methods 21
    2.3.1.1 Backface Culling 21
    2.3.1.2 Small Feature Culling 22
    2.3.1.3 Occlusion Culling 23
    2.3.1.4 View Frustum Culling 24
    2.3.1.5 Comparative Studies 25
  2.3.2 Intersection Test Techniques 26
    2.3.2.1 Axis Aligned Bounding Box 27
    2.3.2.2 Oriented Bounding Box 27
    2.3.2.3 Bounding Sphere 29
    2.3.2.4 k-Discrete Orientation Polytope 30
    2.3.2.5 Comparative studies 31
2.4 View-Dependent LOD Techniques 32
  2.4.1 Terrain Representation 32
    2.4.1.1 Regular Grid 32
    2.4.1.2 Triangulated Irregular Network 33
    2.4.1.3 Advantages and Disadvantages 34
2.4.2 Geometric Primitives
   2.4.2.1 Independent Triangles 35
   2.4.2.2 Triangle Fans 36
   2.4.2.3 Triangle Strips 36
   2.4.2.4 Comparative Studies 37

2.4.3 Previous Works 37
   2.4.3.1 Real-time Optimally Adapting Meshes 38
   2.4.3.2 Real-time Generation of Continuous LOD for Height Fields 43
   2.4.3.3 View-Dependent Progressive Meshes 45
   2.4.3.4 Out-of-Core Terrain Visualization 48
   2.4.3.5 Comparative Studies 52

2.5 Summary 53

3 METHODOLOGY 55

3.1 Introduction 55

3.2 Partitioning Terrain Data 58
   3.2.1 Extracting Data 58
   3.2.2 Splitting Data into Tiles 59

3.3 Loading Terrain Tiles 63

3.4 Culling Invisible Terrain Tiles 64
   3.4.1 Plane Extraction 64
   3.4.2 Intersection Test 67

3.5 Simplifying Geometry of Visible Terrain Tiles 68
   3.5.1 Distance Calculation 69
   3.5.2 Subdivision Test 69
   3.5.3 Generation of Triangle Fans 70

3.6 Summary 71
4 IMPLEMENTATION

4.1 Introduction

4.2 Pre-Processing

4.2.1 Reading a DEM File and Creating TXT Files

4.2.2 Reading TXT Files and Creating TILE Files

4.3 Run-time Processing

4.3.1 Creation of Classes

4.3.2 Initialization

4.3.3 Rendering

4.3.3.1 Configuring Information for Camera

4.3.3.2 Loading and Unloading Terrain Tiles

4.3.3.3 Calculating Frustum’s Planes

4.3.3.4 Removing Unseen Terrain Tiles

4.3.3.5 Generating Quadtree Traversal

4.3.3.6 Generating Triangle Fans

4.3.4 Interaction

4.3.5 Shutdown

4.4 Summary

5 RESULTS, ANALYSIS AND DISCUSSIONS

5.1 Introduction

5.2 System Specifications

5.3 Testing Data

5.4 Screenshots from the Prototype System

5.5 Evaluations of the System

5.5.1 Pre-Processing
5.5.2 Run-time Processing 105
  5.5.2.1 Memory Usage 106
  5.5.2.2 Loading Time 108
  5.5.2.3 Frame Rate 109
  5.5.2.4 Triangle Count 111
  5.5.2.5 Geometric Throughput 113
5.6 Comparison with Previous Techniques 114
  5.6.1 Memory Usage 115
  5.6.2 Loading Time 116
  5.6.3 Frame Rate 117
  5.6.4 Triangle Count 118
5.7 Summary 120

6 CONCLUSION 121
  6.1 Conclusion 121
  6.2 Recommendations 124

REFERENCES 126
APPENDICES A - C 133 - 139
## LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Types of busses and the corresponding memory bandwidth and geometric</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>throughput (Hill, 2002; Peng et al., 2004)</td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Advantages and disadvantages of visibility culling methods</td>
<td>25</td>
</tr>
<tr>
<td>2.2</td>
<td>Comparison of intersection test techniques</td>
<td>31</td>
</tr>
<tr>
<td>2.3</td>
<td>Comparison among the geometric primitives</td>
<td>37</td>
</tr>
<tr>
<td>2.4</td>
<td>Comparison of view-dependent LOD techniques</td>
<td>53</td>
</tr>
<tr>
<td>3.1</td>
<td>Clipping planes with corresponding co-efficients</td>
<td>66</td>
</tr>
<tr>
<td>4.1</td>
<td>Data structure</td>
<td>75</td>
</tr>
<tr>
<td>4.2</td>
<td>Declaration of variables</td>
<td>80</td>
</tr>
<tr>
<td>5.1</td>
<td>Hardware specifications</td>
<td>99</td>
</tr>
<tr>
<td>5.2</td>
<td>Software specifications</td>
<td>99</td>
</tr>
<tr>
<td>5.3</td>
<td>Testing data</td>
<td>100</td>
</tr>
<tr>
<td>5.4</td>
<td>Environment settings for testing the proposed algorithm</td>
<td>101</td>
</tr>
<tr>
<td>5.5</td>
<td>The results of time required for converting DEM data and the</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td>corresponding file size for number of vertices per tile: 129 x 129</td>
<td></td>
</tr>
<tr>
<td></td>
<td>and 257 x 257</td>
<td></td>
</tr>
<tr>
<td>5.6</td>
<td>Number of generated tiles</td>
<td>104</td>
</tr>
<tr>
<td>5.7</td>
<td>Environment settings for comparison purpose</td>
<td>115</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Visualization of full resolution terrain model</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>(number of vertices = 16,848,344,430 and number of triangles = 33,695,786,468)</td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Kd-tree</td>
<td>14</td>
</tr>
<tr>
<td>2.2</td>
<td>Quadtree</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>R-tree</td>
<td>17</td>
</tr>
<tr>
<td>2.4</td>
<td>Partitioning data into tiles</td>
<td>18</td>
</tr>
<tr>
<td>2.5</td>
<td>Tree structure for tiles</td>
<td>19</td>
</tr>
<tr>
<td>2.6</td>
<td>Vector angle between view direction and polygons normal</td>
<td>21</td>
</tr>
<tr>
<td>2.7</td>
<td>Small feature culling a) OFF, b) ON</td>
<td>22</td>
</tr>
<tr>
<td>2.8</td>
<td>Occlusion culling of occluded object</td>
<td>23</td>
</tr>
<tr>
<td>2.9</td>
<td>Camera’s view frustum</td>
<td>24</td>
</tr>
<tr>
<td>2.10</td>
<td>The red objects lie completely outside the view-frustum and will be discarded from rendering when view frustum culling is performed</td>
<td>24</td>
</tr>
<tr>
<td>2.11</td>
<td>AABB of an object</td>
<td>27</td>
</tr>
<tr>
<td>2.12</td>
<td>OBB of a rotated object</td>
<td>28</td>
</tr>
<tr>
<td>2.13</td>
<td>BS of an object</td>
<td>29</td>
</tr>
<tr>
<td>2.14</td>
<td>Aircraft model and the corresponding k-dops</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>a) Level 1, b) Level 2, c) Level 3, d) Level 4</td>
<td></td>
</tr>
<tr>
<td>2.15</td>
<td>Variations of grid representations</td>
<td>33</td>
</tr>
<tr>
<td>2.16</td>
<td>TIN representation</td>
<td>33</td>
</tr>
<tr>
<td>2.17</td>
<td>Independent triangles</td>
<td>35</td>
</tr>
<tr>
<td>2.18</td>
<td>Triangle fan</td>
<td>36</td>
</tr>
</tbody>
</table>
2.19 Triangle strip  36
2.20 Levels 0-5 of a triangle bintree  38
2.21 Split and merge operations on a bintree triangulation  39
2.22 Forced splitting process  39
2.23 Wedgies in ROAM implementation  40
2.24 A sample triangulation of a 9 x 9 height field
   a) quadtree matrix, b) the corresponding quadtree structure  43
2.25 Measuring surface roughness  44
2.26 Vertex split and edge collapse operations  45
2.27 Steps in pre-processing block-based simplification  46
2.28 Interleaved quadtree structure  49
2.29 Four levels of longest-edge bisection operation  50
3.1 Research framework  56
3.2 Overlapping tile  62
3.3 Dynamic load management  63
3.4 Six planes of view frustum  64
3.5 Plane-sphere intersection test for view frustum
   culling technique a) sample scene, b) intersection test, c) result after culling  68
3.6 Distance (l) and edge length (d)  69
3.7 Generation of triangle fans  70
3.8 Terrain tiles after applying view-dependent LOD technique  71
4.1 Menu for partitioning DEM data  73
4.2 Structure of a DEM file  74
4.3 First data point for DEM data a) West of the central meridian b) East of the central meridian  77
4.4 Parameters after extraction of important header information  78
4.5 The indices and the corresponding elevation values  79
4.6 List of generated TILE files 85
4.7 Structure in a TILE file 86
4.8 Class diagram 88
4.9 Sixteen possibilities of generating triangle fans 95
5.1 Small dataset a) wireframe, b) textured 102
5.2 Moderate dataset a) wireframe, b) textured 102
5.3 Large dataset a) wireframe, b) textured 102
5.4 Total time for converting DEM datasets to TILE files 104
5.5 Memory usage for the proposed system 106
5.6 The results of memory usage 107
5.7 The results of loading time 108
5.8 The results of average frame rate based on different FOV 109
5.9 The results of average frame rate based on different length of view frustum 110
5.10 The results of average triangle count based on different FOV 111
5.11 The results of average triangle count based on different length of view frustum 111
5.12 The results of frame rate versus triangle count 112
5.13 The results of average geometric throughput based on different FOV 113
5.14 The results of average geometric throughput based on different length of view frustum 114
5.15 Comparison based on memory usage 116
5.16 Comparison based on loading time 117
5.17 Comparison based on frame rate 118
5.18 Comparison based on triangle count 119
### LIST OF ALGORITHMS

<table>
<thead>
<tr>
<th>ALGORITHM NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Pseudo code for split queue</td>
<td>41</td>
</tr>
<tr>
<td>2.2</td>
<td>Pseudo code for merge queue</td>
<td>42</td>
</tr>
<tr>
<td>2.3</td>
<td>Pseudo code for selective refinement</td>
<td>47</td>
</tr>
<tr>
<td>2.4</td>
<td>Pseudo code for force vsplit</td>
<td>48</td>
</tr>
<tr>
<td>2.5</td>
<td>Pseudo code for mesh refinement (<em>mesh-refine</em>)</td>
<td>51</td>
</tr>
<tr>
<td>2.6</td>
<td>Pseudo code for sub mesh refinement (<em>submesh-refine</em>)</td>
<td>51</td>
</tr>
<tr>
<td>2.7</td>
<td>Pseudo code for generating triangle strips (<em>trianglestrip-append</em>)</td>
<td>52</td>
</tr>
<tr>
<td>4.1</td>
<td>Pseudo code for obtaining maximum number of vertices per side for original terrain data</td>
<td>81</td>
</tr>
<tr>
<td>4.2</td>
<td>Pseudo code for assigning new maximum number of vertices per side for terrain data</td>
<td>81</td>
</tr>
<tr>
<td>4.3</td>
<td>Pseudo code for determining the maximum number of vertices per side for tiles in vertical direction (<em>V_{t,v_max}</em>)</td>
<td>82</td>
</tr>
<tr>
<td>4.4</td>
<td>Pseudo code for obtaining starting point and existence status for each tile</td>
<td>83</td>
</tr>
<tr>
<td>4.5</td>
<td>Pseudo code for partitioning data and creating TILE files</td>
<td>84</td>
</tr>
<tr>
<td>4.6</td>
<td>Pseudo code for obtaining starting coordinates of existing tile</td>
<td>85</td>
</tr>
</tbody>
</table>
4.7 Pseudo code for obtaining the current tile index based on camera position 91
4.8 Pseudo code for obtaining the terrain indices to be loaded 92
4.9 Pseudo code for detecting the visibility status and assigning flag for loaded tiles 93
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>ABBREVIATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>Two-dimensional</td>
</tr>
<tr>
<td>3D</td>
<td>Three-dimensional</td>
</tr>
<tr>
<td>AABB</td>
<td>Axis Aligned Bounding Box</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BS</td>
<td>Bounding Sphere</td>
</tr>
<tr>
<td>CBIS</td>
<td>Computer-based information system</td>
</tr>
<tr>
<td>CFD</td>
<td>Computational fluid dynamics</td>
</tr>
<tr>
<td>CPU</td>
<td>Central processing unit</td>
</tr>
<tr>
<td>DEM</td>
<td>Digital Elevation Model</td>
</tr>
<tr>
<td>DSS</td>
<td>Decision Support System</td>
</tr>
<tr>
<td>DTED</td>
<td>Digital Terrain Elevation Data</td>
</tr>
<tr>
<td>ESS</td>
<td>Executive Support System</td>
</tr>
<tr>
<td>FOV</td>
<td>Field of view</td>
</tr>
<tr>
<td>FPS</td>
<td>Frames per second</td>
</tr>
<tr>
<td>GIS</td>
<td>Geographic Information System</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics processing unit</td>
</tr>
<tr>
<td>GRASS</td>
<td>Geographic Resources Analysis Support System</td>
</tr>
<tr>
<td>IS</td>
<td>Information system</td>
</tr>
<tr>
<td>KMS</td>
<td>Knowledge Management System</td>
</tr>
<tr>
<td>LOD</td>
<td>Level of detail</td>
</tr>
<tr>
<td>MBB</td>
<td>Minimal bounding box</td>
</tr>
<tr>
<td>MBR</td>
<td>Minimal bounding rectangle</td>
</tr>
<tr>
<td>MIS</td>
<td>Management Information System</td>
</tr>
<tr>
<td>NE</td>
<td>Northeast</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>NW</td>
<td>Northwest</td>
</tr>
<tr>
<td>OAS</td>
<td>Office Automation System</td>
</tr>
<tr>
<td>OBB</td>
<td>Oriented Bounding Box</td>
</tr>
<tr>
<td>OS</td>
<td>Operating system</td>
</tr>
<tr>
<td>PC</td>
<td>Personal computer</td>
</tr>
<tr>
<td>RAPID</td>
<td>Rapid and Accurate Polygon Interference Detection</td>
</tr>
<tr>
<td>ROAM</td>
<td>Real-time Optimally Adapting Meshes</td>
</tr>
<tr>
<td>SE</td>
<td>Southeast</td>
</tr>
<tr>
<td>SW</td>
<td>Southwest</td>
</tr>
<tr>
<td>TIN</td>
<td>Triangulated irregular network</td>
</tr>
<tr>
<td>USGS</td>
<td>United States Geological Survey</td>
</tr>
<tr>
<td>UTM</td>
<td>Universal Transverse Mercator</td>
</tr>
<tr>
<td>VDPM</td>
<td>View-dependent progressive meshes</td>
</tr>
<tr>
<td>ViRGIS</td>
<td>Virtual Reality GIS</td>
</tr>
</tbody>
</table>
LIST OF APPENDICES

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>DEM Elements Logical Record Type A</td>
<td>133</td>
</tr>
<tr>
<td>B</td>
<td>DEM Elements Logical Record Type B</td>
<td>138</td>
</tr>
<tr>
<td>C</td>
<td>List of Published Paper Works</td>
<td>139</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Introduction

Computer-based information system (CBIS) has changed the people works from traditional manual system to an automation system that provides more effective and systematic way in solving specific problems. In addition, the works can be done and completed much faster than the previous approach. Thus, time consumption can be reduced and managed efficiently. This is due to the rapid development of computer's technology over the last few years. Several types of CBISs have been developed and those systems have contributed to the evolution of information system. Among of them are Management Information System (MIS), Office Automation System (OAS), Executive Support System (ESS), Decision Support System (DSS), Knowledge Management System (KMS), Geographic Information System (GIS) and Expert System (Gadish, 2004).

Geographic information system (GIS) is one of the important developments of CBIS that combines digital map and geographic data; and facilitate assessments of related characteristics of people, property and environmental factors. In general, GIS is a set of computer tools for collecting, storing, retrieving, transforming and displaying data from the real world for a particular set of purposes or application domains (Burrough and McDonnell, 1998). There are two main elements in developing GIS applications: database and visualization. This system provides an opportunity to store both kinds of spatial data (location-based data) and non-spatial
data (attribute data) in a single system. In current GIS applications, these two types of data are often stored in two separate databases. However, these databases are linked or joined together later depending on the particular characteristic.

Computer graphics disciplines are exploited in visualizing the GIS data. It is useful for analyzing and verifying the related activities in terms of accuracy, efficiency and other factors. For this reason, GIS is becoming increasingly important to a growing number of activities in many sectors including urban planning, flood planning, oil exploration, evaluations of vegetation, soil, waterway and road patterns (Fritsch, 1996; Burrough and McDonnell, 1998).

Lacking in several aspects such as user interaction (limited to panning and zooming operations), presentation (points, lines and symbols) and interpretation in two-dimensional GIS (2D GIS) was lead to the development of higher dimensional GIS especially three-dimensional GIS (3D GIS), which is also known as Virtual GIS (Burrough and McDonnell, 1998). The use of colors and symbols representation in previous GIS is replaced with 3D photo-textured models, which improves the level of understanding of the system. Moreover, Virtual GIS has the ability to have detailed 3D views and provide real-time navigation whether fly-through or walkthrough for analyzing large volumes of spatial data and related information. At the same time, the operations used in 2D GIS can be integrated easily into the system (Noor Azam and Ezreen Elena, 2003).

Real-time terrain visualization is one of the subfield in GIS that attracts a number of researchers involved in this area. Normally, terrain and the other objects such as trees, buildings and roads are managed separately due to their different data representation. Speed and realism are two factors frequently considered in solving related terrain problems. At present, there are more than 800 commercial terrain visualization software have been developed by different vendors and developers as reported by United States Army Topographic Engineering Center (2005). Among the popular software are ESRI® ArcGIS, ERDAS® Imagine VirtualGIS, Geographic Resources Analysis Support System (GRASS) and Manifold System.
In this thesis, the research focuses on real-time 3D terrain data organization, visualization and on-the-fly navigation. There are two key issues have to be encountered in developing terrain-based applications in GIS field: spatial database management and fast interactive visualization (Kofler, 1998). These two obstacles take into account with the capabilities of current hardware technology that supports the system to be developed. Large amount of terrain dataset need to be handled efficiently in real-time environment with restricted main memory capacity. These include the management of geometric and texture data. At the same time, the interactive frame rate needs to be achieved consistently when facing with too many polygons generated for each frame. The detailed information on problem background is covered in Section 1.2.

There are many solutions have been proposed by a number of researchers in the last few years. Among of them are the implementation of hierarchical data structure, visibility culling methods and level of detail (LOD) techniques. Briefly, the most common types of data structures can be employed to manage data are Kd-tree (Bentley, 1975; Bittner et al., 2001; Langbein et al., 2003), quadtree (Samet, 1984; Röttger et al., 1998; Pajarola, 1998), R-tree (Guttman, 1984; Kofler, 1998; Yu et al., 1999; Zlatanova, 2000) and tiled block (VTP, 1999; Lindstrom et al., 1996; Pajarola, 1998; Ulrich, 2002). The difference is depending on how the space (data) is divided in hierarchical manner (Gaede and Günther, 1998).

For visibility culling, there are four types of methods can be adopted: backface culling (Laurila, 2004), small feature culling (Burns and Osfield, 2001; Assarsson, 2001), occlusion culling (Stewart, 1997; Mortensen, 2000; Martens, 2003) and view frustum culling (Hoff, 1996; Assarsson and Möller, 2000). These are useful for removing the unneeded data from being processed. One of the important components in culling methods is the intersection test. The related research works include Axis Aligned Bounding Box (Beckmann et al., 1990; Morley, 2000), Oriented Bounding Box (Gottschalk et al., 1996; Assarsson and Möller, 2000), Bounding Sphere (Hubbard, 1995; Dunlop, 2001; Morley, 2000; Picco, 2003) and k-Discrete Orientation Polytope (Klosowski et al., 1998).
LOD techniques are used to reduce the number of polygons and give details to the certain area in a scene. The examples of well-known techniques are Real-time Optimally Adapting Meshes (Duchaineau et al., 1997), Real-time Generation of Continuous LOD (Röttger et al., 1998), View-Dependent Progressive Meshes (Hoppe, 1998) and Out-of-core Terrain Visualization (Lindstrom and Pascucci, 2001). Although the LOD technique can reduce the problem complexity but lots of polygons that are out-of-sight still being rendered, thus deteriorating the system performance. Most of the abovementioned researchers have integrated their system with culling method at vertices or polygon level in order to solve the problem. However, lots of computations involved in implementing such method especially when facing with large amount of datasets.

In this thesis, a set of techniques has been designed and proposed for handling massive terrain dataset in interactive rate. The idea behind this research is to expand the capability of triangle-based LOD technique and improve the performance of real-time terrain visualization and navigation system. Triangle-based LOD technique has been chosen for this research because it has potential to remove lots of terrain geometry and capable to speed up the system at interactive rates without losing the visual appearance of visualization part (Duchaineau et al., 1997; Hoppe, 1998; Pajarola, 1998; Röttger et al., 1998; Lindstrom and Pascucci, 2001).

There are two separate phases involved, which are pre-processing and run-time processing. In first phase, the specific terrain data is converted to the internal data structure that is based on tile representation. The original terrain data is partitioned into several small tiles or patches. In run-time, several tiles are loaded depending on camera position. Then, these tiles are tested for determining the visibility status to remove the unseen block of data. Finally, the geometric simplification of visible terrain tiles is done which is based on triangle-based LOD technique in order to minimize the number of triangles to be processed and being rendered.
1.2 Problem Background

There are three challenging issues in developing terrain-based applications. These involve the massive terrain datasets, geometric complexity and capability of current triangle-based LOD techniques.

1.2.1 Massive Terrain Dataset

Various types of terrain datasets are widely being used in GIS applications, including Digital Elevation Model (DEM), Digital Terrain Elevation Data (DTED), satellite data and aerial photographs. Commonly, terrain consists of geometry and imagery data. Geometric data represents the location or coordinate of related terrain’s vertices, while imagery data represents as a texture map for producing realistic 3D model. In fact, large amount of this terrain dataset have to be faced when dealing with GIS applications. It is not surprising if the size of dataset will occupy a number of Gigabytes or even Terabytes of storage (Gaede and Günther, 1998).

The use of this massive dataset will affect the capability of main memory to load and store data for visualization. A number of researchers have pointed out that it is impossible to hold the entire detailed dataset at once (Kofler, 1998; Pajarola, 1998). Besides that, accessing this data quickly enough to use it in real-time rendering is a difficult problem. It will limit the performance of the system during interactive visualization. This is because when the new frame is generated, all polygons will be displayed whether they are within the view frustum or not. It is time consuming to compute and render the unneeded data continuously in real-time (Hill, 2002).
1.2.2 Geometric Complexity

For any detailed terrain model, the number of triangles that must be rendered is prohibitively high. For instance, a 100 km x 100 km terrain sampled at 1-meter resolution and represented as a regular grid will contain roughly 20,000,000,000 triangles. At present, it is not be able to render anywhere near this full resolution data in real-time. According to the research that has been done by Hill (2002) in managing large terrains, rendering 20,000,000,000 triangles per second would not be feasible until 2010. Furthermore, rendering this many triangles at 60 frames per second would not be possible until 2016. Figure 1.1 shows an example of visualizing original terrain model with billion of vertices and polygons. The white line shows the triangles generated in wireframe mode and the black color is a background of a scene.

Figure 1.1 Visualization of full resolution terrain model (number of vertices = 16,848,344,430 and number of triangles = 33,695,786,468)
With current graphics hardware, all communications between the card and the CPU takes place over the memory bus. Presently, the primary bottleneck related to rendering performance is the memory bandwidth on the bus. The graphics hardware is capable of rendering substantially more geometry per second than can be read in over the bus. In order to render 60 million triangles per second, over 5 Gigabytes of data would need to be transferred to the card each second. According to the maximum memory bandwidth and million triangles per second can be transferred to the graphics card as shown in Table 1.1, it is impossible to achieve the maximum rendering speed directly. Moreover, this bandwidth is also used for transferring texture data to the card, so the 60 million triangles per second target are even further out of reach (Hoppe, 1999; Hill, 2002).

Table 1.1: Types of busses and the corresponding memory bandwidth and geometric throughput (Hill, 2002; Peng et al., 2004)

<table>
<thead>
<tr>
<th>Type of Bus</th>
<th>Bandwidth Limitation (MB/s)</th>
<th>Geometric Throughput (Million triangles per second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCI</td>
<td>133</td>
<td>1.385</td>
</tr>
<tr>
<td>PCI 64</td>
<td>533</td>
<td>5.552</td>
</tr>
<tr>
<td>AGP 1x</td>
<td>266</td>
<td>2.771</td>
</tr>
<tr>
<td>AGP 2x</td>
<td>533</td>
<td>5.552</td>
</tr>
<tr>
<td>AGP 4x</td>
<td>1066</td>
<td>11.104</td>
</tr>
<tr>
<td>AGP 8x</td>
<td>2150</td>
<td>22.396</td>
</tr>
<tr>
<td>PCI Express</td>
<td>4300</td>
<td>44.792</td>
</tr>
</tbody>
</table>
1.2.3 Limited Capability of Triangle-Based LOD Techniques

In order to deal with so much dataset, level of detail (LOD) techniques need to be employed. Theoretically, terrain close to the viewer must be presented with more geometry than terrain that far away from the viewer; and rough terrain requires more geometry than flat terrain does. Frequently, triangle-based LOD algorithm is applied for simplifying the terrain geometry. Although this technique provides fast visualization through polygon simplification in real-time but it is not practical for very large terrain dataset because the amount of data to be loaded exceeds the data that can be stored into main memory. Even if the system is able to load the data, it is time expensive for waiting the loading process completed (Hill, 2002).

1.3 Problem Statements

According to the problems mentioned earlier, the following question needs to be solved in this research:

How to store and display large amount of terrain datasets efficiently for real-time visualization and navigation system with restricted capacity of current main memory and memory bus bandwidth?

1.4 Goal

The research goal is to expand the capabilities of triangle-based LOD technique by developing a set of techniques using hybrid method that will enhance the efficiency of real-time 3D terrain visualization and navigation for the development of Virtual GIS.
1.5 Objectives

In order to accomplish the main goal, several objectives have to be fulfilled. These include:

(i) To design and construct an efficient data structure that will be able to store arbitrary-sized terrain model.
(ii) To develop an algorithm that will be able to provide fast data loading and reduce the memory usage in run-time.
(iii) To develop a culling technique that will be able to remove large amount of terrain geometry.
(iv) To integrate aforementioned data structure, loading algorithm and culling technique with the chosen triangle-based LOD technique into the real-time terrain visualization and navigation system.

1.6 Scopes

The research focuses on managing large-scale terrain dataset in real-time. Several scopes need to be considered. These include:

(i) The system manages static terrain dataset. It does not include the buildings, trees, rocks and other related objects.
(ii) The system focuses on managing terrain geometry. Imagery terrain data is only used as a texture map for enhancing the visual appearance.
(iii) Terrain datasets are the regular grid representation.
(iv) Triangle-based LOD technique is based on view-dependent LOD technique.
(v) The system is developed in real-time environment, which involves user interaction through computer’s keyboard and mouse on desktop PC.
1.7 Organization of the Thesis

The thesis is organized into six chapters. Chapter 1 gives a general picture on related research fields, problem background and existing solutions involved in developing real-time terrain visualization and navigation in Virtual GIS. Besides that, the goal, objectives and scopes of this research have been specified.

Chapter 2 provides surveys on spatial data structures, visibility culling methods, intersection test techniques and view-dependent LOD techniques.

Chapter 3 presents the methodology of this research, which consists of four main steps. The explanations include partitioning original terrain data, loading tiles, culling invisible tiles and simplifying geometry of visible tiles.

Chapter 4 describes the implementation details of prototype system from pre-processing to run-time processing. For pre-processing, there are two pairs of steps have been explained, which are reading a DEM file and creating TXT files, and reading TXT files and creating TILE files. For run-time processing, the section has been divided into five subsections: creation of classes, initialization, rendering, interaction and shutdown procedure.

Chapter 5 presents and discusses the results obtained from the implementation phase. System specifications and testing data have been specified. Subsequently, the evaluations of the system and comparison among the existing technique have been explained in further details.

Chapter 6 summarizes the thesis and includes recommendations of future works that can be extended with the prototype system.
Moore’s law for CPU. Thus, the implementation of real-time terrain system using this processing can be well-suited for the next few years.

(iv) **Implementing parallel processing**

The prototype system developed in this research can be expanded to the networking environment. By adopting the client-server communication, parallel processing can be done in order to obtain higher level of system performance. The use of load balancing algorithm can be used to distribute the job efficiently based on priority and burden of processing power.
REFERENCES


Generation of Continuous Levels of Detail for Height Fields. Proceedings of
WSCG ’98. February 9-13. Plzen, Czech Republic: University of West
Bohemia, 315-322.


Samet, H. (1990). The Design and Analysis of Spatial Data Structures. Reading,
MA: Addison-Wesley.

Shade, J.W. (2004). View-Dependent Image-Based Techniques for Fast Rendering of

CIS 781.

systems in a multi-user environment. Proceedings of the 23rd Annual Hawaii
International Conference on System Sciences. January 2-5. Washington, DC,

Eurographics Workshop on Rendering Techniques ’97. June 16-18. London,
UK: Springer-Verlag, 217-228.

22, 27-41.

ACM Press, Course 35.


Interior, United States Geological Survey, National Mapping Division, USA:
National Mapping Program Technical Instructions.

http://www.vterrain.org/Textures/ground_textures.html

Guide. Canada: Addison-Wesley Longman, Inc.
